Impact Estimation: IT Priority Decisions

Lindsey Brodie

School of Science and Technology

Middlesex University

A thesis submitted to

the School of Science and Technology,

Middlesex University

in partial fulfilment for the degree of

Doctor of Philosophy

November 2014

Abstract

Given resource constraints, prioritization is a fundamental process within systems engineering to decide what to implement. However, there is little guidance about this process and existing IT prioritization methods have several problems, including failing to adequately cater for stakeholder value. In response to these issues, this research proposes an extension to an existing prioritization method, Impact Estimation (IE) to create Value Impact Estimation (VIE). VIE extends IE to cater for multiple stakeholder viewpoints and to move towards better capture of explicit stakeholder value. The use of metrics offers VIE the means of expressing stakeholder value that relates directly to real world data and so is informative to stakeholders and decision makers. Having been derived from prioritization factors found in the literature, stakeholder value has been developed into a multi-dimensional, composite concept, associated with other fundamental system concepts: objectives, requirements, designs, increment plans, increment deliverables and system contexts. VIE supports the prioritization process by showing where the stakeholder value resides for the proposed system changes. The prioritization method was proven to work by exposing it to three live projects, which served as case studies to this research. The use of the extended prioritization method was seen as very beneficial. Based on the three case studies, it is possible to say that the method produces two major benefits: the calculation of the stakeholder value to cost ratios (a form of ROI) and the system understanding gained through creating the VIE table.

Keywords: IT prioritization; stakeholder value; impact estimation; metrics; value-based systems engineering.

Acknowledgements

I am grateful to Middlesex University, which has enabled me to take these ideas to this level.

Special thanks go to my Director of Study, Dr. Elke Duncker for her support, patience and help, and to my supervisor, Prof. Juan Augusto for his support and guidance. Thanks too are due to Prof. Mark Woodman and Prof. Tony White for their assistance and insights along this journey.

The opportunity to carry out this research came about thanks to Ms. Elli Georgiadou and my late supervisor, Prof. Colin Tully, who were enthusiastic that I took that first initial step and apply to become a Research Student Tutor (RST) at Middlesex University. I am very grateful to Colin, who saw the potential in my embryonic, planned research and enabled me to invest time in pursuing those plans. Thanks also must be given to the School of Science and Technology at Middlesex University, who have supported me carrying out this research.

On arriving at Middlesex University, I found myself part of a team of RSTs. I have fond memories of the days when we were put though our teaching and PhD training, and then supported each other as we tussled with getting our early academic writing underway with the help of Elke. Special thanks for their friendship are due amongst others to Amala, Dili, Jim, Dmitri, Mary, Hany, Nalini, Fatema, Dhawal, Saeed and Javed, and of course, there must be mention of the engineering PhD student, who was a constant in our office, the late Dr. Michael Foster.

Thanks too are due to the research office at Middlesex University, especially to Prof. Dick Comley, Terri Demetriou and Pia Wallington for all their help and assistance.

Thanks also are due to Dr. Anthony Lucas-Smith for his on-going encouragement over the years after my MSc that I should continue on and complete more research.

Thanks too, to Tom Gilb for creating Planguage and Impact Estimation, which turned out to have such a vital part to play in this research. His ideas continue to be a source of inspiration.

The vital case studies could not have been carried out without the help and enthusiasm of Ryan Godridge, Harold Souter, Renze Zijlstra, Alan Goldberg, Terje Grimstad, Vanda Ridley and Juan Augusto. Their patience and willingness to be ‘guinea pigs’ in trialling my research is hugely appreciated and I look forward to taking these methods further with you all.

Finally, thanks are due to my family for their support throughout this process. Also friends like Linda and Lucy for always asking and listening, and my neighbour, Jackie for managing to drag me away from my computer into the fresh air for many a much-needed walk. Particular mention must go to my daughter Jane, who has been endlessly encouraging and exceedingly patient.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Research background and problem

The initial motivation for this research came from the author's observations while working as an information systems analyst. Senior management and IT staff were accepting feasibility studies and user requirements specifications for information systems that contained only 'broad brush' statements of the potential system benefits. This absence of detailed information seemed to result in failure by the organizational management to fully understand and communicate the expected system benefits to the IT staff. In turn, the selection of what requirements and designs to implement also lacked this input. This mattered because given limited systems development resources and very large numbers of requirements, priority decisions needed to be made over what to deliver and when. More detailed system benefits information could have contributed significantly as evidence for those decisions.

The aim of the research project reported here was to research the problem anecdotally described above, to address it and to provide a potential solution. For this purpose, the initial high-level research questions were:

- How can we improve communication between organizational management and IT staff, concerning system benefits?
- Is it possible to improve IT prioritization based on the potential value to be obtained, and what methods might we use to achieve this?

- How can better support for priority decision-making regarding IT investment be provided?

The basic idea is that collecting improved value data and using it appropriately in prioritization should result in greater system benefits being delivered.

In the sense that data is often only collected if it is to be used, the problem of the lack of detailed system benefits information appears to stem from the absence of any recognised prioritization method demanding its collection. Looking across IT industry guidance, there does appear to be a widespread issue with prioritization, in so far as the term is referred to within guidance such as SWEBOK, (Abran et al., 2004), but there is no explanation of the theory or methods by which such prioritization should actually be achieved. As such, there appears to be a gap concerning prioritization in the body of knowledge of software engineering. It seems high time to determine best practice IT prioritization methods and to establish the underlying theories they utilize.

An additional motivation for this research came from the author's use of Gilb's Impact Estimation (IE) (Gilb, 2005a). The IT corporate strategy unit within the computer manufacturer where the author worked had identified (early 1990s) Gilb's metrics (Gilb, 1988) as best practice for identifying and specifying objectives. The author expanded the brief in a number of cases to Gilb's IE method. Although the method showed promise, the author identified several issues, one main issue being that value could fail to be adequately captured. This observation resulted in an interest in solutions for the perceived issues with IE. This interest combined with the previously mentioned need for better prioritization methods motivated this research. Establishing what the academic literature could contribute to improving such prioritization seemed an appropriate next step.

Finally, an even bigger question could perhaps be posed as to whether the current levels of IT project failure could be due in part to inadequate prioritization: are perhaps the wrong requirements and the wrong designs being implemented and contributing towards failure? A 2004 report by a working group from the Royal Academy of Engineering and British Computer Society identifies several key success factors for complex IT projects as follows: "client-supplier relationship, evolutionary project management, requirements management, change management, measuring progress, contractual arrangements, risk management and technical issues" (RAE and BCS, 2004). Whilst this list can be considered rather high-level, it is of note that *prioritization* of stakeholders' requirements and designs for implementation, has a part to play in almost all of these success factors.

1.2 Current trends in software engineering

Several relevant current trends in software engineering can be identified that impact on this research into prioritization:

Value-based systems engineering: The increasing awareness of the need for value-based systems engineering, which demands that more attention is paid to stakeholder value. Boehm and Sullivan (2000) propose that current software developers fail to address 'value added' in adequate depth when designing systems. They outline the key research pointers to include methods for reasoning about investment, competitive advantage and change, that can resolve multi-attribute decision issues in software development, and models that allow consideration of benefits, opportunities, costs and risks, which cater with uncertainty and incomplete knowledge. Overall they want the integration of economic considerations into software methods. Further, in the introductory text to the Economics of Software and Computation workshop (ESC'07), Sullivan (2007) reiterates the lack of "formal, testable and tested theories, methods, and tools to support economic-based analysis and decision-making (and value-based analysis more broadly)." Research into prioritization for IT investment touches on most of these research pointers, especially if the topic of stakeholder value is given specific attention.

Rational versus intuitive decision-making: In recent years, there has been debate about rational versus intuitive decision-making in the context of management (Buchanan and O'Connell, 2006). As a fairly young discipline, systems engineering is caught up in this debate, as there are no well-established processes for priority decision-making. This is partly due to systems engineering originating from technology-orientated engineering focusing on hardware and software capabilities. However, as systems interact more and more with people in everyday life, attention has shifted to the people factors leaving many engineers lagging in the skills to integrate these factors.

While the people factors are indeed extremely important, the focus of this research is on improving the underlying rational processes supporting decision-making and so move people away from their current "informal" (Lehtola and Kauppinen, 2006) prioritization processes. The hope is that if the rational processes are better understood then improved support can be given to the decision-makers. Keeney (2013) states,

> "Typically, the substance relevant to complex problems comes from various fields and disciplines and from various individuals and organizations. There is no way to logically integrate all of this relevant information without a model that can be used for analysis."

Keeney's mention of "for analysis" raises the question of how the analysis is carried out. Some prioritization methods can be rather "black box" in their approach (Berander, 2007 quoting one of his research participants). However, the stance take in this research is to agree with Davis (2003) that any method/process followed is simply used to assist decision-making. Davis makes an additional valid point that decisions have to be based on more than "just mechanics": if stakeholders connect to a decision, they are more likely "to work towards a successful outcome." From the perspective of this research, the idea is that providing improved systems benefits information should help engage the stakeholders, and if the stakeholders understand and agree with the prioritization process, then they are more likely to give their support to the resulting decisions and act on them.

Not just requirements: Increasingly, requirements engineers are questioning whether simply focusing on the requirements is sufficient when considering the potential solutions to a systems problem. There is realisation that the impacts of the different designs have to be taken into account as well and that the requirements do not exist in isolation (Nuseibeh, 2001; Jarke et al., 2011). This research takes the stance that the argument for a holistic approach involving more than just the requirements data can actually be widened further. It should also include consideration of all contextual aspects, including operational use. Keeney and Raiffa

(1976) pointed this out several decades ago: "It is artificial to completely separate the identification and analysis of a problem from its implementation."

Agile methods: To some extent, agile methods are also contributing to this trend of a wider, holistic approach. However in its own right, the adoption in the last decade of agile methods (Abrahamsson et al., 2002) represents a major change in systems development working practices and presents practical challenges. Many aspects of project management processes can be considered to be still in a state of flux or have yet to fully adapt. With regard to IT prioritization, such development introduces the need for more numerous decisions spread throughout the systems implementation process (with the need for reprioritization to support these decisions). This impacts on the amount of time required, and the timeframe for involving management in decision-making (Hanssen and Faegri, 2006).

All these current trends underpin the importance and necessity of research into IT prioritization. They also serve to inform the direction of this research: key aspects to be considered being stakeholder value and coverage of the system concepts.

1.3 IT prioritization

The need for further research in the area of IT prioritization has been identified for some time. Reviewing the literature on prioritization, in a state-of-the-art paper, Zave (1995) identified "understanding priorities and ranges of satisfaction" as an issue for requirements engineering to address. Since then recognition has grown about the need for improved prioritization processes and methods. Karlsson et al. (1998, quoting Siddiqi and Shekaran, 1996) report "There is a growing acknowledgment in industrial software development that requirements are of varying importance. Yet there has been little progress to date, either theoretical or practical, on the mechanisms for prioritizing software requirements." Nuseibeh and Easterbrook (2000) identify requirements prioritization as a topic for further research in their roadmap for requirements engineering. Firesmith (2004) suggests a viewpoint for industry, that "the prioritization of requirements [is] a critically important part of requirements analysis … Unfortunately, there is little agreement within industry as to how, when, and why requirements should be prioritized."

Khan (2006) having carried out a survey on case studies of software requirements prioritization in the literature, reports a need for more empirical research on prioritization to be carried out. His findings include that much of the existing research has been carried out using under 20 requirements, that the requirements used tend to be high-level ones, that non-functional requirements[1] tend not to be prioritized (he found only one paper where they had been prioritized), and that research on requirements dependencies is lacking. Further, Lehtola (2006) reports a lack of research concerning whether existing prioritization methods are in use in industry and of their applicability. She suggests "it is even not clear if any of the techniques can solve the existing challenges in the area of requirements prioritization". A recent systematic literature review of software requirements prioritization (Achimugu et al., 2014) also reports a continued lack of research on using prioritization methods in industry. The authors suggest one of the problems is that the existing prioritization methods are too complex and time-consuming. Further they point out that most prioritization methods do not support communication amongst stakeholders.

It should be noted all these references are talking about the need for further research within *requirements prioritization*. While requirements prioritization dominates the literature on prioritization, it is not seen as the sole focus for this research. As discussed before, the stance taken by this research is that a holistic view should be taken and that the prioritization process should also include consideration of the other system concepts, such as design, increment and stakeholder value.

[1] The term 'non-functional requirements' is problematic as it is too negative (Woodward 2003). In this document, 'quality requirements' is generally used instead – except when reporting on use of the term by others.

1.4 Research aims and research question

Having identified IT prioritization as a major area of concern, the main research question of this research is as follows:

> **"How can better support be provided for priority decision-making regarding IT investment?"**

In the following chapter, the literature review will provide the conceptual resources to refine this research question and a number of propositions will be developed.

1.5 Contribution to knowledge

This research makes a contribution to knowledge as follows:

> **An extended model of IT prioritization is proposed, which takes into account the prioritization factors identified in the literature. This prioritization process is turned into a method, which in turn is exposed to live projects to provide proof of concept.**

1.6 Thesis structure

This introductory chapter has described the background to this research, discussed current trends within software engineering that support the perceived need for such research, presented an initial research question and outlined the contribution to knowledge.

Chapter 2 presents the literature review, an overview of existing IT prioritization methods and their theoretical underpinnings such as the system concepts and prioritization techniques utilized. Furthermore, it discusses the factors relevant to IT prioritization as identified in the literature in order to arrive at a more detailed research problem.

Chapter 3 uses the outcomes of the literature review to identify specific requirements and criteria for a successful prioritization process and proposes an IT prioritization method fulfilling these criteria.

Chapter 4 discusses relevant research methods for the research problem at hand. It describes the research design and justifies the use of case study research and presents the internal logic guiding the case studies.

Chapter 5 describes the case studies carried out and reports the findings for each case study with regards to the proposed IT prioritization method.

Chapter 6 aggregates and compares across the findings of the individual case studies to arrive at a concept of IT prioritization based on value analysis.

Chapter 7 concludes this research. The method of IT prioritization based on stakeholder value and its meaning for a number of academic and practitioner fields is discussed. It also outlines suggestions for further research.

Having identified a first cut research question, the following chapter will review the literature to mobilize theoretical resources for a new, IT prioritization method that takes stakeholder value into account.

Chapter 2

Literature Review

2.1 Introduction

This chapter presents a literature review including critical analysis. After establishing the scope of the review, an overview of the identified existing IT prioritization methods is provided, and their use of the prioritization techniques is analysed. The identified requirements for an IT prioritization process are then discussed, with specific focus on the coverage of the system concepts, and the identified prioritization factors and stakeholder value. Finally, the research question and its propositions are revised in the light of the findings of the review.

2.1.1 The initial research question

The initial research question for this research was introduced in Chapter 1. It is as follows:

> "How can better support be provided for priority decision-making regarding IT investment?"

2.1.2 Scope

Research in the area of IT prioritization increased from the late 1990s to 2010, but since then has reduced (Achimugu et al., 2014). The reasons for the increase in the late 1990s were perhaps the awareness that IT projects were not delivering to stakeholders' expectations (Jones, 1995; Flowers, 1996; Glass, 1998; Yardley, 2002), and that requirements prioritization had specifically been identified as an area for research interest (Zave, 1995; Nuseibeh and Easterbrook, 2000). Interest further increased from around 2006 and that could perhaps be due to the uptake of agile methods (Larman and Basili, 2003; Larman, 2004). Why research in the last three years has decreased somewhat is puzzling, but it could maybe be that a general lack of progress in the area of IT prioritization is an issue.

The years of main interest for this research are from 1997 to date, though it is noted that both the major prioritization methods discussed in the literature predate this: Analytic Hierarchy Process (AHP) (Saaty, 1990) originated in the early 1970s, and Quality Function Deployment (QFD) (Akao, 1990) originated in the late 1960s.

IT prioritization can be seen as a diverse subject spanning numerous system development lifecycle areas from strategy to implementation. The research found in the literature is typically fragmented into specialist areas. These include requirements prioritization (Karlsson et al., 1998; Moisiadis, 2002; Berander and Andrews, 2005), release planning (Greer and Ruhe, 2004), architecture selection (Kazman et al., 2001), COTS selection (Mohamed et al., 2007a), financial management (Favaro, 2002; Sivzattian, 2003), and decision-making and negotiation methods (Park et al., 1999). To report on this research, it is seen as essential to cover as wide a range of the literature as possible, but within each area to focus only on the aspects relevant to the determination of priority and value. Certain areas/approaches are specifically excluded because they are not seen as being of primary interest given the research focus. The scope of this literature review is therefore defined as follows:

- Investigation of the IT prioritization process
- Investigation of existing IT prioritization methods including prioritization of objectives, requirements, designs, implementation plans/increments, and other system concepts, such as stakeholder value.
- Financial accounting mechanisms, such as discounted cash flow, are not within scope. While recognised as being needed in priority decision-making, it is felt in the first instance that these are well-understood mechanisms.
- The area of cost estimation is not within scope. It is recognised it could be interfaced with for resource estimates

- The area of earned value management is not within scope as it measures conformance to the planned schedule and development budget (that is, if a project is on its planned track, then value has been delivered)
- In the first instance, mathematical models, such as those found in Multi-Criteria Decision-Making (MCDM) papers, are not of interest. The issue being that the variables used are typically at too high a level of abstraction (For example, the use of cost and value as variables (Durillo et al., 2009)). Once a prioritization process has been developed, it would be appropriate to see how the scope of the mathematical models maps to it
- Artificial Intelligence (AI) is out of scope, though it is recognised that it could provide mechanisms for dealing with heuristics
- The human computer interface (HCI) for IT prioritization methods is not within scope
- Existing project management/requirements management application tools in industry are not within scope. For example, a leading requirements tool, IBM's Rational DOORS[2] can be integrated with Rational Focal Point[3], which was originally developed by Karlsson and is based on the Analytic Hierarchy Process (AHP).

To direct the literature search and structure the literature review, an overall framework is devised, which identifies specific topics: see Figure 2.1. These topics are derived from the initial research question and consideration of IT prioritization as discussed in the literature. A specific focus is placed on the system concepts.

[2]See http://pic.dhe.ibm.com/infocenter/rfphelp/v6r5/index.jsp?topic=%2Fcom.ibm.rational.fp.help.doc%2Ftopics%2Fc_fp_doors_intro.html [Accessed September 13 2014]

[3] See http://www-03.ibm.com/software/products/en/ratifocapoin [Accessed September 13 2014]

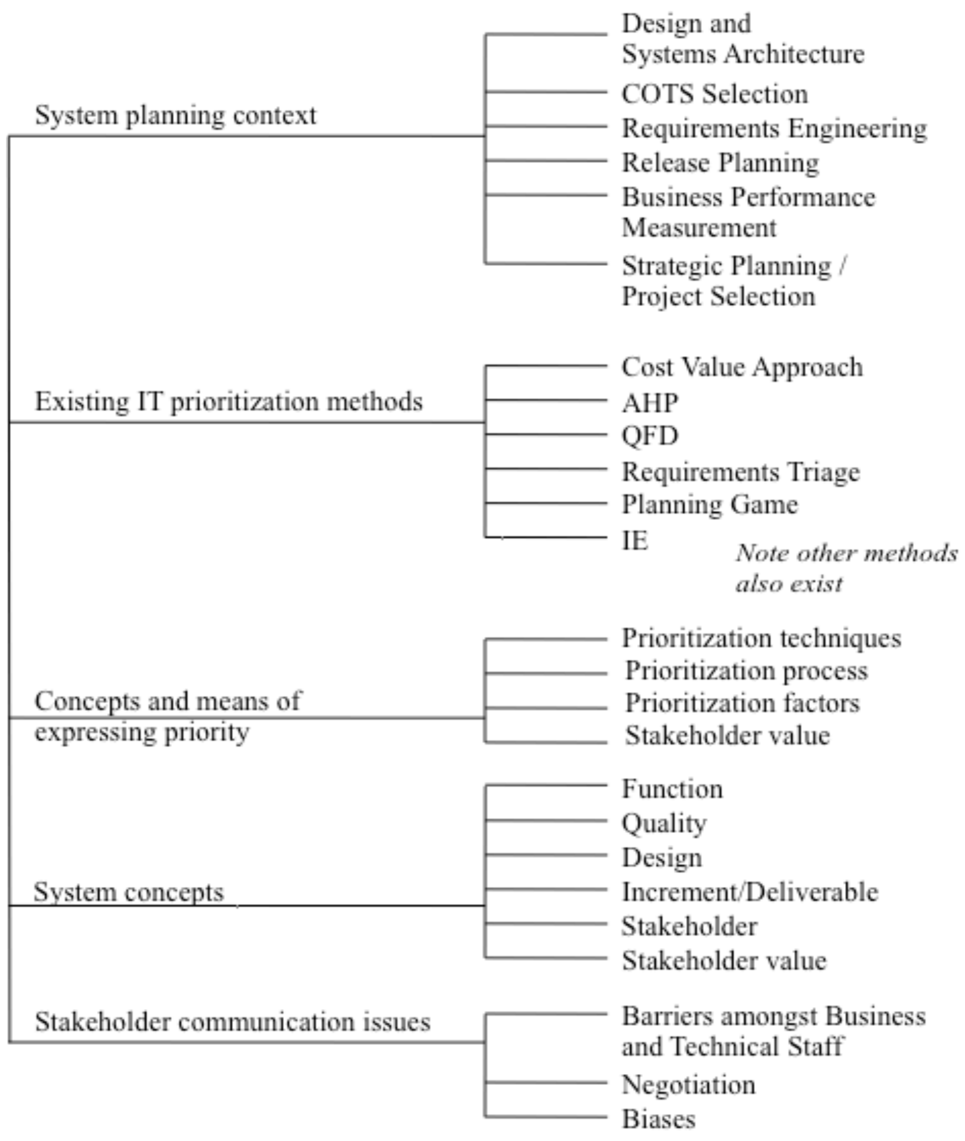

System planning context
Design and Systems Architecture
COTS Selection
Requirements Engineering
Release Planning
Business Performance Measurement
Strategic Planning / Project Selection
Existing IT prioritization methods
Cost Value Approach
AHP
QFD
Requirements Triage
Planning Game
IE
Note other methods also exist
Concepts and means of expressing priority
Prioritization techniques
Prioritization process
Prioritization factors
Stakeholder value
System concepts
Function
Quality
Design
Increment/Deliverable
Stakeholder
Stakeholder value
Stakeholder communication issues
Barriers amongst Business and Technical Staff
Negotiation
Biases

Figure 2.1. Specific topics for IT prioritization

Having identified the specialist areas in which IT prioritization methods are found, the next task is to find out more about these methods and what is reported on them.

2.2 Existing IT prioritization methods

Looking at the development of IT prioritization methods over the years, the most significant developments in chronological order are seen as including:

- In the late 1960s in Japan and then subsequently introduced into the USA and Europe in 1983, Mizuno and Akao (Akao, 1997) developed Quality Function Deployment (QFD). QFD was not conceived initially as an IT prioritization method (it was intended to address product quality), but its initial phase, HoQ (House of Quality) has been adopted as one.

- In the early 1970s, Saaty (1990) developed the Analytic Hierarchy Process (AHP)
- From the early 1980s, Gilb developed Impact Estimation (IE) (Gilb, 1988; 2005a) There were also earlier versions as the original idea was sparked by a talk given by Boehm in 1968 (Gilb, 2005a)
- In 1997, Karlsson and Ryan (1997) introduced their Cost-Value Approach based on AHP, which led on to the development of the Rational Focal Point application. Given its level of citation, (ibid) was an extremely influential paper, which can be seen as initiating interest in IT prioritization
- In 1999, Beck (2000) developed the Planning Game, as part of Extreme Programming (XP)
- In 2003, Davis (2003) introduced the idea of Requirements Triage. This only prioritized the requirements that needed further consideration, ignoring requirements that were essential or could be left for a later increment.

The use of AHP dominates much of the IT literature on requirements prioritization. Having carried out a survey of papers on software requirements prioritization, Khan (2006) concluded that AHP was "the most widely studied ... both in industry and academia." A more recent systematic literature review of software requirements prioritization reports that in the period 1996 to 2013, the top three most-cited methods are AHP, QFD and the Planning Game (Achimugu et al., 2014). AHP's influence on the literature is of even more note when you consider that AHP introduced the pair-wise comparison technique, which is often embedded in other methods, for example, the Cost-Value Approach method, which comes sixth in the most-cited methods' list of Achimugu et al. (ibid).

However, it is not clear to what extent all the prioritization methods are used by software development in industry, or how successful they have been (Lehtola, 2006; Achimugu et al., 2014). Indeed, there appear to be some problems with the take-up and continued use of some of the well-known prioritization methods, such as Quality

Function Deployment (QFD) (Martins and Aspinwall, 2001) and Analytic Hierarchy Process (AHP) (Lehtola and Kauppinen, 2006). Many organizations are relying on "informal" methods (Personal Communication; Lehtola and Kauppinen, 2006).

There are numerous different prioritization methods described in the literature. A selection categorized by their specialist area (that is, by the main system concepts used in their prioritization process) is as follows:

- **Requirements Prioritization:** MoSCoW (Stapleton, 2003), the Hundred-Dollar Test (Berander, 2007) and Requirements Prioritization Tool (RPT) (Moisiadis, 2002)
- **Requirements (and Effort) Prioritization:** Cost-Value Approach (Karlsson and Ryan, 1997) and Wiegers' Method (Lehtola and Kauppinen, 2006)
- **Requirements (and Design) Prioritization:** Analytic Hierarchy Process (AHP) (Saaty, 1990), House of Quality (HoQ) within Quality Function Deployment (QFD) (Cohen, 1995; Akao, 1997) and Impact Estimation (IE) (Gilb, 2005a)
- **Architecture (Design) Prioritization:** Cost Benefit Analysis Method (CBAM) (Kazman et al., 2001) and Reasoning Frameworks (Bass et al., 2005)
- **COTS (Design) Prioritization:** Procurement-Orientated Requirements Engineering (PORE) (Mohamed et al., 2007a) and Mismatch Handling for COTS Selection (MiHOS) (Mohamed et al., 2007b)
- **Release Planning:** Planning Game (Beck, 2000), EVOLVE/EVOLVE* (Greer and Ruhe, 2004; Saliu and Ruhe, 2005) and Requirements Triage (Davis, 2003)
- **Financial Prioritization:** Business Case Analysis/ROI (Favaro, 2003), Incremental Funding Method (IFM) (Denne and Cleland-Huang, 2004) and Real Options Analysis (Favaro, 2002)
- **Negotiation Prioritization:** Quantitative WinWin (Ruhe et al., 2003) and Distributed Collaborative Prioritization Tool (DCPT) (Park et al., 1999)
- **Others:** Conjoint Analysis (Green and Wind, 1975) and Even Swaps (Hammond et al., 1998).

Over 60 prioritization methods have been found in the literature, a selection of them is described briefly in Table 2.8 at the end of this chapter.

As a basis for further discussion about IT prioritization, seven prioritization methods are now briefly described. These methods are selected on the basis of representing different categories of prioritization method and/or their frequency of discussion in research papers. They are used later in this chapter as examples to support the critical analysis. They are as follows:

Quality Function Deployment (QFD) (Cohen, 1995): QFD consists of four phases, but often only the first phase known as the House of Quality (HoQ) is used. HoQ consists of a matrix structure of requirements against designs. The relative importance of the requirements is first established (AHP is sometimes used to achieve this). Then the relationship matrix is filled in by determining the impact of each of the designs on each of the requirements. The impacts are expressed using relationship values such as 'strong positive', 'medium positive', 'medium negative' and 'strong negative'. In addition to filling in the relationship matrix, the baseline values for the designs and relevant competitors' designs can be captured. The target values and additional information about difficulty of implementing and cost are also sometimes added for the designs. Additional information can also be captured against the requirements giving customer and sales preferences.

Analytic Hierarchy Process (AHP) (Saaty, 1990): In AHP, an aim and a criteria hierarchy are set up and then some potential designs (solutions) are identified. Pair-wise comparisons are worked through for all the criteria, then for the sub-criteria within the criteria, and this continues until the bottom of the criteria hierarchy is reached. Then normalized relative weightings are calculated for each of the criteria/sub-criteria (so the relative importance of each criteria/sub-criteria is known). Next pair-wise comparison of the designs is carried out against each of the lowest level of each branch's criteria/sub-criteria. Normalised relative values are then calculated. The normalized relative values are then multiplied by the relevant criteria/sub-criteria weighting and the results summed to give the next level's comparison figure. This multiplying and summing continues up the criteria hierarchy until the top is reached and a set of comparison values for the potential designs is obtained.

Impact Estimation (IE) (Gilb, 2005a): IE uses a matrix structure of requirements against designs. The requirements are expressed using the quality and resource (typically the development budget) requirements of the system concerned. Each quality requirement is described using a scale of measure and details captured of baseline and target levels. The estimated impact of each of the designs on each of the requirements is determined and expressed as the resulting level on the scale as an absolute value and as a percentage given the baseline level is 0% and the target level is 100%. Evidence to support each impact estimate is asked for and also the uncertainty in the estimate (a credibility rating (0.0 - 1.0) ranging from 'This is a guess' (0.0), to 'Another project has achieved this before' (0.8) to 'I have done this' (1.0). The percentages for quality can then be summed vertically to see the contribution of the individual designs towards meeting the requirements, and quality to cost ratios can be calculated using the data on the design's resource attributes (the costs). By summing horizontally the contribution of all the designs towards meeting each of the requirements some idea of the level of risk involved in meeting the quality and resource requirements can be assessed.

Cost-Value Approach (Karlsson and Ryan, 1997): Karlsson and Ryan developed a cost-value approach for prioritizing requirements based on AHP. "Customers and users" carry out pair-wise comparisons of the requirements and then software engineers estimate the relative cost of implementing each requirement. AHP calculations are then carried out and a diagram showing relative value against implementation cost for each requirement is created (value is on the y-axis and implementation cost on the x-axis).

MoSCoW (Stapleton, 2003): MoSCoW separates requirements into priority groups of 'must have', 'should have', 'could have' and 'want to have but will not have this time round'.

Planning Game (Beck, 2000): In the Planning Game, the business people write a set of user stories. Then the user stories are classified by business value into 'essential', 'less essential, but of significant business value' and 'nice to have'. Development people then estimate the amount of effort (often using story points: 1 story point = 1 day) and classify the stories by risk into 'can estimate the effort precisely', 'can estimate reasonably well' and 'cannot estimate at all'. Then development set the velocity for the implementation, and business people select the scope and select the appropriate user stories.

Requirements Triage (Davis, 2003): The aim is to establish what requirements need to be in the next increment, what requirements will not be in the next increment, and which are optional and so should be triaged. The requirements for triage have to be prioritized by relative importance by the relevant customer stakeholders and any dependencies noted. Relative importance is determined using the Hundred-Dollar test (Berander, 2007) or by the stakeholders voting for each requirement on whether it should be included or excluded based on how useful they see a requirement.

The resources needed for each of the requirements have then to be estimated by the development team and any resource dependencies noted. Finally, a subset of the requirements has to be chosen that “optimizes the probability of the product’s success in its intended market”.

At this point it is appropriate to discuss the different ways in which priority is expressed within the prioritization methods by the use of prioritization techniques, and this is the subject of the next section.

2.3 The different prioritization techniques – how priority is expressed

How priority is expressed is a key factor in a prioritization method. If insufficient or inaccurate detail is captured about priority then any subsequent processing using the information is jeopardised. Several different ways of expressing prioritization data can be identified, termed prioritization techniques (Berander, 2007). These techniques capture the prioritization data on several different types of scale: nominal, ordinal, interval, ratio and absolute (Kaposi and Myers, 1994; Fenton and Pfleeger, 1998). This is specifically important as the type of scale used determines the extent to which arithmetic can be used on the data.

The prioritization techniques can be categorized under four main types, namely grouping, ranking, weighting and metrics (Brodie and Woodman, 2011):

Grouping: The individual items being prioritized are each categorized into one of a set of priority groups. The results are on an ordinal scale of measure as although there is a clear priority order expressed amongst the groups, there is no information about the differentials. There are several concerns expressed: (*Note requirements are being prioritized in most of the discussions below, so for accuracy in reporting, the term 'requirements' is retained. The comments can however also apply when prioritizing other system concepts.*)

- There are problems over how exactly the different priority group names are interpreted (Lehtola and Kauppinen, 2006). Different stakeholders could be using different meanings unless there is some definition made available to clarify things.
- Another issue is that all the requirements in a given group have the same priority making it difficult to select amongst them (Berander and Andrews, 2005). Therefore this method does not scale up well as the priority groups grow too large to be meaningful.
- Stakeholders tend to believe that all their requirements are necessary and will therefore allocate the requirements on a 85% : 10% : 5% basis (Berander and Andrews, 2005). Wiegers (1999) suggests the percentage of classified essential requirements is likely to be even higher at 85-90%. One suggested way of resolving this is to force a reasonable distribution by fixing the ratio of the number of entries in the different groups, but this can lead to other problems, for example by forcing some misallocation (ibid). This is sometimes called "Forced Priority Groups" (Lehtola, 2006)
- In terms of several stakeholders being involved in the process, it is assumed that they will gain consensus over the allocations to the priority groups, but such agreement is not always possible in practice. It is impossible to combine different priority group allocations.

Ranking: The stakeholders have to sequence the requirements into a preference order. Ideally each rank is unique. There is no indication of the relative difference between ranks, so this method provides ordinal scale measures. Reported issues include:

- If there are several stakeholders then using the ranking method becomes problematical if stakeholders differ over the rankings. Calculating the mean priority has been proposed as one possible solution for this (though there might then be some shared ranks and in certain cases, priority rankings could get damped too much, say only one stakeholder wanted a specific requirement). However, arithmetically this is suspect as there are no fixed intervals between ranks.
- Another issue is that this technique does not scale up very well. Handling a lengthy list of ranked items is cumbersome.

Ranking is not a commonly used prioritization technique, however it is used by the methods of bubble sort (Berander and Andrews, 2005), binary search tree (Karlsson et al., 1998) and even swaps (Hammond et al., 1998).

Weighting: Stakeholders assign their preferences and relative weightings are calculated and sometimes normalized. The results are on a ratio scale.

Voting[4] (Berander and Andrews, 2005): Stakeholders are each given say, 100 points to allocate amongst the requirements being prioritized. The voting can be carried out several times, each time reflecting a different aspect (usually importance, cost and risk). The results are given on a ratio scale. Points raised include:

- There are issues with allocating points if there are numerous requirements as 100 can only be split in so many ways. To remove this, a greater number of points (say 100,000) can be allocated for distribution (Regnell et al., 2001).
- A more serious issue is if stakeholders decide to act irrationally. They could for instance put all their points on their favourite requirement. One way to avoid this is to set a cap on the number of points awarded to any

[4] Voting as used here is rather an unfortunate term as it can be confused with say, Top Ten Selection where stakeholders vote or specify which are their preferred ten requirements. The voting in Top Ten Selection results in two priority groups (the prioritization technique therefore being grouping, not weighting).

one requirement, but that could also skew the outcomes (Leffingwell and Widrig, 1999).

- Mead (2006) suggests that voting only works for one attempt as if the voters see the outcome of the first attempt then they are likely to use tactics in any subsequent attempts to move their requirements up the priority order.
- One benefit mentioned is that stakeholders can choose to give a requirement zero points (Berander, 2007). This is unable to be done when using pair-wise comparison.

Pair-wise comparison (Saaty, 1990): Pair-wise comparisons are made for all pairs of requirements utilising comparison values in the range 1-9 (with 9 representing highest importance. Typical meanings are 1= equal importance, 3=slightly more important, 5=essentially more important, 7=demonstrated importance, 9=extremely more important (Moisiadis, 2002)). Stakeholders work through carrying out all the pair-wise comparisons and then normalised relative weightings are calculated. (Karlsson and Ryan, 1997) sets out a detailed explanation of how pair-wise comparison works. Results are on a ratio scale.

There are concerns expressed about pair-wise comparison:

- The main concern expressed is the large number of pair-wise comparisons required (Karlsson et al., 1998). For n requirements in a hierarchical level grouping, n(n-1)/2 pair-wise comparisons are required.
- If there are a large number of requirements, they must be divided into hierarchical sets of similar requirements to reduce the number of comparisons required. Moisiadis (2002) reports Saaty stating that anything over seven requirements in a hierarchical set is too great and is prone to inconsistencies in the results.
- To reduce the number of pair-wise comparisons, Karlsson, Wohlin and Regnell (1998) suggest removing the need to undertake the reciprocal pair-wise comparison. However, while reducing the number of comparisons, this also removes the consistency checking of the stakeholder’s input by removing the redundancy in the comparisons.

- Perez, Jimeno and Mokotoff (2001) have identified that rank reversal can also occur when a problem is reworked after an additional "non-contributing" solution is added. Saaty (1997) is adamant that rank reversal is to be expected on occasion.
- Reprioritization is a further problem. Where there are n requirements, adding a new requirement leads to (n-1) new pair-wise comparisons being necessary (and this can mean having to return to ask all the stakeholders involved). Having obtained the additional comparisons, then recalculation of the priorities has to be done. Automated tools (such as the Expert Choice application) are considered to help ease this problem (Botta and Bahill, 2007).
- Moisiadis (2002) also discusses that dependencies amongst requirements are not completely catered for as inter-relations of requirements are only handled through the use of hierarchies (and not all dependencies can be modeled within such hierarchical structures). While handling of interdependencies is a more general issue for prioritization, it is worth specific consideration within pair-wise comparison because the hierarchical structuring is embedded into the technique.
- In a similar way to the priority group categories not being adequately explained to the stakeholders, there often is no rationale provided for the comparison values ("preferences") (Moisiadis, 2002).
- The use of a 1-9 comparison scale causes complications for many stakeholders (Moisiadis, 2002). Lehtola and Kauppinen (2006) discovered in one case study that in practice "even 5 steps was too much for users". None of their users used more than 3 comparison values when prioritizing a group of requirements.
- Another issue reported is the lack of visibility of the results throughout the process. This means there is no opportunity for the stakeholders to steer the results whilst the method is being worked through. It is also not easy to see how the results obtained are accounted for. The stakeholders can sometimes disagree with the results obtained and request changes in the prioritization order (Lehtola and Kauppinen, 2006). Of course, there is an argument that it is a good thing if stakeholders are unable to

influence the outcome, but obviously there is a problem if the method is capable to providing the wrong outcome as far as the stakeholders are concerned (unless it is just a case of conflicting opinions).

Metrics: Absolute scales of measure, sometimes referred to as 'counting', are used to express certain attributes and then numerical levels on the scales are assigned. These metrics form the basis for priority selection, for example by enabling calculation of ROI figures (Firesmith, 2004). ROI calculation needs data on the amount of benefit (stakeholder value) that would be achieved by implementing a given design and the implementation cost associated with it. Only absolute scale data enables such ROI estimates to be calculated, as explicit stakeholder data such as the requirement is "a cost saving over the next year of 220,000 monetary units" and "the implementation cost of a potential design is 50,000 monetary units" can be captured, which allows the estimated ROI to be calculated.

To date there is little discussion in the IT prioritization literature about the use of metrics for expressing priority. So the lack of any reported issues needs to be viewed with some caution. However, measuring real-life system data is tangible, and as such warrants further attention with regards its use in prioritization methods.

From the identified problems above, it is apparent that the existing prioritization techniques of grouping, ranking and weighting have some significant issues. Analysing the underlying origins for these issues, they can be categorized as either being due to the prioritization data, the way that the stakeholders interact with the prioritization technique or the inherent nature of data structures (abstraction, hierarchies and interdependencies). The next three sections explore these issues in more depth.

2.4 Comparing the prioritization data resulting from using the different prioritization techniques

Each of the different prioritization techniques creates prioritization data, which expresses priority in different ways. As discussed above, this leads to a number of issues.

Table 2.1 looks at the prioritization data across the types of prioritization technique and some of the main issues raised. These issues can be summarised as failure to capture explicit stakeholder value, a lack of support for multiple stakeholder viewpoints, the amount of effort required to achieve reprioritization, and inability to handle scaling up.

From Table 2.1's top row, *Example(s) of captured prioritization data*, it can be seen that grouping, ranking and weighting all create new data attributes to capture the priority assignments (priority group, rank or weight respectively). One issue being interpreting what the various assignments mean: for example, there is a need to understand how *Medium*, *2* or *0.3* should be interpreted. Of course, an interpretation of *Medium*, *2* or *0.3* could be provided to aid assignment and understanding, or the actual rationale behind the priority assignment could be captured. However, none of the methods make explicit provision to capture such rationale (Park et al., 1999). Without such explanation, checking, auditing and reprioritization of priority is made difficult, if not impossible, without returning to the stakeholder(s) making the original assignments.

Metrics, as implemented within the IE method, however take a completely different approach over the prioritization data. Priority is expressed in an indirect way as the change in the quality attributes required from the current (past) levels to the target (goal) levels. This does not explicitly state the priority, but results in data that has more meaning; the view being taken that if the target levels are correct, then all have a priority until they are reached (Gilb and Maier, 2005). In the absence of any further explicit stakeholder value, this mechanism of expressing priority works (that is, as long as you are working towards delivering some required target then the work is valid). The question though is whether some explicit stakeholder value data could be linked to the target levels to understand where the high stakeholder value resides? Looking at the example given in Table 2.1 under metrics, it can be seen (in the priority and stakeholder value line) that with additional information, the metric can be used to calculate explicit stakeholder value. This is estimating actual business benefits, which looks an attractive option, as once again it is data that the stakeholders can understand and discuss.

For grouping, ranking and weighting, any concept of stakeholder value is typically held implicitly within the priority assignment: as discussed above exactly on what stakeholder value basis the priority was assigned is not captured. Grouping and ranking seek to convey the value through the assigned priority group or the priority ranking respectively. Weighting (ratio scale data) provides more information as the priorities are in a proportional ratio with each other.

Table 2.1: Expressing priority using the different types of prioritization technique

Type of Prioritization Technique >	**Grouping**	**Ranking**	**Weighting**	**Metrics**
Example(s) of captured prioritization data	High, Medium or Low	1, 2, 3, ... N	30/100 or 0.30	Time to carry out task to be reduced from 1 day to 5 minutes
Priority and Stakeholder Value	Priority is say, Medium and the related stakeholder value is implicit	Priority is say, 2 and the related stakeholder value is implicit	Priority is 30% of whatever 100% equates to - the issue being what 100% represents as the priority is just apportioned across the requirements. Stakeholder value is implicit	The priority is expressed as the need to achieve all the specified target levels. Depends on the metric. While usually not totally explicit, an understanding of the likely stakeholder value is provided and further estimation is supported. For the metric given above, the estimated time or effort saved could be used and/or an estimate of cost savings is able to be calculated if say, monetary rate of pay is known and the workload
Multiple Stakeholder Viewpoints on Priority (Note here assuming 2 stakeholders)	Would be captured as say, Medium and High. Can't be combined	Would be represented as say, 4 and 15. Can't be combined	Would be represented as 30/100 and 2/100. Suggest do not aggregate different viewpoints	Time to carry out task to be reduced from 1 day to 5 minutes or to 2 hours. Suggest do not aggregate different viewpoints
Reprioritization	Y Add a new requirement by assigning to group. No extra effort	Y Would have to re-examine existing ranks. Depends on amount of change	(Y) Considerable effort required by stakeholders involving pairwise comparison or voting considering existing data and recalculation	Y Add to existing data and recalculate
Scaling up	N Too many in a group	N Difficult to keep track of numerous rankings	N Considerable effort to carry out numerous pair-wise comparisons or votes	(Y) Would select key requirements
Key: Y=Yes, (Y)=Yes, but with some difficulty N=No				

How the different prioritization techniques are able to handle combining priorities for multiple stakeholder viewpoints also varies. However, averaging the priority assignments across differing stakeholder viewpoints has little logic (see next section), but averaging stakeholder views within the same stakeholder group (the same viewpoint) could be a way forward. Looking at the arithmetic possibilities, for grouping, it is not valid to average the assignments to different priority groups – *High* and *Medium* do not add and anyway there is no priority group in the middle ground between them. For ranking, it is also not valid - if one stakeholder thought the ranking was 4 and another thought 15, splitting the difference is not going to have any accuracy as there is no fixed interval between ranks (Berander and Andrews, 2005). Weighting and metrics will support averaging arithmetically and averaging is valid if the stakeholders share the same viewpoint and the same rationale. If they don't, there's the danger of dampening down a valid high priority just because say, only one stakeholder viewpoint values it highly.

Regarding reprioritization, the weighting prioritization technique has the most difficulty. For example, if using pair-wise comparison and there is an additional requirement added, then additional comparisons are needed against all the existing requirements in that part of the hierarchy followed by recalculation. For grouping and ranking it is a case of understanding which assignment to make, there is no recalculation required. For metrics, an additional requirement could simply be added to the list of requirements, but then the ROI calculations would have to be reworked (but this would not take as much effort as with the pair-wise comparisons).

Over scaling up, both grouping and ranking fail to cope when there are a large number of requirements. The priority groups grow too big and become a pool of requirements of the same priority, and managing the priorities in a long list of ranked requirements becomes unwieldy. Weighting techniques also suffer badly on scaling up. As discussed earlier, with voting, the weighting scale has to increase to allow sensible assignments (Regnell et al., 2001) and with pair-wise comparison, the number of comparisons grows too large (Karlsson and Ryan, 1997). Use of abstraction can help or focusing on only the key requirements. Metrics can capture the numerous requirements, however as shall be discussed later, its technique for coping with a large numbers of requirements is, as with weighting, some combination of using abstraction and selecting only the key requirements.

The use of the different types of prioritization techniques and the specific prioritization techniques by the selected IT prioritization methods is showed in Table 2.2. This table also shows the scale types against the prioritization techniques. Of the selected methods, grouping and weighting with ordinal and ratio scale types are the most commonly used. Only IE uses metrics with the absolute scale type in its prioritization process (QFD does capture certain metrics, but these are not used in its prioritization processing, rather they are additional pieces of information that can be referred to).

Table 2.2: Mapping of the types of prioritization technique, the prioritization technique(s) and scale type(s) to the selected IT prioritization methods

Prioritization Method	**Type(s) of Prioritization Technique**	**Prioritization Technique(s)**	**Scale Type(s)**
QFD	Weighting and Grouping	Pair-wise Comparison and Priority Groups	Ratio Ordinal
AHP	Weighting	Pair-wise Comparison	Ratio
IE	Metrics (Counting)	Metrics Measurement	Absolute
Cost-Value Approach	Weighting	Pair-wise Comparison	Ratio
MoSCoW	Grouping	Priority Groups	Ordinal
Planning Game	Grouping	Priority Groups	Ordinal
Requirements Triage	Grouping and Weighting	Priority Groups and Voting	Ordinal Ratio

2.5 Stakeholder issues over expressing priority

Within the prioritization technique issues, there are several relating to how stakeholders interact with the techniques. The issues of stakeholders being able to understand the assignment options in order to make valid assignments and providing their rationale for making their assignments has already been discussed. There are in addition, views expressing that stakeholders tend to consider that all their requirements are important and that they will be likely to game to ensure their requirements obtain high priority.

There is also a concern that multiple stakeholders are catered for. The requirement to support multiple stakeholder viewpoints has been recognized for some time (Park et al., 1999; Moisiadis, 2002; Grünbacher et al., 2006). However few methods cope with multiple stakeholders and there are issues with the way some methods handle them (See Table 2.3). The majority of prioritization methods seem to make the assumption that the stakeholders can confer and agree their priority assignment *prior* to it being captured within a method. However in practice, this is often unlikely to be possible, especially if there are numerous diverse stakeholders. Ideally handling multiple stakeholder viewpoints should be part of the prioritization process with the

Table 2.3: How the selected IT prioritization methods address selected prioritization data issues

Key: **Y = Yes** **(Y) = Qualified Yes** **((Y)) = Yes, but only with considerable extra effort** **N = No** **(N) = Qualified No**	**Prioritization Method**						
Prioritization Data Issues	House of Quality (HoQ) within Quality Function Deployment (QFD)	Analytic Hierarchy Process (AHP)	Impact Estimation (IE)	Cost-Value Approach	MoSCoW	Planning Game	Requirements Triage
Explicit Stakeholder Value	N	N	(N)	N	N	N	N
Multiple Stakeholder Viewpoints	N	N	N	N	N	N	N
Reprioritization	((Y))	((Y))	(Y)	((Y))	Y	Y	((Y))
Scaling Up	N	N	(Y)	N	N	N	N

relevant evidence being captured and processed to help support subsequent stakeholder negotiation make the tradeoffs.

Regarding multiple stakeholders, there are two different situations that require handling: stakeholders who share the same viewpoint, and those who have different viewpoints. As mentioned previously, averaging priorities from the same stakeholder viewpoint is acceptable, but priorities from different stakeholder viewpoints need to be handled by negotiation and tradeoffs, as averaging is invalid. However, there are methods that suggest such averaging. Some methods as well as using such averaging are assigning weightings to job roles, that is, they adjust the weightings according to who assigned the priority (Regnell et al., 2001). This has to be questioned as to its accuracy because there is a risk for example, of mixing up power and business process knowledge.

In terms of stakeholders being able to express a valid opinion, Park et al. (1999) raise the issue that some of their users were not able to vote on both importance and difficulty. They point out a typical user would not be able to estimate the amount of effort (cost) required to implement a requirement. This is an important issue, as many methods require stakeholders to make priority assessments across all the requirements including areas where they lack knowledge. Ideally stakeholders should only provide input where they have sufficient knowledge.

In a further twist on stakeholders being able to communicate and negotiate, Faulk et al. (2000) state that in their experience “effective communication” between the business and technical sides of an organization is very uncommon. They consider that a product specification written for the business will have lost all the business rationale by the time it is translated into a technical specification. They consider the separation of the different organizational units in terms of culture, language and understanding of the goals hinders the flow of information. Moisiadis (2002) also perceives the same problem and to address it, he proposes a requirements prioritization method, RPT to introduce more connection between the business objectives and the “high-level requirements”. This aims to promote more discussion between the business and systems development, or at least ensure the link between business objectives and requirements is made.

There is in addition, the aspect that different stakeholders are likely to obtain differing stakeholder value from any system feature. In some cases there will be conflicting viewpoints. Again the prioritization process must be able to capture and handle such information.

Finally, there are stakeholder usability aspects to IT prioritization. Lehtola and Kauppinen (2006) consider any method should be easy for the stakeholders to use, and to use correctly. In fact, several researchers identify that different prioritization methods have different levels of usability. Questions are raised over the methods concerning their efficiency and effectiveness, their ease of learning and use, the acceptability of their use, and the transparency of both the processing and the results. Lehtola and Kauppinen (2006) consider the stakeholders have to sufficiently understand the data and the prioritization process to have trust in it.

In addition, there are the issues concerning not all the data being available (not collected or not in existence) or captured (not input into a technique/method) or having the correct values (for example, miscalculations, typos, biased information (Moisiadis, 2002), or out of date information). Boehm and Sullivan (2000) consider there will always be some uncertainties and missing data. Moisiadis (2002) suggests the prioritization process should go further and try to reduce the risk of any missing important requirements.

2.6 Data structure issues over expressing priority

There are additional prioritization issues reported that are due to the innate nature of the data being handled. Abstraction, data hierarchies (including matrix structures) and data interdependency all concern how data is structured.

Abstraction: There are problems prioritizing requirements, which are at different levels of abstraction (Moisiadis, 2002; Gorshek and Wohlin, 2006; Lehtola and Kauppinen, 2006). For example, stakeholders tend to express priority preference for the higher level, more abstract requirement (Lehtola and Kauppinen, 2006). There is an additional issue in that more abstract requirements also tend to be more ambiguous about their precise content, and that is not helpful when prioritizing.

Hierarchies: The system specification data is often hierarchical in nature with more abstract items towards the top of a hierarchy and more detailed items towards the bottom. This presents problems with many methods if differing levels of the same hierarchy have both to be prioritized because there is overlap: the higher item includes the lower item. This could ultimately result within prioritization in double counting.

A further problem is when there is imbalance within a hierarchy with some branches considerably more important than others. Only if a ratio scale technique is being used can this be adequately handled (Berander, 2007).

Matrix relationships: There are additional relationships between requirements, that Dahlstedt and Persson (2005) term cross-structure relationships. This presents issues as it is difficult to keep track of these relationships and assess their impact on priorities.

Interdependencies: There are interdependencies amongst the requirements and also amongst the designs. Requirements interdependencies are identified to be of varying kinds (Pohl, 1996; Dahlstedt and Persson, 2005; Zhang et al., 2014). Zhang et al. (2014) identify 6 dependency types split into two categories: intrinsic dependencies – business, implementation, structure and evolution, and additional dependencies – value and cost. For example, implementation dependency is where one item has to be implemented before another; cost dependency exists where an item affects the cost of implementing another item; a business dependency could be because an additional item has to be present to please a customer. Such interdependency can constrain both precedence and impact stakeholder value.

Note that using dependency models such as the one proposed by Zhang et al. (2014) captures hierarchical and cross-structure relationships under the umbrella of dependency. While some of the abstraction issues will fall under hierarchical and cross-structure dependency relationships, the problem of comparing the priorities of a high abstraction requirement with an unrelated, detailed one still remains a separate issue.

Dahlstedt and Persson (2005) consider priority should be handled distinct from dependency, they suggest dependencies have to be considered as well as priority, "Requirements can hence not be selected based solely on priority". The question can be asked whether consideration of dependency should perhaps be part of the prioritization process?

It seems essential that requirements specification captures the various interdependencies as the requirements are gathered – a prioritization process has to have the basic information available to be able to account for such data interdependencies. Dahlstedt and Persson (2005) raise the issue that capturing all the interdependencies is “difficult and time-consuming to maintain”. They consider further research is needed, but suggest that maybe just capturing the information for the critical requirements would be a solution. Zhang et al. (2014) consider more empirical research is needed to identify which dependencies are the most significant to capture. Moisiadis (2002) makes a practical suggestion to address requirements interdependency by bundling together the requirements where it makes little sense to implement them separately.

At this point the discussion of the reported literature on IT prioritization and prioritization techniques now concludes – in many ways it represents a ‘bottom up’ approach. Attention now turns to looking ‘top down’ at the overarching IT prioritization process.

2.7 Towards an IT prioritization process

This section reports on the identified requirements for an IT prioritization process and the identified prioritization factors that an IT prioritization process would have to cater for.

Typically, the need for prioritization is seen as driven by resource constraints: the available timescales, financial budget and staff effort (Gilb and Maier, 2005). The prioritization process itself needs to be efficient and cost-effective (Karlsson et al., 1998) with the level of benefits involved dictating the amount of effort expended on prioritization (Moisiadis, 2002).

Considering the trends in software engineering discussed earlier in the Introduction (Chapter 1), certain requirements for an IT prioritization process can be established. Assuming a rational decision-making approach is adopted, there is a requirement for a logical, evidence-based process (Park et al., 1999). Moisiadis (2002) states that the resulting priorities should always be capable of validation with the rationale behind the priorities captured. Lehtola and Kauppinen (2006) agree that the decisions made should always be underpinned by data.

The trends discussed earlier in the Introduction (Chapter 1) of considering designs as well as requirements, and adopting agile methods, are both drivers for taking a holistic view across the system concepts and as a result, introducing a more integrated approach to prioritization. Further, the fragmented specialized areas of the existing prioritization methods spread out across the entire system lifecycle beg the question as to whether a more integrated approach could be found consolidating them. Little discussion of an overall prioritization process integrating these areas has been found to date; this is a weakness in systems development. It motivates the approach taken by this research in investigating the fundamental system concepts involved in prioritization: only by further developing the underlying theory are the requirements for IT prioritization methods to be better understood.

Much of the current prioritization simply considers the functional requirements, which is limiting. Several authors remark on the lack of handling of quality requirements in the prioritization methods (Moisiadis, 2002; Firesmith, 2004; Ozkaya et al., 2007). Ozkaya, Kazman and Klein are specifically concerned about the lack of research into quality requirements as a driver for generating and assessing value. In practice, several methods can actually handle quality requirements

Table 2.4: Use of the main system concepts by selected IT prioritization methods

Key: **Y = Yes supported** **(Y) = Qualified Yes** **N = Not supported** **(N) = Qualified No** **H = By use of hierarchy** **V = One viewpoint only** **(M) = Metrics provide some indication** **I = Implicit** **System Concepts**	**Prioritization Method** House of Quality (HoQ) within Quality Function Deployment (QFD)	Analytic Hierarchy Process (AHP)	Impact Estimation (IE)	Cost-Value Approach	MoSCoW	Planning Game	Requirements Triage
Stakeholder	V	V	(V)	V	V	V	V
Stakeholder Value	I	I	(M)	I	I	I	I
Objective	N	H	H	N	N	N	N
Functional Requirement	Y	Y	(N)	Y	Y	Y	Y
Quality Requirement	Y	Y	Y	Y	Y	Y	Y
Design	Y	Y	Y	N	N	N	N
Impact of Design	Y	Y	Y	N	N	N	N
Increment/Deliverable	(N)	(N)	Y	N	N	N	N
Results of Delivery	N	N	Y	N	N	N	N

alongside functional requirements, for example, AHP, QFD and IE, though it is probably true that their handling of the quality requirements is limited, with the exception of IE. A further aspect to consider is the integration of functional and quality requirements as quality levels often vary across functionality (see further discussion of IE in Chapter 4).

To date, no comment has been found about the lack of handling of the other main system concepts, such as design, increment and feedback/result. See Table 2.4. From the table, IE emerges as the IT prioritization method providing best coverage across the system concepts.

2.8 Prioritization factors

Within the prioritization process, there are many prioritization factors (also sometimes called “criteria” (Wohlin and Aurum, 2005b; Botta and Bahill, 2007; Barney et al., 2008) or “aspects” (Lehtola, 2006; Berander, 2007)) that can be considered when assessing stakeholder value. See Table 2.5, *Prioritization factors by stakeholder viewpoint and system concept*.

A list of over 50 prioritization factors is identified from the literature; the main sources include (Carlshamre et al., 2001; Moisiadis, 2002; Greer and Ruhe, 2004; Firesmith, 2004; Berander and Andrews, 2005; Lehtola and Kauppinen, 2006; Berander, 2007; Botta and Bahill, 2007; Barney et al., 2008). These factors are mapped into Table 2.5 with related factors being grouped together. On analyzing the resulting groups, the main prioritization factor groupings presented on the left-hand side of the table were identified: opinion, strategy, time, legal, financial benefit, cost, fit, external dependency and risk.

Additionally, the identified prioritization factors are subdivided by stakeholder viewpoint and then by system concept. For simplicity, just three stakeholder viewpoints are given representing the mandatory viewpoints in any systems development prioritization process: strategy management, systems development and operations management. (Clearly there are many more stakeholder roles than these in a system.) The sub-division by system concept uses the four system concepts (objective, requirement, design and increment). The mapping between the stakeholder viewpoints and the system concepts is as follows: strategy management

Table 2.5: Prioritization factors by stakeholder viewpoint and software engineering concept

STAKEHOLDERS	**Strategic management**	**Systems development**		**Operations management/ customers**
CONCEPTS	**Organizational objectives (objective)**	**Systems requirements (requirement)**	**Design solutions (design)**	**Delivery plans (increment/delivery/ deliverable)**
PRIORITIZATION FACTORS				
OPINION	Vision/intuition/gut feeling/preference/bias	Preferences/ bias/importance	Intuition/preferences/ bias	Preferences/bias
STRATEGY	Strategic alignment/business objectives/product strategy		Long-term strategy for systems architecture	
	Competition	Quality		
	Customer demand	Originator of requirement		End user value
	New business potential			
TIME	Urgency/time to market/ lead time		Time schedule/ time constraints	
	Long term versus short term		Long term versus short term	
LEGAL	Legal mandate/regulations	Legal mandate/ regulations		
	Contracts in place			
FINANCIAL BENEFIT	Market value/price			
	Financial benefits			
	Financial penalties			
	Benefit/cost ratio			
	Cost of not implementing			
COST	Development costs/ implementation costs/ support costs		Development costs/ support costs	Implementation costs/ support costs
	Operational costs			Operational costs
FIT	Fit with operational context: - Business processes - Skills/training - Delivery timing		Staff competence Balanced workload	Fit with operational context: - Business processes - Skills/ training - Delivery timing
			Resource availability/ effort constraints	Resource availability/ effort constraints
	Fit with other products		Change impact/ base code dependencies	Change impact
			Logical implementation order	
			Reuse potential	
EXTERNAL DEPENDENCY	Intermediary channels		External dependencies	
RISK	Business risk Sales barriers		Technical risk in: - current system - proposed system - implementation process	
			Difficulty of implementation/ complexity	Difficulty of implementation/ complexity

has responsibility for the objectives, systems development is primarily responsible for the requirements and designs, and operations management has responsibility for accepting the planned and delivered increments.

Note that the list of prioritization factor groupings in Table 2.5 is not considered complete (for example, environmental is missing): this table only reflects the main prioritization factors found in the literature. The following observations can be made:

- Regarding the three stakeholder viewpoints selected, these are similar to those selected by Lehtola (2006): business, implementation and customer, and by Barney et al. (2008): management, system, business (for this read, customer) and additional criteria (a 'catch-all' category).
- In the mappings between the stakeholder viewpoints and the prioritization factors, the table provides support that different stakeholder viewpoints exist as different stakeholders are shown to relate to different prioritization factors. This means any prioritization process or prioritization method must cater for multiple stakeholder viewpoints.
- The prioritization factors span all the four system concepts. This argues for a prioritization process that offers support for all these concepts, that is, that a holistic approach across the system concepts is required to capture the 'complete picture'.
- A general set of prioritization factors groupings (opinion, strategy, etc.) with their associated factors emerges from the table that could be proposed as an initial set for an integrated prioritization process.
- A proposition can be made that the prioritization factor groupings (for example, strategy, legal, cost and risk) are the dimensions for assessing stakeholder value.

Given the importance of the prioritization factors within the IT prioritization process, further explanation of these factors is appropriate. A stakeholder when determining the priority of a system change should consider which factors apply and then assess the relevant factors further and attempt to put in place some evidence to justify their assessment.

The prioritization factors discussed in the literature (Carlshamre et al., 2001; Moisiadis, 2002; Greer and Ruhe, 2004; Firesmith, 2004; Berander and Andrews, 2005; Lehtola and Kauppinen, 2006; Berander, 2007; Botta and Bahill, 2007; Barney et al., 2008) include:

- Stakeholder opinion: This factor is concerned with personal opinions, preferences, biases and intuition.
- Strategy: Strategic alignment, product strategy, architectural strategy and competitive strategy can all be taken into account.
- Time: Urgency, time to market, short-term versus long-term timescales.
- Legal obligations: There can be mandatory legal obligations that have to be met.
- Financial value: Potential financial benefits (or on the negative side, financial penalties), cost/benefit ratios and the cost of not implementing can all be assessed.
- Cost: There are many different types of cost to consider: for example, development costs, installation costs and operational costs. These should be associated with specific designs.
- Fit: This is the fit with existing components, such as existing products, current staff skill levels, current business processes, existing organizational culture and current workload.
- External dependency: There can be dependencies with external organizations. These may be customers and/or suppliers.
- Risk: Risk can take many forms: for example, business risk, sales barriers, volatility of requirements, the degree of change involved and technical risk. It depends on individuals and/or the organization's attitudes towards risk as to how much risk can be considered acceptable. Often risk is seen as a negative concept, though there is an emerging acceptance that risk can be seen as positive – and that 'No risk, no gain' is often the case when pursuing high gains.

Not all factors are relevant to all types of stakeholder: the different stakeholders will have different viewpoints on each of these factors and they will have expertise in different areas. For example, a product's price impacts the customers and the systems developers in different ways: it is a customer's cost, but it also impacts the systems developers' profit margin or cash flow. To give another example, assessing the contribution towards competitive strategy will most likely be the responsibility of Marketing, while the architectural strategy will be the responsibility of the systems architect.

Tentatively, the prioritization factors can be seen as mapping to the different types of stakeholder value. Certainly 'strategic value' and 'financial value' exist. Further research work is needed to determine if such a mapping proves useful when considering stakeholder value.

See Table 2.6, which shows how the selected prioritization methods specifically cater for the prioritization factors

Table 2.6: Mapping of prioritization methods to the prioritization factors

PRIORITIZATION FACTORS	PRIORITIZATION METHODS						
	MoSCoW	Requirements Triage	AHP	QFD	IE	Cost-Value Approach	Planning Game
Stakeholder opinion/preference	Y	Y	Y	Y	Y	Y	Y
Strategy	N	N	N	(Y)	(Y)	N	N
Time	N	Y	N	N	Y	N	Y
Legal obligations	N	N	N	N	N	N	N
Financial value	N	N	N	N	(Y)	(Y)	N
Cost	N	Y	(Y)	(Y)	Y	Y	Y
Fit	N	N	N	N	N	N	N
External dependency	N	N	N	N	N	N	N
Risk	N	N	N	N	(Y)	N	(Y)

Key: Y = Yes, N = No, (Y) = Some coverage

To explain the background to some of the entries in Table 2.6:

- MoSCoW captures the stakeholder opinion about the individual requirements at a high, aggregated level with no indication of what factors should be considered. The method has no means of specifically addressing any of the other factors
- Requirements Triage captures the importance of each requirement and estimates the effort and timescales required. The issues are seen as a lack

of definition of what ‘importance’ consists of and, that effort is being estimated against a specific requirement rather than against a specific design.

- In AHP, both requirements and designs are considered. The weighting of the requirements is carried out by pair-wise comparison and it is not clear what factors are being taken into account during the comparison. Subsequently the designs are pair-wise compared for their impact on the individual requirements
- QFD’s HoQ evaluates the impact of designs on the requirements. Once again it is not specified what factors have to be considered in categorizing the impact. There is specific consideration of comparison to other competitors’ products, customer and sales preferences. Also cost and ease of implementation can be addressed, though in practice this is not usually done
- IE also evaluates the impact of designs on the requirements. Given the requirements are expressed using metrics any evaluation is specifically based on asking about the resulting change – by how much does the design impact on altering the level of the metric from the baseline level towards the target level. Development costs (and maybe other costs) are captured and performance to cost ratios are calculated. These ratios give some idea of value (and relative value among the designs) in terms of meeting the requirements. The requirements are expressed as the required target level by a specific date – so time is taken into account. If increments are being determined, more detailed consideration of elapse time is involved. Risk is handled at the level of whether the designs cover the pre-determined safety-margin. So not all risk is explicitly considered
- The Cost-Value Approach considers only the requirements. It uses AHP and so suffers from the same problems as AHP. However, the Cost-Value Approach method does explicitly consider cost and evaluates value against cost by plotting a graph
- The Planning Game takes much the same approach as Requirements Triage – the ‘value’ that is assessed is essentially the same as the

‘importance’ assessed in Requirements Triage – once again there is no specific definition of what should be taken into account. The Planning Game labels the ability to estimate effort as ‘risk’, but this is only one aspect of risk and once again, it is the effort to deliver a requirement rather than a specific design that is being assessed. It does take into account the increment loading – so timescales are being considered.

2.9 Means used to express value within IT prioritization methods

All the existing IT prioritization methods demonstrate problems with the expression of value. Specifically value is:

- Often implicit
- Rarely considered as multi-dimensional
- Frequently limited or aggregated to a single stakeholder viewpoint
- Restricted in its linkage to system concepts, such as requirements, designs, and increment deliverables, as well as lacking in due consideration of system context (time, place and event)
- Other problems

The main flaw in the existing prioritization methods is that stakeholder value is often expressed indirectly. For example, it is often treated within the umbrella term of ‘importance’ (Karlsson et al., 1998; Berander and Andrews, 2005) or by means of requirements being classified using such terms as ‘mandatory’, ‘desirable’ or ‘inessential’ (Brackett, 1990 quoted in Karlsson, 1996). The majority of these terms are expressing stakeholder value in an ambiguous way. The problem is compounded when there is no means of capturing multiple stakeholder values (Karlsson et al., 1998; Moisiadis, 2002) or inappropriate aggregation and/or weighting of stakeholders or stakeholder values is carried out. Such methods fail to cater adequately for stakeholders having different system viewpoints and/or system usage.

Value is often Implicit: Within prioritization methods, value can be captured using groupings, rankings, weightings or metrics. To give some examples, MoSCoW

(Stapleton, 2003) captures value under the groupings: ‘Must Have’, ‘Should Have’, ‘Could Have’ and ‘Would Like to Have, but Won’t Have This Time’. The Planning Game (Beck, 2000) also uses groupings (‘essential’, ‘less essential’ and ‘nice to have’), but also ranks its user stories in order of preference (‘1’, ‘2’, ‘3’, etc. where there is a stated descending or ascending order of preference). AHP (Saaty, 1990) calculates normalized weightings (Expressed very simply, this results in say, values such as ‘0.15’ for a requirement x and ‘0.30’ for requirement y. In which case, requirement y can then be said to be twice as preferred as requirement x). Only IE (Gilb, 2005a) is found to make use of ‘real world’ metrics (QFD can demand the capture real world metrics, but it fails to use them in its calculations). The benefit being that real world metrics provide a more explicit basis for understanding value, and also support arithmetical calculations, such as return on investment (ROI). To give an example, for a performance requirement of ‘usability’, metrics using ‘real world’ scales of measure such as ‘average number of errors made per 100 transactions’ or ‘average time taken in minutes for a new customer to place an order’ would be used rather than saying that usability was ‘very important’ (grouping) or ranked with a priority of ‘1’ (ranking).

The benefit of ‘real world’ metrics is that numeric data instances of a ‘metric’ can be expressed for defined levels, such as the target levels for goals and budgets, and the constraint levels for system survival and failure. Such ‘real world’ data has the advantages of not only being unambiguous and testable, it also aids communication amongst stakeholders. Further, it enables more explicit determination of value: if you know the current and targeted levels, you have a basis for discussion (if we reduced the number of errors occurring in a week by 100, how much staff time would that save on average? What if you reduced errors occurring in a week by 200?) If you can estimate the cost of developing the proposed solution(s), then you can calculate potential ROI(s). This contrasts with trying to assess value against ‘very important’ or priority ‘1’.

A further point is that the rationales behind the stakeholders’ choice of value tend not to be captured. So not only is there *implicit* expression of value, which is difficult to correlate to real world value, there is also no explicit information about what the stakeholder was taking into account in expressing value. This especially hinders reprioritizing or auditing value.

Value is Rarely Considered as Multi-Dimensional: Value for a requirement or a design is typically expressed as a single selection or estimate. However, there are different types of value, for example strategic value, financial value, legal value and environmental value. As discussed earlier, the literature (Carlshamre et al., 2001; Moisiadis, 2002; Greer and Ruhe, 2004; Firesmith, 2004; Berander and Andrews, 2005; Lehtola and Kauppinen, 2006; Berander, 2007; Botta and Bahill, 2007; Barney et al., 2008) identifies over 50 prioritization factors, which can all be considered to contribute to the various different types (dimensions) of value. These identified prioritization factors can be grouped by concept area: opinion, strategy, time, legal, financial, cost, fit, external dependency and risk (see Table 2.5 for further detail). Several of these concept areas are also identified by Berander (2007) and termed by him as "aspects". These groups/concept areas can be taken to express – as a first-cut – the different types/dimensions of value.

Identifying such types/dimensions of value is a focus for research. While some of the prioritization factors can be reduced to financial terms, it is unlikely all can be. Further, aggregating the financial value of the different types of value would lose information necessary for a prioritization process. That is, while you can translate environmental impact to a financial value, there is merit in considering the environmental value in its own right as well.

Value is Frequently Limited or Aggregated to a Single Stakeholder Viewpoint: Within any system there are numerous stakeholder groups, who have different perspectives on what types of value matter, where value resides, and also the amount of value. As discussed earlier, the stakeholders' areas of expertise and interests in a system differ. Handling of multiple stakeholder viewpoints seems universally poor within existing prioritization methods. It is not clear if any method presents an overview of different stakeholder viewpoints. RPT (Moisiadis, 2002) is perhaps the closest.

In addition, some suspect practices such as weighting stakeholders according to job title/role or averaging across stakeholder's preferences can be found. Such practices distort stakeholder value and/or hide where the value resides.

Value is Often Restricted in its Linkage to System Concepts: Many prioritization methods focus on value associated with one or a small set of system concepts (for example, only using requirements data or only design data). As identified earlier, in the literature the research on prioritization methods tends also to be segregated into subject areas, for example considering requirements prioritization or architecture prioritization or release planning. While such research specialization is not necessarily a problem, the key issue is the lack of an integrated approach. This is for two reasons: first, IT projects in industry need some form of prioritization process that works across the system development lifecycle (they are unlikely to use a diverse set of prioritization methods), and secondly, perhaps more importantly, the prioritization process itself can be seen as requiring a wide scope across all system concepts if value is to be adequately addressed. To give a simple, admittedly non-IT example: buying a car. The single stakeholder, requirements-only approach would be to simply go out and buy the car of your dreams (and admittedly if the car was looked after, this could be a long-term investment!) However, a more common reality is that a wider system, multi-stakeholder approach is taken. You consider the stakeholders (for example, other family members, the servicing garage, the government, the environment, and other passengers). You then consider the quality requirements (for example, less petrol consumption, improved reliability, improved safety, improved security, better in-car sound, minimum amount of storage space, cheaper running costs and the more modern appearance of the car). Next you consider the potential cars (designs) that you could buy. Purchase cost is often an immediate consideration and can rule out some designs. Then you investigate each design's impacts on your set of requirements. Further, in some cases, the availability (timescales) for the delivery of the car becomes a consideration: maybe there is a waiting list or your financial planning requires a certain delay. The point being made using this example is that there is a need to balance stakeholder requirements, potential designs and delivery options. When considering these concepts you are thinking about the resulting value across the different stakeholders. In turn, the question becomes why it is acceptable in many IT prioritization methods to just consider a very limited set of system concepts, when in reality there is actually a need to take into account many more factors.

2.10 Conclusions

2.10.1 Summary of the main findings from the literature review

Overall the picture that emerges from the literature in the area of IT prioritization is of an immature, yet developing research area. Too many papers focus mainly on the procedures/techniques of the prioritization methods. As yet, there is little discussion of the underlying theory in terms of fundamental system concepts (such as value, quality, constraints, and risk) and this is recognized in the literature (Firesmith, 2004). A specific observation is that 'value' is a concept requiring better definition and integration into decision-making methods. Again, the lack of underlying theory is identified in the literature (Sullivan, 2007).

A broader scope is needed with regards the prioritization process: it should be more holistic in its approach considering the entire range of system concepts (objectives, requirements, design and implementation plans, and also value as well as others) and it should consider the whole span of the systems engineering lifecycle (from strategy to operational use over time). Certainly the idea that requirements prioritization should be carried out independently of considering a wider set of system concepts needs challenging. It emerges from Table 2.5 that the prioritization factors can be allocated across system concepts and across stakeholders. This provides limited theoretical evidence of the need to support multiple stakeholder viewpoints and to consider value in relation to a wider set of system concepts.

The literature review specifically identifies problems with the structure and content of prioritization data. See Table 2.7, which provides an overview of the issues discussed in the literature review. The distinct areas are interlinked: a prioritization process is carried out by an IT prioritization method that utilizes one, maybe two, prioritization techniques. Further, prioritization is carried out by, or on behalf of, the stakeholders. The key issues considered relevant to expressing prioritization data include:

- Explicit stakeholder value: There is a need to state stakeholder value explicitly. Too often stakeholder priority is used to express implicit stakeholder value
- Multiple stakeholder viewpoints: Multiple stakeholder viewpoints need supporting. The different stakeholders and their associated stakeholder

value need capturing. Account must be taken that stakeholders have differing areas of expertise and knowledge

- Coverage of all system concepts: A focus only on prioritizing requirements and therefore a lack of consideration of other system concepts, such as design, increment and system context
- Handling abstraction: System concepts need to be able to be handled at different levels of abstraction
- Catering for interdependencies: The ability to handle interdependencies amongst the requirements and also amongst the designs
- Supporting reprioritization: The prioritization data must be captured in a way that it can be reused without needing further input from the stakeholders (unless something significant has changed in the system and/or its environment that means additional data is necessary)
- Coping with scaling-up: There needs to be the ability to scale up to cope with large systems with large numbers of system concepts. Often prioritization is only carried out at a high-level of data abstraction
- Enabling communication of information: Often prioritization data is created in a form that conveys little meaning to the stakeholders and so fails to support negotiation and decision-making.

From the findings to date, it can be seen that the selected existing prioritization methods do not provide much support to address the prioritization process issues. Of the selected methods, IE emerges as an IT prioritization method worth further consideration. It not only is the only prioritization method found to actively use metrics, but it also offers better coverage of the system concepts. For these reasons, IE was chosen as the basis for further research work. However, it is noted that IE only offers limited coverage of multiple stakeholder viewpoints and mainly expresses stakeholder value implicitly.

Table 2.7: Overview of prioritization issues

Prioritization Process (Section 2.7) - Stakeholder value - Prioritization factors - Coverage of the system concepts: - objectives - requirements - designs - deliverables/increments - stakeholder value	Stakeholders (Section 2.5) - Multiple viewpoints - Different stakeholder value - Different knowledge and expertise - Conflicting views - Communication/negotiation among the stakeholders
Prioritization Technique (Sections 2.3 & 2.4) - Four techniques: - grouping - ranking - weighting - metrics - Expression of priority - Lack of explanation of priority assignments options - Failure to capture rationale - Implicit stakeholder value - Catering for multiple stakeholder viewpoints - Reprioritization - Scaling-up	Prioritization Method - Numerous current methods (Section 2.2) - Evidence-based - Support for multiple stakeholder viewpoints - Support for all required system concepts - Handling of data structures - Usability (Section 2.5) - Effective and Efficient - Manual versus Automated - Ease of learning and use - Acceptability of use - Transparency of processing - Transparency of the results - Avoidance of data errors (missing data, inaccuracies, inconsistencies or miscalculations)
Data Structure (Section 2.6) - Abstraction - Hierarchies - Matrix relationships - Interdependencies	

2.10.2 Revision of the research question and propositions

At the end of the Introduction (Chapter 1), the research question was given as follows:

> **"How can better support be provided for priority decision-making regarding IT investment?"**

As a result of the literature review, IE has emerged as the most promising IT prioritization method and so the research question is now modified to reflect this. The research question becomes as follows:

> **"Within Gilb's Planguage method, how can the current provision to support IT priority decision-making (namely, Impact Estimation), be improved?"**

The literature review and the subsequent critical analysis have identified that the reported prioritization methods handle priority and value in an inadequate manner. Therefore, Proposition 1 aims to address these aspects in the field.

Proposition 1: Current IT projects fail to capture value and priority information adequately in their system specifications.

One of the key aspects of the choice of IE is its use of metrics to specify the quality requirements. So Proposition 2 is declared to specifically investigate whether metrics are indeed helpful in specifying stakeholder value and further, if they are of use in supporting determination of priorities.

Proposition 2: The use of metrics is beneficial to understanding the quality requirements and for supporting IT priority decision-making.

The importance of capturing the different stakeholder viewpoints and their associated stakeholder value is also established in theory by the literature review. So Proposition 3 aims to place a focus on the stakeholder experience and, specifically the specification of stakeholder value.

In addition, stakeholders incur costs. IE derives quality-to-cost ratios as one means of establishing priority: the design solutions with the higher quality-to-cost ratios are seen as having greater priority due to being more cost-effective. Therefore, considering stakeholder costs is of interest.

Proposition 3: Identifying stakeholders more explicitly and considering the impacts on stakeholders of the requirements in terms of value and cost is useful for IT priority decision-making.

Proposition 4 aims to determine whether the proposed method is efficient and effective. The amount of effort that IT prioritization methods require is seen as a key factor in their uptake.

Proposition 4: The additional effort spent structuring the data using the proposed method helps identify and understand the system data and supports IT priority decision-making.

Finally, Proposition 5 addresses the required outcome to see if the impact of the proposed IT prioritization method does indeed help stakeholders obtain a better understanding of the proposed system changes and results.

Proposition 5: If stakeholders are provided with a functioning IT prioritization method, then a better understanding of the planned project outcomes is the result.

The above modified research question and propositions now inform and underpin the proposed research. The task of Chapter 4, the Research Design, is to show how they will be tested. However first, the next chapter explains the basic workings of IE, and describes the proposed, extended IE method, which is to be evaluated.

Table 2.8: Overview of IT prioritization methods

Prioritization method	**Year of origin & people**	**Brief description / notes**	**Scale Type**	SC	**Reference(s)**
REQUIREMENTS PRIORITIZATION					
Priority Groups/ Grouping/ Numerical Assignment/ Numeral Assignment	Unknown	Priority groups are typically three categories of importance using a variety of category names	Ordinal	R	von Mayrhauser 1990 quoted in (Andrews et al., 2004); Lehtola and Kauppinen (2006); Mead (2006)
MoSCoW Method/ MoSCoW Analysis/ MoSCoW List	Unknown. Dai Clegg (He gave IPR to DSDM which originated in 1994)	Priority group method with four levels of importance. Part of Dynamic Systems Development Method (DSDM).	Ordinal Priority Groups	R	Clegg and Barker (1994); Stapleton (2003)
Quality Grid	Before 2002 Lauesen and Christiansen	Categorises the different "quality factors" (attributes) into 5 groups: critical, important, as usual [?], unimportant and ignore.	Ordinal Priority Groups	R	Lauesen (2002)
Ranking	Unknown	Each item given individual rank. Can be carried out using a binary search or a bubble sort	Ordinal. Not Interval	R	Karlsson et al. (1998); Ahl (2005); Berander and Andrews (2005); Mead (2006)
Case-based Ranking (CBRanking)	2004 Avesani, Bazzanella. Perini and Susi.	Based on AHP modified to machine learn and apply preferences.	Ratio	R	Avesani et al. (2005)
Top Ten	Unknown	Selection of a group of ten items with the	Ordinal	R	Lauesen (2002); Berander and Andrews (2005)

		highest priority			
$100 Dollar Test/ 100-Point Method/ Cumulative Voting (CV)	Unknown	Assign the $100 across the requirements according to their perceived value	Ratio	R	Leffingwell and Widrig (1999); Berander and Andrews (2005); Mead (2006)
HCV Hierarchical Cumulative Voting	2005 Berander and Jönsson	Like CV, but create hierarchy and vote within hierarchical groups	Ratio	R	Berander (2007)
Requirement Prioritization Tool (RPT)	2002 Moisiadis	Users specify their preferences and rationale using an application tool (RPT). They are asked to specify a range over which their preference lies.	Ordinal – seems to be ranking	O R	Moisiadis (2000; 2001; 2002); Mead (2006)
REQUIREMENTS (AND EFFORT) PRIORITIZATION					
Cost Value Approach / Cost-Value Method / Contribution-based Method	1997 Karlsson and Ryan	AHP is used to determine value and then value is plotted against implementation cost	Ratio as for AHP	R	Karlsson (1996); Karlsson and Ryan (1997)
Wiegers' Method	2003 Wiegers	Priority is calculated as %Value divided by the sum of weighted %Cost and weighted %Risk	Ratio	R	Lehtola and Kauppinen (2006); Moisiadis (2002); Wiegers (1999)
Attribute Driven Design ADD	2001 Bachmann and Bass - SEI	Quality attributes are rated for their business value and technical effort separately using High, Medium and Low, e.g. HL. Utility trees of the quality attribute	Ordinal	R	Ozkaya et al. (2008)

		hierarchy are built.			
REQUIREMENTS AND DESIGN PRIORITIZATION					
Analytic Hierarchy Process (AHP)	1972 Saaty	Pair-wise comparison of requirements at same level of abstraction within hierarchical groups	Ratio	R D	Khan (2006); Berander and Andrews (2005); Moisiadis (2002); Karlsson et al. (1998); Karlsson and Ryan (1997); Saaty (1990; 1997)
MAGIQ	2006 McCaffrey	Similar to AHP but faster assignments	Ratio	R D	McCaffrey (2009)
Connection Matrix Or value-orientated process improvement approach	2007 Omasreiter	Matrix of goals against actions (process improvements) using simple numeric allocations. Goals are evaluated on 0-4 scale and impact of action on goal on 0-1 scale. No further calculation.	Ordinal given the scales that are used	R D	Omasreiter (2007)
Quality Function Deployment (QFD)	Late 1960s in Japan. Mizuno and Akao	Matrix of requirements against design components.	Mainly Ordinal (Ratio among requirements)	R D	Akao (1990; 1997); Martins and Aspinwall (2001); Burke et al. (2002); Chan and Wu (2002)
Impact Estimation	From early 1980s Gilb	Matrix of quantified impacts of designs on quantified requirements.	Absolute - Metrics	O/ R D/ I	Gilb (1988; 2005a)
ARCHITECTURE (DESIGN) PRIORITIZATION					
Cost Benefit Analysis Method (CBAM)	2001 SEI – Kazman, Asundi & Klein Builds on	Evaluates benefit and cost of architectural strategies. Weight the quality attributes to total	Ratio	R D	Kazman et al. (2001)

	ATAM	100. Assess designs towards quality attributes on -1 to +1 scale. Multiply contribution by weight and sum to get total for design. Plot graph of benefit against cost.			
RELEASE PLANNING PRIORITIZATION					
Planning Game (PG)	1999 Beck. Part of Extreme Programming (XP)	Write user stories. Assess by value of essential/ significant/nice. Assess by risk (effort) of precise/well/unabl e to. Set velocity (Effort available in increment). Select user stories.	Ordinal	R	Beck (2000); Mead (2006)
Requirements Triage	2003 Davis	Triage requirements into next/might/no for next release. Vote by importance using 100-dollar test or next release voting on the 'mights'. Assess also for effort using points or development hours.	Ordinal	R	Davis (2003); Mead (2006)
EVOLVE	2002 Greer and Ruhe	Stakeholders weighted using AHP and normalized to 1. Use scale of 1-5 to assess each requirement for	Ordinal / Ratio	R	Greer and Ruhe (2004)

		urgency and value. Scale of 1-10 to assess implementation effort. Limit set for effort per increment.			
EVOLVE*	2005 Saliu and Ruhe	Extend EVOLVE to cater for risk. Scale used is 1-9. Three assessments urgency, value and effort as EVOLVE. Tool is Release Planner.	Ordinal	R	Saliu and Ruhe (2005)
FINANCIAL PRIORITIZATION					
Incremental Funding Method (IFM)	Prior to 2004 Denne and Cleland-Huang	Plot relative value against relative cost for MMFs (Minimum Marketable Features) found by functional decomposition. Architectural elements (AEs) attached to MMFs. Financial value assessed. Net Present Value (NPV) can be used.	Ratio	R D	Denne and Cleland-Huang (2004)
NEGOTIATION PRIORITIZATION					
Theory-W or Win Win	1989 Boehm and Ross		n/a		Mead (2006); Park et al. (1999); Boehm and Ross (1989)
Quantitative Win Win	2003 Ruhe, Eberlein and Pfahl	Uses AHP on requirements split into classes. Classes are weighted and then requirements within classes. Priority obtained	Ratio	R	Ruhe et al. (2003)

		by multiplying the two. Mandatory requirements assigned value of 1. Also consider effort, time and quality involved for each requirement. Assign to increment using value and tradeoff using resource and quality constraints			
WinCBAM	2001 SEI and Texas A&M In, Kazman and Olson	Extending the Cost Benefit Analysis Method (CBAM) with Win Win	Ratio	R D	Kazman et al. (2005)
Distributed Collaborative Prioritization Tool (DCPT)	1999 Park, Port, Boehm and In	Stakeholders vote on a scale of 1 to 10 on importance and difficulty. Different stakeholder votes are combined into bins: Average, ROI, Net Value and Risk.	Ordinal	R	Park et al. (1999)

Key:
SC = System concepts involved
O = Objectives prioritized
R = Requirements prioritized
D = Designs prioritized
I = Implementation deliverable for operational use prioritized
n/a = not applicable

Chapter 3

Proposed IT Prioritization Method

3.1 Introduction

As a result of the literature review, Impact Estimation (IE) emerged as the most promising candidate IT prioritization method for further consideration. This chapter describes IE in outline, assesses its shortfalls and evaluates its potential. As a result, an extension to the IE method is proposed creating the Value Impact Estimation (VIE) method. In addition, an IT prioritization process incorporating the proposed extension is discussed and specified.

This chapter discusses the relevant aspects of IE. Several features of IE, such as how it deals with uncertainty, credibility, safety deviations and safety margins are considered beyond the scope of this research and are therefore not discussed in detail[5].

3.2 Findings from the literature review

From the literature review, the majority of the current IT prioritization methods are using ordinal and/or ratio scale data. With regard to arithmetic calculations, ordinal

[5] Uncertainty, credibility and the required safety margins most likely do have an impact on priority decisions: a more specific, credible design impact would have more weight than a more ambiguous impact. However, the need for considerable additional data would be introduced by catering for these additional factors and at this initial stage of the research the resulting additional complexity is not considered as essential or helpful.

scale data offers little support as it only allows equivalence or greater than/smaller than relationships to be declared (Kaposi and Myers, 1994). For example, MoSCoW has four priority groups and requirements are allocated into one of them. All that can be stated is that the requirements in the Must Have priority group are of greater priority than those in the Should Have priority group, and so on. The only resulting prioritization data is typically the priority group name. Ratio scale data is much stronger and supports sufficient arithmetic calculations. The issue is that the prioritization data is simply a ratio and its relationship to the real world is usually not captured in a meaningful way. For example, determining that the usability requirements are of 20% importance to the stakeholder as opposed to the security requirements being of 30% importance fails to capture sufficient information about the current levels and the required levels and their perceived stakeholder value. It is not inherent in the ratio-scale prioritization methods that the real-world data is collected, and even if it was mandated, the methods do not utilize this data in their prioritization processes.

Perhaps the most telling issue is the problem of the prioritization data having little meaning to the stakeholders. Absolute scale data (Kaposi and Myers, 1994) given it relates to real world measurement is easier for stakeholders to understand, engage with, and reason about. This points to methods that use absolute scale data being preferred, and the only candidate prioritization method found to date is IE, which uses metrics.

An additional reason for the selection of IE is its integrated coverage of the system concepts: the method's ability to cater for prioritization planning with regard to designs, increments and actual results, as well as quality requirements. A prioritization process should cater for a holistic approach. Botta and Bahill (2007) in their discussion of the prioritization process recognise the need for prioritizing other system concepts as well as requirements, however their discussion considers prioritizing the different types of system concept separately.

3.3 Use of metrics within IE

As explained above, one of the two main reasons for considering IE as a candidate method for IT prioritization is its use of metrics. When reasoning about priorities, it is helpful to the stakeholders to know the current levels and the proposed target levels for the quality requirements. They can assess the stated quality levels that are within their area of expertise. In turn, these quality levels also form the basis for assessing stakeholder value. The stakeholders, who are asking for the quality requirements, ought to be able to estimate the stakeholder value associated with their requirements especially when they are stating the current levels and specifying the target levels.

It is the quantified quality levels that enable the estimation of stakeholder value. For example, trying to estimate the stakeholder value of the system requirement, "easy to use rules administration" is fairly difficult. However, once a metric is put in place interpreting this requirement as improving the time taken to update the rules, and there is a current (Past) level and at least one target (Goal) level, then with additional information about staff costs and the number of rule updates for a specific time period, it is possible to estimate the associated stakeholder value in terms of the value associated with the reduced staff effort. An additional consideration might be the impact on the business of having the updated rules in place earlier. Note it is likely in some cases that more than one metric could be needed to fully capture a system requirement and its associated stakeholder value. An additional consideration is that identifying all the stakeholder value is not always necessary, it depends on the context. For example, once it has been shown that the stakeholder value of a proposed design exceeds the development and operational costs, and provides greater stakeholder value than the alternative designs, then providing any organizational policy standards about the required levels of stakeholder value are also met (such as, the required level of ROI), sufficient stakeholder value is identified to proceed.

Stakeholder value will vary across the stakeholders or in other words different stakeholders will receive different stakeholder value (For example, providing a system that records students' attendance at lectures and seminars frees up the lecturers and tutors from having to maintain registers, and means that central administration have much earlier access to the attendance data, which in turn means they can address student attendance issues). For any given quality requirement, only a specific set of stakeholders will be impacted and this could be both in a positive or negative way (net gain or loss). Moreover, the extent of the impact will vary across these stakeholders, as it will depend on their level of use of the system functionality involved. For example, one stakeholder might consist of 100 staff using the business rules throughout their working day, whereas another stakeholder might only consist of ten staff using the business rules intermittently. Obviously the stakeholder value is likely to be greater for the former stakeholder. Provision of explicit stakeholder value by stakeholder is therefore ideally needed for prioritization decision-making, and that means supporting multiple stakeholder viewpoints.

3.4 Use of system engineering concepts within an IT prioritization process

Adopting a holistic view means any IT prioritization process must include consideration of a wide range of prioritization data, which includes the fundamental system concepts: objective, requirement, design and increment. See Figure 3.1, *Increment delivery cycle based on Deming's Plan-Do-Study-Act (PDSA) Cycle* (Deming, 1993). This figure was developed by the author during this research. It shows an increment's delivery cycle with iteration around these concepts as software development progresses. For Waterfall methods (Royce, 1970) there is one iteration, while for evolutionary and agile methods (Abrahamsson et al., 2002; Larman, 2004), there are many.

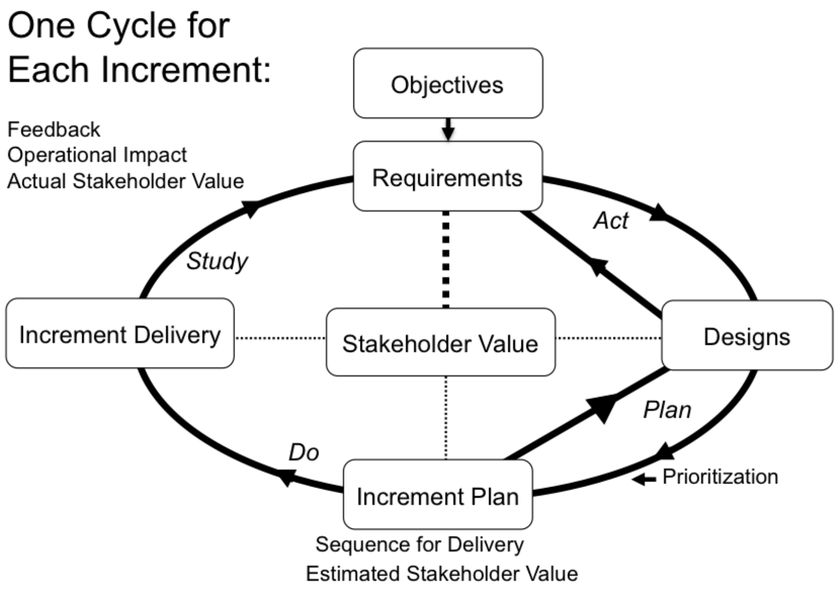


Figure 3.1: Increment delivery cycle based on Deming's Plan-Do-Study-Act (PDSA) Cycle

The concept of stakeholder value is positioned in the centre. Given prioritization involves decisions about IT investment (Barney et al., 2008) and the need to deliver benefits (Botta and Bahill, 2007), there is a requirement to capture information about stakeholder value in order to ensure that implementation plans offering high value are selected. Value is determined by considering more than simply the requirements. Different designs satisfy the requirements to varying extents and have different costs (Gilb, 2006). Adding considerations of operational use provide a more complete view of value. In fact, prioritization ideally should take place once the potential designs for the increment are known, and crucially it must consider what value can be delivered in the next delivery. Prioritization results in an increment plan (the selected design(s) within specific context(s)) that is then followed by increment delivery. Feedback from the field then determines whether the estimated value is achieved.

There are some specific needs to support evolutionary and agile methods. Such methods require the on-going capture of data, including changing requirements and feedback of the actual results from each incremental delivery. The prioritization process has to cater for this changing data while also accommodating reuse of data. Ideally the prioritization data can be reused in subsequent prioritizations (future increments) without needing further inputs from stakeholders (unless something significant has changed in the system and/or its environment that they need to provide additional data on). Reprioritization has to occur within each increment to help stakeholders to determine what to implement next: prioritization takes on a central, on-going role throughout development. In contrast under Waterfall methods, prioritization ideally only has to be carried out once, early on in the systems development process, and involves deciding which requirements are prioritized to be in the system and which are not. However, there is still a concern that any changes in requirements that occur are captured: Moisiadis (2002) considers that the prioritization process should support any "evolution" of the requirements being captured in the specifications, which acknowledges that new information might come to light during prioritization. There is also concern about the volatility of requirements (Moisiadis, 2002). So reprioritization is not seen solely as the concern of evolutionary and agile system development.

3.5 How IE uses the main system concepts

As discussed above, one of the two identified main reasons for IE being a candidate was its use of the system concepts. This section expands on this by investigating in more detail how IE uses the main system concepts: the Planguage system model is first explained and then it is mapped to the IE table.

3.5.1 The Planguage system model

The Planguage system model, see Figure 3.2[6], shows how the main system concepts are seen to relate to each other. Planguage integrates the quality, resource, condition and design attributes with the functionality. The model can relate to any part of a system's functionality: at the highest-level the function could be that of the entire system or alternatively, it could be any lower-level sub-function(s) of the system.

Each function is considered to exhibit a set of quality attributes: these are shown as 'output' by the function. The quality attributes are sub-divided into the -ilities, resource saving and workload capacity. The resource attributes are shown as being 'input' to or consumed by the function and are typically decomposed into the

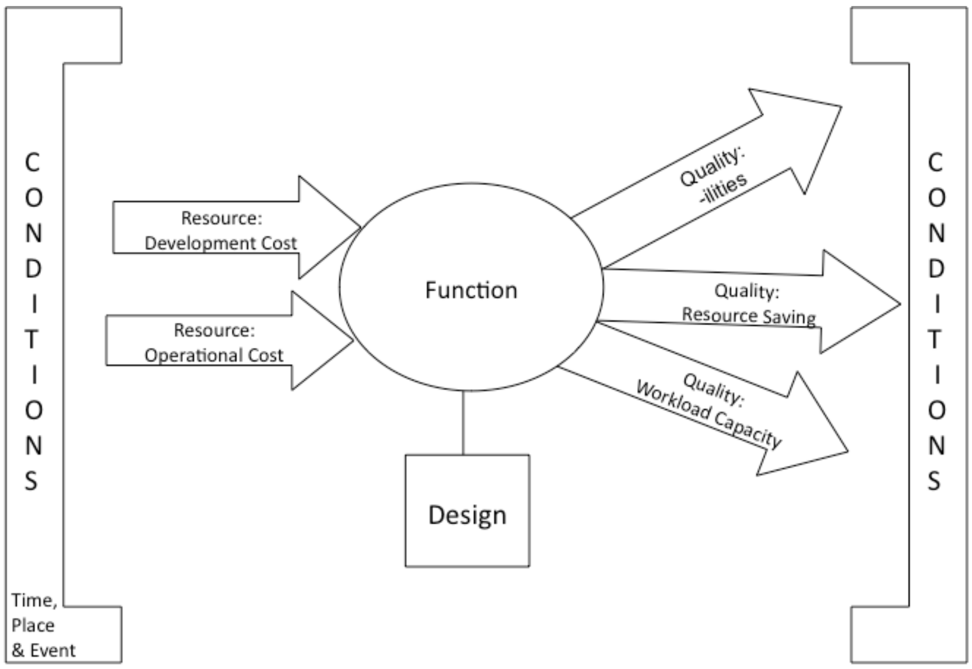


Figure 3.2: The Planguage system model. Developed by the author (relationship between function and performance/quality, and also resource) and Gilb as an illustration for (Gilb, 2005a) and modified slightly here by replacing 'performance' by 'quality'

[6] In Figure 3.2, the system model has been modified to show quality rather than performance at the top-level. This is because quality is the dominant term used in the academic literature. Quality has been broken down here into –ilities, resource saving and workload capacity. Note Planguage uses performance in the way used by the USA Department of Defense (DOD), and performance is then decomposed into quality, resource saving and workload capacity.

resource needed to support the *development* of the functionality with its quality attributes (the development cost) and the resource to enable its *operational* delivery (its on-going cost).

Both quality and resource attributes are scalar and can vary in level. Any particular system function, with stated levels for the quality attributes, implies certain levels of resources are needed. The specific levels for the quality and resource attributes are stated using the relevant scales of measure that have been specified for them. Further detail could be discussed such as the different types of constraint and target levels for these attributes, but for brevity will be omitted here. As mentioned previously, a more detailed account can be found in (Gilb, 2005a).

Further, the functionality with its quality and resource attributes is delivered within specific time, place and event conditions, depicted in the diagram as large enclosing brackets. The Planguage conditions can be seen as setting the system context.

Finally, there is the design, also termed the design solution or the deliverable, which is proposed to implement the functionality with its specified quality and resource attributes. There can of course be several design options, each with specific resource consumption and delivery of specific levels of the quality attributes.

3.5.2 Mapping of the system concepts to the IE table

An IE table is a matrix of requirements against designs, see Figure 3.3. To create an IE table, the quality requirements are placed down the left-hand column. Each quality requirement shows its current baseline (past or 0%) level and the required target (goal or 100%) level. Beneath the quality requirements, the resource requirements are listed. Typically this is the development budget (the resources within which the system changes must be delivered). Then across the top row, the designs are listed with any dependent designs being placed to the right of any design it is dependent on. If the designs are complementary, then the sequencing should be roughly in the proposed implementation order showing a series of increments. If the designs are alternatives, then order is not important. Sometimes designs that are dependent on each other are bundled together to reflect that they have to be implemented together.

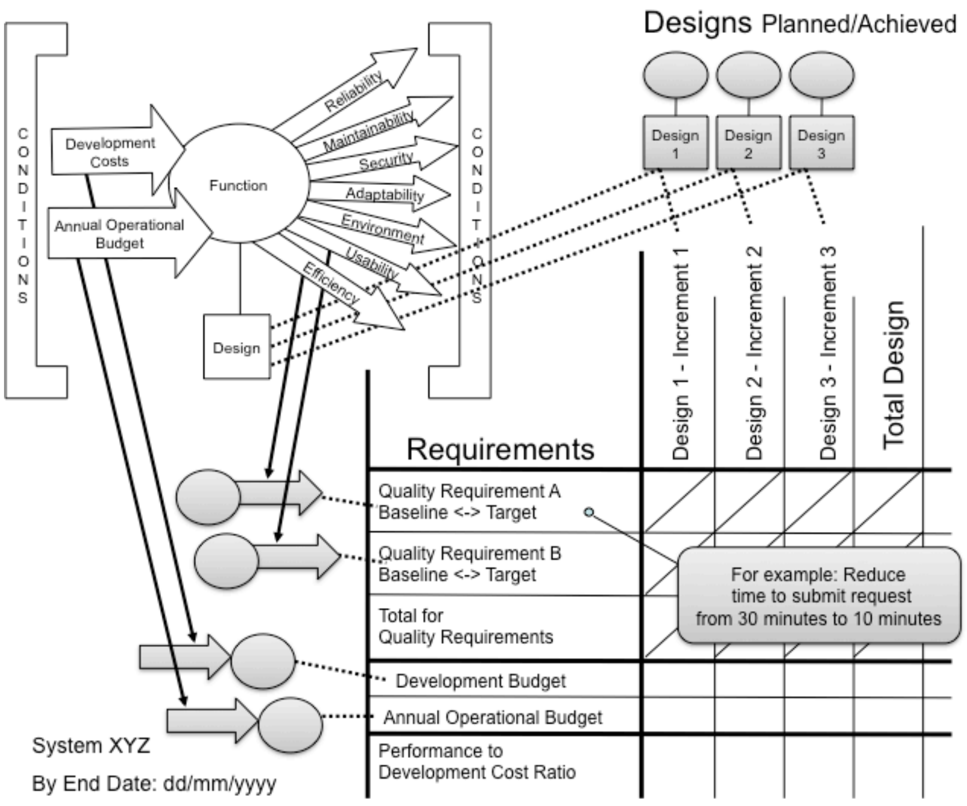
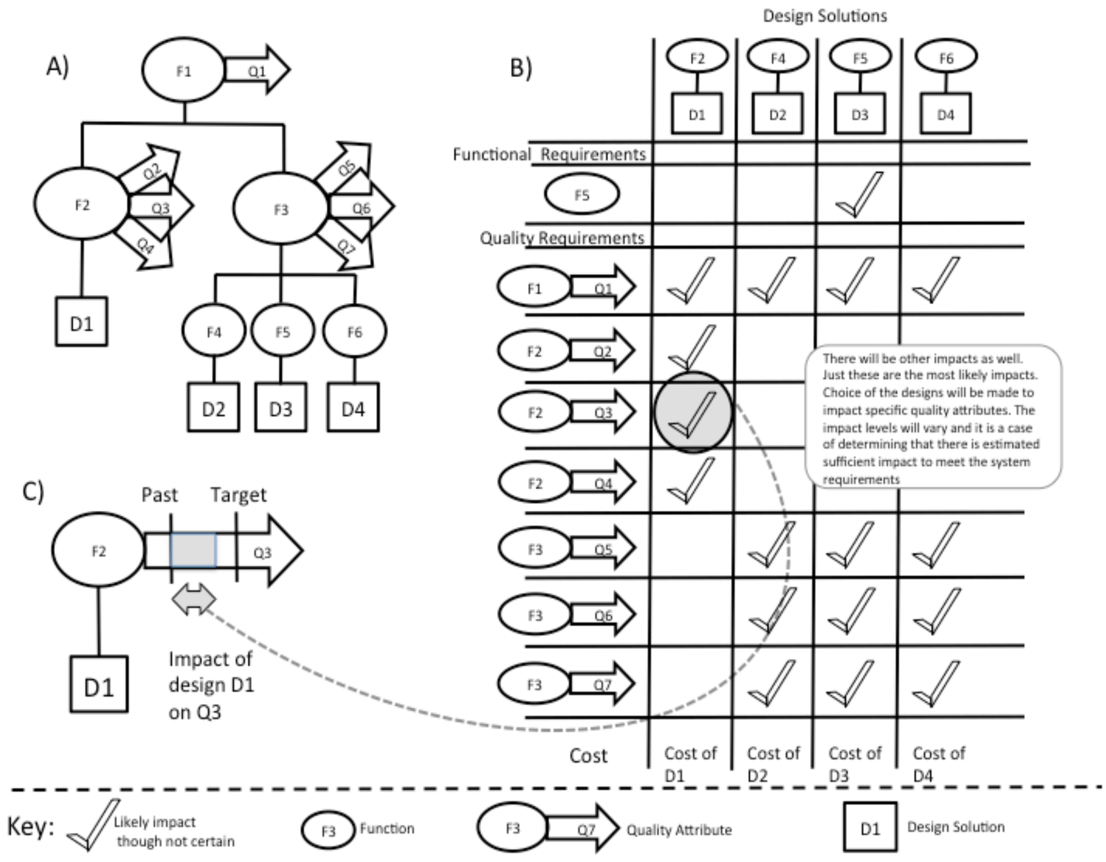


Figure 3.3: Mapping of the Planguage system concepts to the IE table

Figure 3.3, developed by the author during this research, shows how the Planguage system concepts are captured within an IE table. The detailed mapping of the concepts between the Planguage system model and the IE table is shown by the dotted lines. The mapping of the requirements and the design system concepts is straightforward and immediately apparent. The mapping of the functionality is more implicit. Functionality is implied for both the quality and resource requirements in the sense that these attributes only exist in the context of the functionality that they are related to. This associated functionality is *required* functionality when associated with the requirements. In addition, functionality is also associated with the designs: the designs implement certain associated functionality. Here the functionality is expressed as part of the solution. See Figure 3.4, which captures more detail (Again, this diagram was developed by the author during this research).

How functionality is expressed within an IE table is important because system developers are accustomed to focusing on functionality. This is because existing system development methods tend to be dominated by functionality. IE can be considered to be veering in a different direction with its focus on the quality requirements, rather than functional requirements. However the way that IE captures functionality implicitly will suffice where only quality improvements are being sought. Issues arise where there is novel functionality required, and in these cases additional functional requirements rows above the quality requirements can be justified in the IE table. Function *F5* in Figure 3.4 is shown as an example. Given functions are binary in nature, the impact of a design on a functional requirements is also binary (the design either supplies the functionality or it does not). The main issue over including functionality in IE tables is scalability given the number of functions involved can rise rapidly. However, restricting to just novel functions is likely in most cases to solve this.



A)
F1
Q1
F2
Q2
Q3
Q4
F3
Q5
Q6
Q7
D1
F4
F5
F6
D2
D3
D4
B)
Design Solutions
Functional Requirements
Quality Requirements
There will be other impacts as well. Just these are the most likely impacts. Choice of the designs will be made to impact specific quality attributes. The impact levels will vary and it is a case of determining that there is estimated sufficient impact to meet the system requirements
Cost
Cost of D1
Cost of D2
Cost of D3
Cost of D4
C)
Past
Target
Impact of design D1 on Q3
Key:
Likely impact though not certain
Function
Quality Attribute
Design Solution

Figure 3.4: Mapping functionality in an IE table. The impact specifies the percentage between 0% (current level) and 100% (target level) that the design is considered to impact the requirement

Figure 3.4 also shows the impacts of the individual designs towards meeting the individual requirements. An impact can be expressed on the scale of measure (the scale impact) and as a percentage impact (how much impact will this design have in moving the quality requirement from its existing (past) level towards its required target (goal) level?

Expressing the impacts as percentage impacts has the benefit of allowing the percentage impacts to be summed. Such summing has to take into account any obvious cumulative effects. Assuming the designs are cumulative and account is taken of how the requirement level varies after each design is implemented, the impacts along a row when summed reveal how likely it is that a quality requirement's target level (100%) will be met. To make it more certain that the target level is achieved, a safety factor can be included. For example, a safety factor of 2 would mean that sufficient designs are needed for the percentage impacts across a row to be greater than 200%. That is, additional designs are being identified to be sure of meeting the target level (of course that doesn't mean all these designs are going to be implemented).

By looking down a column for a given design, you can see which quality requirements it is contributing towards meeting. By summing the estimated percentage changes in the quality requirements down a column for a given design, and then dividing by the estimated development cost of the design, an estimated cumulative quality to cost ratio for each design can be calculated.

The quality to cost ratios for the potential designs within an IE table can be compared to determine which design offers the most impact given its cost. For several reasons, this is only a rough guide concerning the relative benefits, but it serves to draw attention to any extreme designs (in terms of cost and/or benefit). Assuming the designs are complementary, the aim is to sequence the implementation order to deliver the highest value as early as possible. Any design interdependencies will also have to be taken into account.

Another point to make about the mapping of the system model to the IE table concerns the conditions (context). An IE table is a snapshot of a system at a given instance. So a specific set of time, place and event conditions are defined for an IE table and those apply to the calculations made. Over time, particularly as designs are implemented, the system changes and these changes can be reflected in an IE table, including updating the conditions.

An IE table can be used to capture the actual results of implementation. Noting how the estimates were matched by reality is useful, and progress towards meeting the system requirements can be monitored and planned for by reprioritizing using the actual results.

There is one final point to make about the structuring of IE tables. For any system, there can be several IE tables at different management levels, typically three: strategic, programme and project. There is a cascade down the levels with one table’s designs becoming the requirements for the next table down (This point will be returned to later in this chapter).

The key point being made in this section is that an IE table includes all the main system concepts, requirements/objectives, designs, planned increments/deliverables and can capture the delivered results. Most other prioritization methods fail to capture or utilize all the system concepts, and they fail to consider the level of system detail (such as a change in the level of a quality attribute) that IE includes. In fact, while creating an IE table, the level of detail required means that it is quite likely that the need for additional designs, additional requirements and indeed, modified requirement goals are identified.

3.6 The need to improve IE

First considering the findings of the literature review on the reported issues with prioritization methods, which concern stakeholder value, multiple stakeholder viewpoints, coverage of all the system concepts, abstraction, interdependencies, reprioritization, scaling-up and communication of information. As has been discussed earlier in this chapter, IE has strengths in coverage of the system concepts and communication of information, but it has weaknesses in its handling of explicit stakeholder value.

IE handles abstraction by handling a system at various management levels and also by decomposition of complex concepts. It also can handle system concepts at different abstraction levels side-by-side given its matrix format. The main issue is if overlap occurs.

Interdependencies as discussed in the literature review are of several different types, abstraction providing one type. In the main, it is left for the systems analyst to be aware of any interdependencies. Where there are implementation interdependencies for the designs, the implementation order of the designs should reflect it. One issue that IE doesn’t handle is if there are interactions between the impacts. For example, a design that implements security features could interact with the impacts of another design that introduced usability features. IE considers the designs and indeed the requirements as being independent, the rationale being that as long as all the estimated quality levels are reached then the system is meeting its demanded requirements.

IE supports reprioritization without too much effort. The impacts of the various designs on the requirements do not alter, it is just the changes that need consideration and then the final calculations of the IE table need reworking depending on the changes that have occurred. There is no need to revisit the stakeholders over the estimates that they have already given.

IE handles scaling-up by selection of the key requirements. It does not cope well with regards communication if the IE table matrix becomes too big. Structuring to allow drilling down would be one means of tackling this, but this would require support by application tools.

Abstraction, interdependencies and scaling-up are all inherent aspects of the system data. To some extent, system quality models (such as ISO/IEC 25010:2011 (ISO/IEC 25010:2011, 2011)) could help with understanding and catering for abstraction and requirements' interdependencies by helping structure the software qualities. However for this research, it is the handling of stakeholder value and stakeholder viewpoints that are the issues of immediate interest. Planguage, and by association IE, enables specification of different stakeholder viewpoints. However, there is no specific way in which the different stakeholders are explicitly captured in an IE table. Likewise, Planguage specification does capture stakeholder value against various system concepts: stakeholder value parameters were added to the Planguage templates for specifications during the writing of (Gilb, 2005a). With regards the IE table, expression of stakeholder value can be considered as being captured in the higher level IE table immediately above the IE table of interest. However, it is quite difficult for IE table users to keep such information in mind. So the idea of capturing stakeholder viewpoints and their stakeholder value within one IE table emerged as being highly relevant to improving communication of systems data.

An additional issue observed for some time with IE tables, also adds to the emerging list of enhancement needs. It is that while specification of a required change in quality level is captured, the data about its relevance in terms of its associated function's frequency of use is not explicitly captured in an IE table. For example, reduction in the time to carry out a process can be stated as a quality requirement, but no data is typically stated about how often the different stakeholders carry out that process. Frequency of use significantly affects stakeholder value. The implication at the current time is that an IE table is treating all the quality requirements as being of equal importance. Capturing stakeholder value could permit differentiation.

Another concern involves the assumption made that stakeholder value increases linearly with increase of quality levels. However, one safeguard is that the required quality levels are chosen for specific reasons associated with stakeholder value: the quality levels are in fact likely to reflect some implicit notion of stakeholder value.

3.7 Extending IE to cater for multiple stakeholders and stakeholder value

As shown in Chapter 2, multiple stakeholders, by role and/or by location, while all having an interest in the system, do not share the same interests. For example, the concerns of an IT systems architect for a system will differ from those of say, a senior business manager. An IT systems architect will be interested in issues such as how easy it is to upgrade the system or expand its workload capacity. However, a senior manager is probably considering strategic business alignment and competitive advantage. Both are valid viewpoints, but they differ. Further, the different stakeholders give rise to different stakeholder value and therefore different priorities, which means that some form of negotiation and decision-making process is necessary.

The question is how best to enable an IE table to explicitly capture the different stakeholder viewpoints and their associated stakeholder value? A decision was made to capture the different stakeholders in columns to the left-hand side of the requirements' column. This would enable stakeholder value to be captured for each stakeholder against each requirement. The total stakeholder value for a stakeholder could be found by summing down the column for that stakeholder. The total stakeholder value for a requirement could be found by summing horizontally across all the stakeholders. The stakeholder values would assume 100% of each of the requirements was being delivered. When assessing the impact of a design, an assumption was made that the same percentage of the impact of the design on the requirement could be claimed for the impact of the design on stakeholder value. The explicit stakeholder value for the impact of the design on the requirement would not be shown in the table, but it would allow calculation of stakeholder value to cost ratios, which could then be compared with quality to cost ratios. The expectation being that the two types of ratio would produce different numbers. The value to cost ratios ought to reflect more accurately the impact on the organization.

Other expectations to test include:

- That the different stakeholders will show varying interest across the set of requirements (and some requirements they will be less able to assess)
- That the different stakeholders will show differences in stakeholder value for each requirement

Knowledge of both the amount of stakeholder value and its variation amongst the stakeholders is important for prioritization decision-making. The IE table without such stakeholder value data merely presents a list of requirements for the development project to deliver. As one consultant observed, “It is a project manager’s list of things to implement.” By adding stakeholder value, the intention is to add more information to assist prioritization. The project manager should be provided with information about where the stakeholder value resides and which stakeholders are going to be impacted. Add to that the increment planning, and the project manager can then understand when stakeholders are to be affected and benefits actually delivered. In most circumstances, design interdependencies allowing, implementation priority should be decided according to the delivery of the estimated high stakeholder value to cost ratios.

This planned extension of adding the stakeholders and stakeholder value to IE seems easy for users to understand. In fact, by introducing the stakeholders, some of the data becomes more closely aligned to the different stakeholders, and ought to make the table more relevant to them. Other benefits being it is immediately apparent who benefits from the implementation of any specific requirement and how much total stakeholder value any specific stakeholder is likely to obtain as a result of the proposed system changes. The idea that system costs might then be levied against the stakeholders receiving the most benefits is also considered a possibility.

To reflect the addition of stakeholder value, it is decided to call the extended IE table, a value impact estimation (VIE) table. The VIE method was subjected to proof of concept trials by carrying out three case studies, as will be discussed in Chapter 5.

3.8 Towards an IT prioritization process with VIE

Finally, having created the VIE method, its prioritization process is specified by extending the IE process, see Figure 3.5. It is noted that further work is required in how the prioritization factors are utilized and how the stakeholder value is calculated. This is left for the case studies to clarify. The next chapter will describe the research design setting out how VIE is to be evaluated.

Prioritization Process for VIE

1. Specify the time period(s) involved. Also specify the required safety margins.
2. Identify the stakeholders.
3. Identify the main quality requirements.
4. Create a requirements hierarchy that breaks down the quality requirements into their component parts.
5. Identify/develop scales of measure for the lowest-level quality requirements.
6. Identify/develop the ways by which to measure the levels in a practical, cost-effective way.
7. Map each quality requirement to stakeholders and determine the associated stakeholder values by considering the prioritization factors. Consider how the stakeholder value varies as the quality requirement's level varies.
8. Establish the baselines (current levels) for the quality requirements.
9. Consider competitors' levels and trends.
10. Set target levels with lowest acceptable and highest acceptable levels (= acceptable ranges) for the quality requirements.
11. Considering the constraints and the required functionality (ideally start with the high-level functionality), identify/develop some proposed designs. Note any interdependencies and if possible, sequence interdependent designs accordingly and/or group dependent designs into supersets (this is dependent on when the designs can deliver stakeholder value). Are any designs alternative choices?

12. Sequence designs into a potential series of increments.
13. Estimate the impact of each design on each quality requirement. Specify as a level on the scale and as a percentage (the impact percentage - using the baseline level as 0% and the target level as 100%). Give an estimate of the uncertainty in the estimate (+/- error margins) and state the credibility rating for the estimate (0.0 = wild guess and 1.0 = certainty). Document the evidence and source of the information. Multiply each estimated impact by its credibility rating.
14. Estimate the costs associated with each design.
15. Sum the relevant percentage impacts vertically for each design to assess the contribution towards meeting the requirements.
16. Calculate the quality to cost ratio for each design.
17. Calculate the value to cost ratio for each design. Where a design delivers some impact on a requirement, consider the stakeholders affected by the requirement and claim the estimated amount of stakeholder value for each stakeholder. For example, if Design D impacts 70% on a requirement that gives Stakeholder A and Stakeholder C potential benefit of 10K and 20K respectively, then you can use a utility function to work out the stakeholder values involved. Assuming a linear function and that Design D is implemented for both stakeholders, Design D can claim as stakeholder value 70% of 10K and 70% of 20K. Divide stakeholder value by cost to obtain the value to cost ratio.
18. Calculate safety deviations for each requirement by summing the estimated impacts horizontally for each quality requirement.
19. Check for sufficient design to meet all the requirements. Is the declared safety margin met for all quality requirements?
20. Also check that there are no selected designs that achieve little impact on the quality requirements – if there are, then maybe there are missing quality requirements?
21. Schedule designs with highest value to cost ratios early. Ensure any design interdependencies are met.
22. Implement the next increment.
23. As each increment is delivered, enter the actual results against the estimated results and observe any deviations. Enter any known changes and recalculate as required, then decide the next increment.

Note, this process is expanded from IE (Gilb, 2005a) to include stakeholder value. As explained earlier in a footnote in Section 3.5.1, the term quality replaces performance. Several additional aspects of an IE table are included here, which as explained earlier in a footnote in Section 3.1 are not discussed further in this research: safety margins, safety deviations, uncertainty and credibility. Also the additional aspects of evidence and source of information are not discussed. See (Gilb, 2005a) for more detailed explanation of all these terms.

Figure 3.5: Prioritization process for VIE

Chapter 4

Research Design

4.1 Introduction

This chapter discusses the relevant research methods and justifies the research design chosen for this research. Decisions regarding the research design had to be made at several levels: First, the underlying epistemological paradigms of the research had to be considered. The term paradigm is used here loosely based on Kuhn (1962) describing how researchers following a similar worldview distinguish themselves from others that follow a different worldview, with the two worldviews being incommensurable, i.e. not translatable into each other. In this context it can be said that that the positivist/post-positivist paradigm and the interpretive paradigm are incommensurable. These paradigms can be seen as epistemologies underlying the strategies of inquiry and research methods, both of which have to be chosen so that they will set the framework for addressing the research question. Then appropriate data collection and analytical techniques have to be chosen so that they will produce data that support or reject the propositions formulated previously. The choice of a theoretical lens provides a framework for the specific questions to be asked in order to collect the data that will support or reject the propositions formulated in the literature review chapter. Finally, questions of validity, reliability and completeness of the research design have to be addressed.

4.2 Philosophical considerations

The below diagram by Saunders et al. (2000) provides an overview of the various choices that need to be made when creating a research design, see Figure 4.1. Appropriate selections for the research design have to be made from each of the layers.

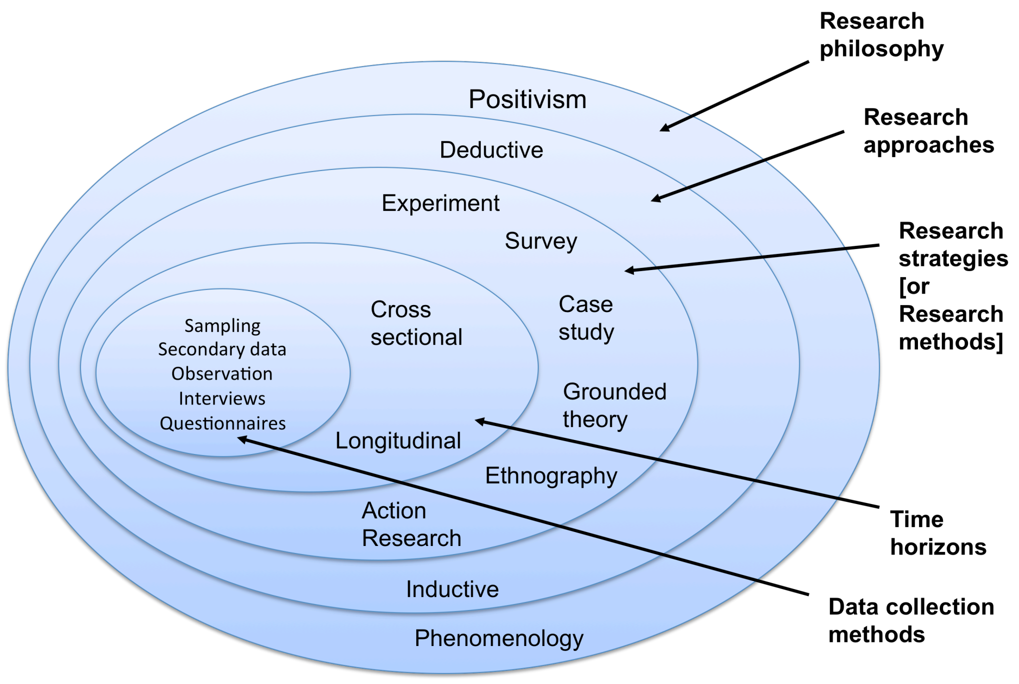


Figure 4.1: The research design 'onion' by Saunders et al. (2000). Note this diagram does not show all options for research methods and data collection methods

First, a series of questions about the research philosophy and the research approach have to be answered. The first consideration is to determine the appropriate research philosophy: do we need a positivistic, post-positivistic or interpretive approach to research the above research questions?

Positivism and post-positivism originated from natural science. Both are hypothesis-based ways of thinking about scientific discovery and laws of nature.

Historically speaking, positivism is a precursor to post-positivism. It requires the formulation of propositions that are subsequently supported or rejected by the research itself.

Post-positivism was developed as criticism of positivism: Instead of seeking support for propositions, post-positivists formulate hypotheses (propositions) and null-hypotheses, which are the logical opposites of the original hypotheses. Instead of seeking support for the original hypothesis, they seek to reject the null-hypotheses, which means that the original hypothesis remains intact.

The "scientific method", widely known in the sciences, is a post-positivistic research method, aiming at the rejection of null-hypotheses, the logical opposite to the original hypotheses. For this purpose, the researcher determines the independent and dependent variables for each hypothesis, collects data in a controlled environment and in order to make inferences from the sample to the wider world statistical methods of analysis are employed.

In contrast to positivism and post-positivism, interpretivism does not work with propositions or hypotheses. Instead it looks for insights emerging from the data collected. Interpretivism aims at the understanding of complex and unpredictable situations.

Different theoretical lenses, such as phenomenology, semiotics and hermeneutics, have to be applied to an interpretive research philosophy in order to form a conceptual framework of inquiry for the research (Anfara and Mertz, 2006). While theoretical lenses are mostly applied to interpretive paradigm type research, in positivist type research theoretical lenses can also be used as a conceptual framework for formulating the questions that will lead to the collection of data that will either support or reject the propositions formulated.

The decision between positivism, post-positivism and interpretivism, mostly depends on the ability to formulate research hypotheses and on the controllability of the research environment in which data are collected. From the existing knowledge of IT prioritization, the researcher (the author) was able to formulate research hypotheses. Based on these hypotheses, experiments might have been carried out to determine, if the null-hypotheses would have been rejected. However, the aim of the research was to develop an IT prioritization method for the practitioners' field of IT project management. In order to determine, if the proposed method will work in practice, it was necessary to expose the proposed method to different "live" environments, in which the researcher has very little control. Furthermore, most of the data collected from the projects had to be qualitative; the data required were only available in the form of systems and requirements specifications, i.e. textual data and observations, i.e. transcripts. Few numerical data could be obtained, such as costs, time, number of employees or number of shops and metrics used in the specifications.

This lead to the decision to choose a positivist paradigm combined with a strategy of inquiry that allows the collection of qualitative data. However, these data were not analysed so that concepts emerged from the data. Instead, questions were formulated for the analysis to find instances that either support or reject a set of propositions derived by the researcher (the author).

4.3 Research methods

The strategy of inquiry seen as most appropriate for this research was case study. Oates (2006) states that, if the evaluation of the construct in question involves a real-life context, then case study, survey or action research is an appropriate choice. Action research, ethnography and Grounded Theory have been discounted, as they best serve interpretive paradigm research. This research is to test a suggested method along predetermined criteria, as opposed to seeing categories and concepts emerge from the data collected in the field (Oates, 2006). Furthermore, mostly for practical reasons, the researcher (the author) cannot immerse herself into the project organisations. In this research, theory is not being derived purely from the data collected in the field as there are research propositions in place prior to the field research being conducted. In fact, Yin (2009, p.35) states that theory development prior to data collection is one point of difference between case studies and methods such as ethnography and grounded theory. He states, "For case studies, theory development as part of the design phase is essential." This serves to support the choice of case study as a strategy of inquiry.

As further type of research, such as "design and creation" as described by Oates (2006) and "proof of concept"- type of research were also considered. The two are very similar. Design and creation type research projects focuses on "developing new IT products" with the aim of developing knowledge in the form of, a construct, model, method or instantiation or some combination of these. They are often also identified as "proof of concept" studies, where a construct, model, method or instantiation is proven to be working. Proof of concept studies can involve controlled physical, psychological or pharmacological experiments. In this case a new method is proven to work. There is no computerisation of the method. The instantiation consists of a table that engenders the model underlying the new method. In other words, while there is not a software prototype, this research proves a new concept of IT prioritization by means of case study research.

Case study research according to Creswell (2003, quoting Stake, 1995) involves an in-depth analysis of one or more cases of a program, an event, an activity, a process or one or more individuals in a real-life context over a stated period of time. Yin (2009) defines a case study as “an empirical inquiry that investigates a contemporary phenomenon in depth and within its real-life context, especially when the boundaries between phenomenon and context are not clearly evident.” In this research, use of a proposed prioritization method, VIE is being investigated. This method is embedded within the broader context of an IT system development and implementation environment. The questions to be addressed are:

1. Does the extended prioritization method work in principle?
2. Does it need some alteration to make it work better?
3. How useful is the method in providing support for IT priority decision-making?

More than one case study will be needed to investigate the VIE method to learn what happens in different types of system environment and to see if the findings translate across systems. All the case studies will be carried out in IT industrial or sponsored academic research project settings. The people concerned in priority decision-making typically include senior management, project managers, business analysts and/or requirements analysts. The main focus for this research is on the project managers and/or business analysts as they, as system planners, will be the primary source of the information required.

The time horizons for each case study will be cross-sectional, rather than longitudinal. That is, each case study will present a snapshot (captured as a VIE table) of the system at a specific period in time. The duration will be the time it takes to create and evaluate the proposed prioritization method, VIE.

4.4 Data collection techniques

According to Yin (2009) evidence for case studies can come from many data sources including "documents, archival records, interviews, direct observation, participant-observation, and physical artifacts." Yin stresses the importance of using multiple, not just single, sources of evidence.

For this research, the case studies will use the data collection methods of secondary data documentation, open question interviews and questionnaires. For each case study, existing system documentation such as, the project plans, requirements specifications, and design specifications will be collected and analysed. Using this documentation, the proposed prioritization method will be trialled. Any required additional data and confirmation of the interpretation of the documentation will be obtained during the trials by asking the system planners (project managers and/or business analysts) a number of open interview questions.

Once the results of the using the proposed prioritization method are obtained, the system planners will be asked for evaluation purposes to complete a short questionnaire. The questionnaire will cover the key points of research interest as identified from the literature review.

4.5 The design of the case studies

Yin (2009) identifies five components of a research design for case studies as follows:

1. "a study's questions;
2. its propositions, if any;
3. its unit(s) of analysis;
4. the logic linking the [collected] data to the propositions; and
5. the criteria for interpreting the findings."

***Study questions*:** Yin explains that the form of the question should be in terms of "who," "what," "where," "how," and "why." A "case study strategy being likely to be appropriate for "how" and "why" questions." The main research question posed for this research is a "how" question:

Within Gilb's Planguage method, how can the current provision to support IT priority decision-making (namely, Impact Estimation), be improved?

***Study propositions*:** The propositions define and narrow the focus within the research. Yin explains "each proposition directs attention to something that should be studied within the scope of study." He considers propositions essential for all, but exploratory, case studies, to "move in the right direction" because they reflect the important theoretical issues and point to where to begin looking for evidence. While this research is exploratory, from the literature review several key aspects emerge as requiring specific attention and propositions are therefore stated. The propositions are as follows:

- **Proposition 1:** Current IT projects fail to capture value and priority information adequately in their system specifications
- **Proposition 2:** The use of metrics is beneficial to understanding the quality requirements and for supporting IT priority decision-making
- **Proposition 3:** Identifying stakeholders more explicitly and considering the impacts on stakeholders of the requirements in terms of value and cost is useful for IT priority decision-making
- **Proposition 4:** The additional effort spent structuring the data using the proposed method helps identify and understand the system data and supports IT priority decision-making
- **Proposition 5:** If stakeholders are provided with a functioning IT prioritization method, then a better understanding of the planned project outcomes is the result.

Unit(s) of analysis: The unit of analysis concerns "defining what the "case" is". Yin (2009) insists that a beginning and end point has to be established for each case, and the people within the case must be distinguished from those who are external. He proposes such definition helps "determine the limits of the data collection and analysis."

The use of the VIE method within the selected projects to support IT priority decision-making is the basis of each case study. The beginning point is the collection of the existing relevant system specification documentation and the data analysis of its contents to establish the current state of prioritization data. The main body of each case study is the use of the VIE method using the specification data to create the VIE table(s). The end point of each case study is the analysis of the resulting VIE table(s) and the opinion of the key stakeholders towards the method and its results.

The people involved in the case studies are the researcher (the author) and the project managers or business analysts for the systems. The researcher uses the existing documentation to create the VIE tables. This is not action research as the researcher is not embedded in the organization and working alongside the project staff. The creation of the VIE table is a proof of concept to establish whether the VIE method works with the existing system data and what support it provides for IT priority decision-making.

The logic linking the data to the propositions: Yin (2009) recommends ensuring that prior to the empirical research there is a theoretical underpinning that links the research question to the data collected. He suggests using the propositions to shape the data collection plan. For this research the linking was achieved by using the literature review to identify relevant questions and these question were then linked to the propositions, see Figure 4.2.

Criteria for interpreting the data**:** Yin (2009) considers "the complete research design will provide surprisingly strong guidance in determining what data to collect and the strategies for analyzing the data." For this research, the questions under the propositions (see Figure 4.2) are seen as helping define the types of data to collect, and a significant part of the analysis will be to ask these questions of the collected data.

Research Question:

"Within Gilb's Planguage method, how can the current provision to support IT priority decision-making (namely, Impact Estimation), be improved?"

Proposition 1: Current IT projects fail to capture value and priority information adequately in their system specifications.

- What prioritization methods have been used prior to the case study?
- What system benefits and costs are identified in the original system documentation?
- What stakeholder value is identified in the original system documentation?

Proposition 2: The use of metrics is beneficial to understanding the quality requirements and for supporting IT priority decision-making

- What quality requirements are identified prior to the case study?
- What problems/benefits are introduced by the use of metrics?
- How well are the quality requirements captured?

Proposition 3: Identifying stakeholders more explicitly and considering the impacts on stakeholders of the requirements in terms of value and cost is useful for IT priority decision-making

- How well are stakeholder viewpoints catered for?
- Do stakeholders show different impacts for stakeholder value?
- How well is stakeholder value captured, especially explicit stakeholder value?
- Is it useful to identify stakeholder cost (as well as value)?

Proposition 4: The additional effort spent structuring the data using the proposed method helps identify and understand the system data and supports IT priority decision-making

- How well are the functional requirements captured?
- Can the designs be mapped to the increments?
- Can the system development costs be captured?
- Is return on investment (ROI) captured and how is it calculated?
- How well are the varying levels of abstraction of the requirements handled?
- How well are the requirements interdependencies captured?
- How well are the design interdependencies captured?
- Does the proposed method scale up to handle numerous requirements?
- How well does the proposed method handle reprioritization?
- How well does the proposed method cater for capturing the actual impacts after implementation?
- Is the proposed method overly complex?

Proposition 5: If stakeholders are provided with a functioning IT prioritization method, then a better understanding of the planned project outcomes is the result.

- Does a better understanding of the system result?
- Does the method produce useful results?

Figure 4.2: Linkage from the research question to the propositions to the questions asked of the data

4.6 Validity and reliability

Having considered the philosophical approach and the theoretical logic, the final aspect of the research design to consider is its reliability and validity. Yin (2009) considers there are four tests: construct validity, internal validity, external validity and reliability.

Construct validity is defined by Yin (2009) as "establishing the correct operational measures for the concepts being studied." He states, "to meet the test of construct validity, an investigator must be sure to cover two steps:

1. Select the specific types of changes that are to be studied (and relate them to the original objectives of the study) and
2. Demonstrate that the selected measures of these changes do indeed reflect the specific types of change that have been selected."

Yin (2009) considers that showing convergence of data from multiple sources of evidence within the same case study is one means of ensuring construct validity. Within these case studies there are multiple sources of data as data is collected from the system documentation, from the system planner by the researcher/author during construction of the VIE table, and finally from the system planner completing a questionnaire.

Yin (2009) also identifies being able to establish a chain of evidence from the initial research question through the logical model to the collected data to the case study report. The linking from the research question to the questions asked of the data collected has already been established earlier in Section 4.5, the theoretical logic of the case study research. To ensure a complete chain of evidence, the final case study report needs to take care to maintain a link from the questions and collected data to its findings.

The third means considered by Yin (2009) to achieve construct validity is to share the case study report with the participants of the case study. For this research, this means the system planners must review and agree the reported facts in the case study reports.

Internal validity is only considered relevant in qualitative research "where researchers explore cause and effect relationships" (Johnson, 1997). Yin (2009) also agrees with this, “internal validity is only a concern for causal (or explanatory) case studies, in which an investigator is trying to determine whether event x led to event y.” The researcher has to consider if what they think is the cause is actually the cause or whether some other phenomena are at work.

External Validity or generalizability is needed when a researcher wants to be able to generalize beyond their findings to other times, places and events. As mentioned earlier, this is often not considered a goal for qualitative research: much qualitative research is looking to capture the particular about a research subject and also the settings tend not to be randomly selected. However, Yin (2009) considers there are two types of generalization: statistical and analytical. He only considers analytical applies to case studies where a researcher wants to generalize a set of results to some broader theory. Replication logic is seen as the way to achieve such generalizability. For this research, three case studies shall be used to see if the findings across the case studies are similar.

Reliability concerns the consistency of the results. Under the same conditions will the same result always be given? The emphasis is on repeating the same case study with the same procedures and arriving at the same results. Yin (2009) considers that documentation of the procedures of the case study is the key to reliability. He recommends using a “case study protocol” (an overview of the case study project, field procedures, case study questions and a guide for the case study report) and a “case study database” (a collection of all the relevant documentation) in order to conduct the research in such a way that it could be audited. For this research, writing and following the research design, and maintaining a collection of the documentation are seen as necessary to support the case study research.

Bias is a threat to validity and reliability. *Researcher bias* “tends to result from selective observation and selective recording of information, and also from allowing one's personal views and perspectives to affect how data are interpreted and how the research is conducted” (Johnson, 1997). Johnson suggests two strategies for dealing with researcher bias: reflexivity (researchers should think about how their personal background will affect their research and strategies to address it) and negative case sampling (searching out “examples that disconfirm their expectations and explanations”). For this research, researcher bias could arise from loyalty to the IE method. However, as the IE method was accepted in advance to have faults, and part of the motivation of the research was to find improved methods, then this is less likely. *Respondent bias* occurs when the respondent gives the answers that they think are required by the researcher or someone else rather than replying what they actually think. In the circumstances of this research, this is unlikely to be a major issue as the majority of the questions being asked are factual concerning data capture within a VIE table.

Finally, the protocol template proposed by Brereton et al. (2008) and the checklists that they recommended given by Runeson and Höst (2008) were used to ensure that this research design is complete.

4.7 Ethics and confidentiality

Relevant management permissions will be obtained in advance for the research. All participants will be briefed about the nature of the research and informed that any findings will be reported anonymously.

Any confidential information involved will be sufficiently protected. In certain cases the project information will be sensitive, as VIE tables capture the quality requirements, the financial budgets and the proposed design solutions. Non-disclosure agreements will be put in place to control any such issues.

Care will be taken about the storage of any of the research information and data relating to participants. Any published information or information appearing in the final thesis report will be sufficiently disguised as to be anonymous.

4.8 Overview of the research process

The literature review results in the identification of the existing IT prioritization methods, the known prioritization factors and any reported issues regarding prioritization. Analysis of this information is used to assess the IE method, which in turn is used to inform and develop an extended IE method, VIE. The literature review and the assessment of IE are key inputs into creating the research design. Case studies involving carrying out proof of concept on the VIE method are then carried out using the research design. The results are analysed and evaluated. Part of the evaluation involves asking the case study participants to fill in a questionnaire about the use of the VIE method. Various aspects of the findings of the research are published as papers.

4.9 Conclusions

This research adopts a positivist paradigm in combination with qualitative data collection. The theoretical lens guiding the data collection is phenomenology. The strategy of inquiry chosen is case study research and the data collection techniques are system documentation and questionnaires. Multiple-case case studies are needed to allow a degree of generalization. Within each case study there is an embedded unit of analysis, proving the concept of the proposed extended IE method, called VIE.

The research question and its propositions are expressed and their logical links to the collected data are described. The logical linkage relies on questions emerging from the literature review and the assessment of the existing IE method. In turn, it is these questions that provide the criteria for assessing the results of using the VIE method.

The expected results of the research include:

- An overview of current prioritization methods
- An extended IE method including stakeholder value
- An evaluation by case studies of the use in industry of the VIE method
- Recommendations for further research.

Having established the research design for the study at hand, the next chapter will describe the details of the case studies, how they were selected, carried out, documented and analysed.

Chapter 5

Case Studies

5.1 Introduction

As discussed in the previous chapter, case study research is the most appropriate way to conduct this proof of concept research. In this chapter explanation is given about how and why the cases were chosen, how the case studies were carried out and finally the findings of each case study.

In total seven cases were explored, four of which had to be dropped for a number of reasons, among which were internal communication problems, resistance to change, and hesitation of participation. Those that were finally chosen did not only wholeheartedly agree to participate, but also had:

1. Some experience with, and were agreeable to, using metrics
2. An interest in expressing stakeholder value and in using the VIE method
3. Agreed to give access to the relevant system development data.

In the following sections, the details of each case study are described.

5.2 Case Study 1: A Bank Loan System

5.2.1 Selection of the case study

The first case study is a bank loan system, involving the introduction of decision software to automate the bank's decision-making when agreeing bank loans. Two consultants were approached because they were known to be interested in quantitative decision-making, which lead to the invitation to the researcher to carry out the case study. The two consultants had no prior knowledge of Planguage or the IE method. Two assumptions were made, first that as they were used to handling quantitative data, the use of metrics would not present any problems, and secondly that they would be likely to already have at least some of the required quantitative data for a VIE table.

5.2.2 Background

The system was a proposal to automate the decision process for customers' bank loan requests. There were two aspects needing improvement: the handling of the bank loan business rules and the turnaround of customer bank loan requests. The business rules were taking up to a month to update, which was too long. The existing process involved the back office staff manually updating the business rules on their PCs. Customers' requests for bank loans were taking about five days to process because the back office staff had to input and vet the details, apply the business rules and then process the result. The delay in receiving a response meant that customers were obtaining other loan offers and the bank was missing business opportunities. The overall manual process was also error-prone and led to a significant number of complaints being raised by both the customers and the back office staff.

The proposal was to put in place a web application, which allowed customers to input their loan requests and receive initial responses from the bank immediately. Automating the business rules would allow this: the business rules applied would be up-to-date and more rapidly able to be changed. The back office staff would have more time to spend on processing the accepted initial responses or handling the more complex bank loan requests.

5.2.3 The step-by-step method used for data collection

First a one-hour meeting was held with one of the consultants. The basics of the VIE table were explained and agreement reached that an existing Bank Loan system specification that had been pitched as a proposal could be used. The consultant agreed to provide any missing data as he had been involved in writing the system specification. It was discussed that there would be a need to determine the main aim of the system (the vision), the stakeholders, the quality requirements with their associated metrics, and the proposed designs (solutions). Then that the impacts of the designs on the quality metrics would have to be estimated, and finally the stakeholder value information identified. It was agreed that the researcher should first attempt to draw up a VIE table using the existing completed system specification.

The Bank Loan system specification was sent by email. The researcher then used this specification to create an initial VIE specification: the initial sets of stakeholders, the potential quality requirements and the potential designs. Two business process diagrams were drawn up to help understand the system, see Figure 5.1 and Figure 5.2. These proved helpful in identifying suitable quality requirements. In addition, reference was also made to a generic framework of quality attributes, in this case the Planguage framework proposed within (Gilb, 2005a) (see later discussion).

For each quality requirement, a scale of measure was identified by considering how best to capture what the requirement was trying to achieve (The scales made the quality requirements testable). Using the scales, estimated numbers were then input to capture the estimated current and target levels (the past and goal levels respectively).

Several potential designs were identified, which complemented each other: there were no alternative designs. The dependencies amongst the designs were considered and this lead to the identified designs being split into four increments. The system specification gave the impression of being Waterfall systems development, but as the designs lent themselves to incremental delivery, it seemed worthwhile showing this. The major benefit of an incremental approach being that the operational implementation, which represented a major process change, could be made lower risk by providing a more gradual transition. For example, the new system could be partially implemented in the first instance in the back office: the bank loan results could be returned to the back office for checking prior to being given to the customer. This would de-risk the process as the back office staff could determine if the results were appropriate and monitor if the results reached the customers in a timely fashion. Any issues with system performance could also be covered by these staff as they could step in and process some customer loan requests more manually (as in the old system) if need be.

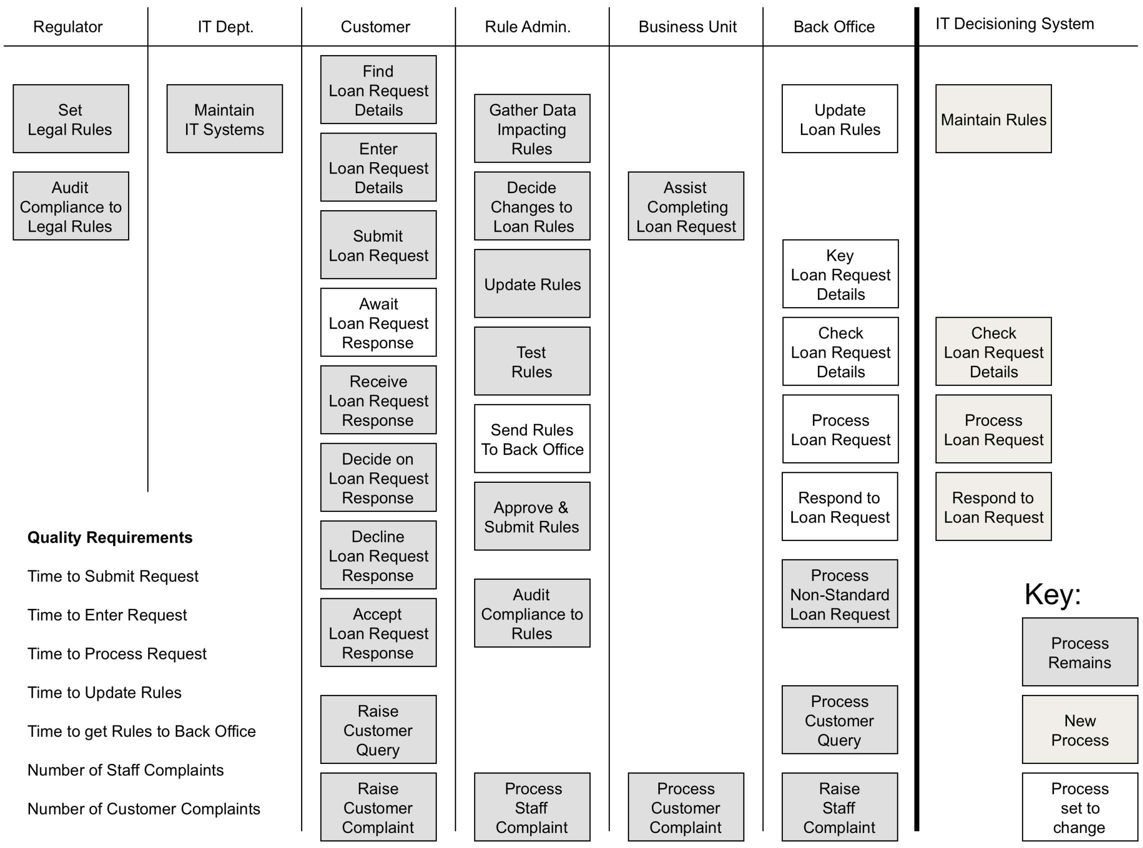


Figure 5.1: Overview of the Bank Loan System

The initial VIE specification was sent by email to the consultants asking them to select which quality requirements and which scales of measure they considered were appropriate. Also to check the proposed designs and provide estimated costs for them.

The consultants spent two hours discussing the initial VIE specification, and emailed back their feedback. As predicted, the consultants had no difficulties understanding the quality requirements and their scales of measure. They had pruned down the quality requirements selecting the ones that they preferred and had even suggested a new quality requirement and scale of measure.

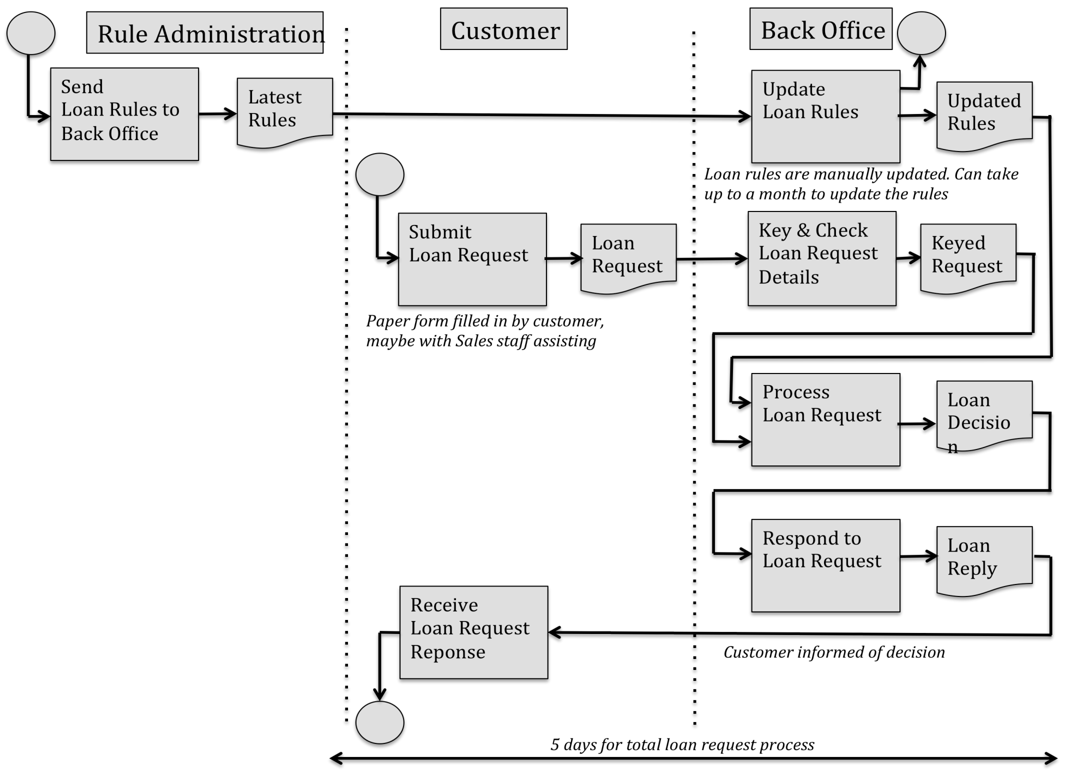

Rule Administration
Customer
Back Office
Send Loan Rules to Back Office
Latest Rules
Update Loan Rules
Updated Rules
Loan rules are manually updated. Can take up to a month to update the rules
Submit Loan Request
Loan Request
Key & Check Loan Request Details
Keyed Request
Paper form filled in by customer, maybe with Sales staff assisting
Process Loan Request
Loan Decisio n
Respond to Loan Request
Loan Reply
Receive Loan Request Reponse
Customer informed of decision
5 days for total loan request process

Figure 5.2: Business process for the Bank Loan System

An initial VIE table was then drawn up and returned by email to the consultants the next day requesting the design impact and stakeholder value data. (The design impact data comprising for each design, its likely percentage impact on moving the level of each of the quality requirements from its present past level (0%) towards the target goal level (100%). The stakeholder value data comprising for each quality requirement, the stakeholder value likely to be obtained by each group of stakeholders).

The consultants replied within half a day. They had filled in the impacts of the designs on the system quality requirements with no difficulty. However, they considered that it was impossible to input the stakeholder value data as the financial data was not available, and anyway would be too sensitive. A telephone conversation between the researcher and the two consultants lead to an agreement that the consultants could estimate the likely staff effort reduction figures given they knew the size of the departments involved and that they could estimate the increased uptake of loan offers. The precise financial data, given the sums involved were large and competitively sensitive, would have to be hidden behind figures of merit. This was accepted by the researcher; the rationale being that determining the exact prioritization result was not essential, and that proof that the VIE table could be completed and the arithmetic calculations carried out was in itself a significant step forward to determining that the extended method was usable.

The consultants completed the table within a couple of hours, and emailed back the results. In total under six hours of the consultants' time had been used. The final task was for the researcher to calculate the ROI values for the VIE table. Two different ROI calculations were carried out for each design. The first was calculating the sum of the percentage impacts of the design on all the quality requirements divided by the development cost (quality-to-cost ratio). The second was calculating the sum of the value delivered by the design for each quality requirement divided by the development cost (value-to-cost ratio).

5.2.4 Results

This section provides the details of the actual results obtained leading to the creation of a VIE table. Any immediate observations or issues are reported here, especially if they impacted on subsequent actions taken. Further evaluation of these results and of the VIE method follows in Section 5.2.5, which identifies the findings.

The results, using a step-by-step approach are as follows:

1. The vision: It was established in the initial meeting with the consultant that the system specification was the only documentation for this system: there was no business case document or feasibility study.

The case study involves a customer business rules "decisioning" system for a bank. As discussed with the consultant, the bank's main objectives are customer satisfaction and, more efficient and effective internal processes. As perceived by the bank, the main processes in need of addressing are two-fold:

- Distributing and using the bank loan business rules: The time, effort, and accuracy of updating and using the business rules
- Processing and responding to customer requests for bank loans: The elapse time taken and the accuracy of dealing with customer requests for bank loans.

These two processes are interconnected as the bank loan business rules are used to determine if a customer qualifies for a bank loan. Specifically, having up-to-date business rules in place impacts the accuracy of the handling of the customer requests.

For the bank system case study, its vision was as follows:

Vision: To reduce the elapsed time to respond accurately to customer loan requests.
Rationale: A faster response will result in more business.

The estimated business benefits were not present in the system specification, which meant an opportunity was lost to engage with the business. Once the system quality requirements were quantitatively specified (see Step 3), the potential business opportunity immediately became more apparent and tangible. In fact, the researcher was actually rather surprised at how the business case brought 'to life' a very technical specification and reflected that it could have helped obtain client buy-in.

2. The stakeholders: The Several distinct stakeholders were identified by role. There was no need to use location to subgroup them further.

Stakeholders: Regulator, IT Department, Customer, Rules Administration, Business Units, Back Office.

3. The quality requirements: The specification is lacking in any testable system quality requirements. These requirements are stated using words such as "up-to-date view", "easy to use rules administration", "low overhead cost", "in a timely manner" and "high performance". Even the acceptable timescales to provide a customer with an answer to a loan request are not specified, so there is ambiguity about the required time saving.

See Figure 5.3. The top of the figure shows a generic Planguage hierarchy of the quality requirements considered relevant for this system (that is, a reduced hierarchy).

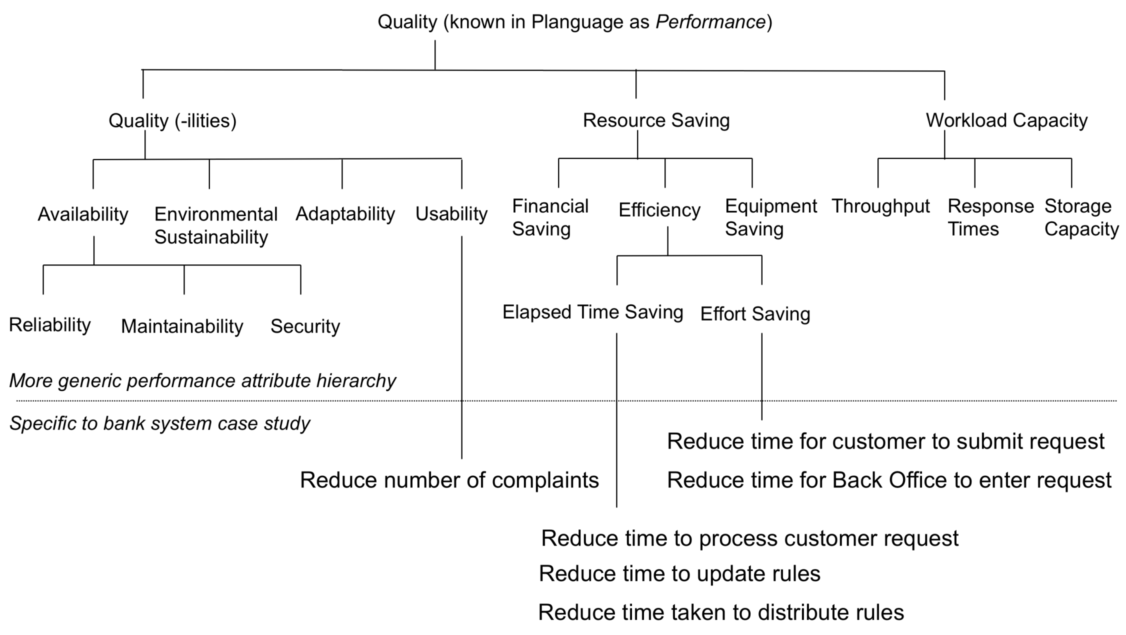

Quality (known in Planguage as *Performance*)
Quality (-ilities)
Resource Saving
Workload Capacity
Availability
Environmental Sustainability
Adaptability
Usability
Financial Saving
Efficiency
Equipment Saving
Throughput
Response Times
Storage Capacity
Reliability
Maintainability
Security
Elapsed Time Saving
Effort Saving
More generic performance attribute hierarchy
Specific to bank system case study
Reduce time for customer to submit request
Reduce number of complaints
Reduce time for Back Office to enter request
Reduce time to process customer request
Reduce time to update rules
Reduce time taken to distribute rules

Figure 5.3: Hierarchy of quality requirements[7]

The bottom of the figure shows the specific bank quality requirements. The consultants chose to focus on the efficiency of the process, and so selected requirements associated with saving resources. This is appropriate given the main system focus was on developing the decisioning software to speed up the bank loan decision process. They also selected one attribute of usability: the number of complaints received about the loan request process.

4. The relevant scales of measure for the quality requirements: Given the choice of system quality requirements, the scales of measure chosen were straightforward involving time saved or number of complaints.

5. The levels on the scales of measure: The consultants accepted the figures input: they made only some minor changes. See the Planguage specification listing in Appendix A or Figure 5.4 to see the full set of system quality requirements with their current (Past) and target (Goal) levels.

6. The potential designs: Several designs are identified and split into four increments. The first increment, 'Automate Rules + Manual Testing' (APTM) proposed to automate the updating of the business rules, but retained manual testing

[7] Rather than using the Planguage term 'performance', the term 'quality' is used in this document. In Figure 5.3, it can be seen how the terminology differs.

on them. The second increment, 'Back Office Loan Decisioning' (BD) automated the application of the business rules. The third increment, 'Web Self-Service' (WSS) introduced the customer web application, and the fourth increment 'Automate Rules and Automate Testing' (APAT) introduced automated testing of the business rules.

The dependencies between the increments are shown on the VIE table as bold arrows linking the relevant designs. The proposed increment ordering copes with the dependencies amongst the designs: BD is dependent on APTM, WSS is dependent on BD, and APAT is dependent on APTM.

Putting all the above elements together for the bank system case study gives the Planguage specification detailing the proposed system changes; see in Appendix A, Planguage specification for the Bank Loan System. Using this specification, a VIE table can be drawn up, see Figure 5.4. In Figure 5.4, the non-shaded area is a basic IE table and the shaded area represents the extensions to IE to create a VIE table.

It is appropriate at this point to provide some further explanation of the VIE table calculations. The calculations of the quality-to-cost ratios and the value-to-cost ratios are of specific interest. In Figure 5.4, by looking down a column for a given design, it can be seen which requirements it contributes towards meeting. The estimated quality-to-cost ratio for each design is calculated by summing the estimated percentage changes in the quality requirements down a column for a given design, and then dividing by the estimated development cost of the design. For example, the design APTM is considered to reduce complaints from the Back Office by 50% (from an average of 10 per week down to 5 per week), reduce the time taken to update the rules to two weeks (approximately 50% of the way from one month to 1 day) and reduce the time taken to distribute the rules to one day (100%). So the sum of the estimated percentage impacts achieved by the design is 50% + 50% + 100% = 200%. Given the development cost for the design is 0.2 M, the estimated quality-to-cost ratio for APTM is 200% / 0.2 = 1000. The quality-to-cost ratios for the potential designs within an IE table were then compared to determine which design offered the most impact given its cost. For several reasons, this is only a rough guide concerning the relative benefits, but it serves to draw attention to any extreme designs (in terms of cost and/or benefit). In this case study, the designs were complementary (rather than being alternatives) so the aim was to sequence the implementation order to deliver the highest value as early as possible.

Stakeholder Value

Key:
s = seconds
m = minutes
d = days
w = week

Bank System

By End Date: dd/mm/yyyy

Designs by expected Increment with design dependencies

Regulator	IT Dept.	Customer	Rule Admin.	Business Unit	Back Office	Total Value / Benefit	Quality Requirements	1 APTM: Automate Rules + Manual Testing	2 BD: Back Office Loan Decisioning	3 WSS: Web Self-Service	4 APAT: Automate Rules + Automate Testing
		4				4	R1: Time for customer to submit request 30 min <-> 10 min	-	-	10 m 100%	-
					3	3	R2: Time for Back Office to enter request 30 min <-> 10 min	-	-	0 m 150%	-
		9		9		18	R3: Time to respond to customer request 5 days <-> 20 seconds	-	1 d 80%	20 s 100%	-
					1	1	R4: No of Back Office complaints 10 per week <-> 0	5 50%	<1 90%	0 100%	(2) (80%)
		1			5	6	R5: No of customer complaints 25 per week <-> 5	-	15 50%	5 100%	-
1			5	4	8	18	R6: Time to update business rules 1 month <-> 1 day	2 w 50%	-	-	1 d 100%
1			3	4	6	14	R7: Time to distribute business rules 2 weeks <-> 1 day	1 d 100%	-	20 s 103%	-
2		14	8	17	23	64	Cumulative Total for Quality Requirements	200%	170%	280%	50%
							Development Budget 2.5M <-> 300K	2.3	2.0	1.0	0.5
							Development Cost for Design	0.2	0.3	1.0	0.5
							Cumulative Quality-to-Cost Ratio	1000	567	280	100
							Cumulative Value-to-Cost Ratio	23.5/0.2 =117.5	17.8/0.3 =59.3	13.7/1.0 =13.7	9/0.5 =18

Figure 5.4. The VIE table for the Bank Loan System

Looking at Figure 5.4, it can be seen that the designs are in descending order regarding the figures for the estimated cumulative quality-to-cost ratios. That is, design APTM has a quality-to-cost ratio of 1000, which is higher than that for design BD, which is 567, which is higher than that for WSS, which is 280, which in turn is higher than that for design APAT, which is 100. The totals for the percentage impacts were calculated based on the estimated *additional* percentage impact over the estimated percentage impact achieved after the last design was implemented. So design BD contributes an additional 40% (90% - 50%) over the 50% estimated for APTM, so its total estimated percentage impact is 80% + 40% + 50% = 170%. A further refinement was to cap any percentage impact at 100% for the calculations. So the estimated percentage impact of WSS on the time for the Back Office to enter a request was taken as 100% within the calculations, rather than its estimate of 150%. The rationale for the capping being that the stakeholders only require 100%; while extra impact might be helpful, it is not stated as a requirement.

Figure 5.4 does not show all the possibilities for the implementation order: it only shows the calculations for the implementation order APTM, BD, WSS then APAT. If APAT was to be implemented immediately after APTM, then the quality-to-cost ratio is 30% + 50% = 80% / 0.5 = 160. This is lower than BD, so the choice of BD being implemented first still stands. For the value-to-cost ratio, 50% of 18 is 9 + 30% of 1 = 9.3 / 0.5 = 18.6. So as before, reversing the implementation order of WSS and APAT should be considered.

In addition, given the stakeholder value information is collected it is possible to calculate the value-to-cost ratios for the potential designs. This is shown in the shaded bottom row of Figure 5.4. The calculation was worked out on the assumption that an estimated percentage impact of a design on a requirement will result in the same percentage of the stakeholder value of the requirement being achieved. For example, APTM is estimated to provide stakeholder value of 50% of 1 + 50% of 18 + 100% of 14 = 23.5. Divided by the development cost, the value-to-cost ratio is 23.5 / 0.2 = 117.5. In terms of prioritization, from the results for the value-to-cost ratios, it appeared that maybe the implementation order of the designs, WSS and APAT should be reversed.

The fact that the quality-to-cost ratios and the value-to-cost ratios differed was to be expected, but it is empirical proof that this is indeed the case and that it impacts the priority implementation order. Value-to-cost ratios given that they use the stakeholder value data are the ratios to use if wanting to achieve the overall highest value system benefits.

5.2.5 Findings

This section discusses the results and gives an account of what was found out during the case study against the research propositions.

Proposition 1: Current IT projects fail to capture value and priority information adequately in their system specifications.

- What prioritization methods have been used prior to the case study?

None.

- What system benefits and costs are identified in the original system documentation?

None.

- What stakeholder value is identified in the original system documentation?

None.

No explicit value or priority data is present in the system specification. However, the lack of priority data is because this system is specified as a Waterfall project. Given the scale of the planned process change, this is puzzling from the perspective of risk. It probably reflects a lack of consideration of the operational implementation. As discussed earlier, this research revealed there are opportunities to implement the system changes in a more iterative, evolutionary or agile manner (that is, a non-Waterfall approach) involving a sequence of increments, which would mitigate risk. Use of increments would help as implementing and delivering smaller ‘chunks’ of system development ensures the developers focus more attention on a specific area. Economies of scale apply as it is then easier to identify any changes that have caused problems and easier to remember how the code, that has only just been written, actually was intended to work. Also it is easier to control smaller changes taking place in the operating environment, and to support the users, who are subject to the changes, adequately. In addition, delivery can start to occur at an earlier date and so business benefits can be realised earlier. Feedback can also be obtained from the live use and changes can be made to correct any unexpected problems or additional requirements that emerge.

The absence of the value data is seen as more significant. Without discussing the system benefits and costs, any justification of the IT investment is lacking. This system specification is relying on the discussion of the system requirements to convey the benefits and this is inadequate. It is revealing that when requested, the additional required value data could be estimated. This shows that it is possible to extend software engineering methods to cover value. If there was more discussion with the stakeholders regarding value, perhaps even more accurate data could be obtained?

Proposition 2: The use of metrics is beneficial to understanding the quality requirements and for supporting IT priority decision-making.

- What quality requirements are identified prior to the case study?

As mentioned earlier, the quality requirements in the system specification are expressed fairly implicitly without use of any metrics (for example, “in a timely manner" and "high performance"). There were no quality requirements explicitly specified.

- What problems/benefits are introduced by the use of metrics?

The main issue with introducing metrics was the estimation of the scale levels (numbers). While it is not difficult to estimate these levels, more discussion with the stakeholders over their setting would have given more confidence.

- How well are the quality requirements captured?

Only a few quality requirements were actually selected for use in the VIE table. It was identified from Figure 5.3 that most of the quality requirements concerned resource savings: saving of staff effort or saving of time. The savings of time involved reducing both the processing time and the time spent waiting for processing.

This is an interesting aspect as typically system specifications try to cover the entire scope of a system – in fact it could be argued that this is one of their prime objectives. Yet here the observation is made that only a limited number of key quality requirements were actually needed. Of course, the skill is in identifying which are the main quality requirements that capture the required system changes. However, it does present an opportunity to save time when specifying a system – the realisation that a narrower focus can still drive change.

Proposition 3: Identifying stakeholders more explicitly and considering the impacts on stakeholders of the requirements in terms of value and cost is useful for IT priority decision-making.

- How well are stakeholder viewpoints catered for?

Identifying the stakeholders explicitly on the left-hand side of the VIE table presented no issues.

- Do stakeholders show different impacts for stakeholder value?

The stakeholders are shown to have different viewpoints in that their stakeholder value from the different quality requirements differs. It was easy to see the differing stakeholder value across the stakeholders and across the quality requirements. Calculating the value-to-cost ratios using the stakeholder value presented no issues.

- How well is stakeholder value captured, especially explicit stakeholder value?

The stakeholder value identified is for the estimated financial savings for staff effort and the estimated additional sales of bank loans. This was justified on the basis that the chosen two value dimensions produced sufficient value to justify the IT investment (that is, the value exceeded the development costs). It is worth asking the question as to whether considering additional dimensions of value would have altered the results. However, on examining the other proposed dimensions of stakeholder value there are no immediate candidates that would have had significant impact. One that was identified as maybe impacting was time-to-market. However, that would be beyond the knowledge of the two technical consultants as it would involve understanding the competitive offerings. This reinforces the benefit of involving other stakeholders in the discussion about stakeholder value.

Another observation is that the stakeholder value has to be considered with regard to the rest of the quality requirement specification. The way that stakeholder value is being expressed is as an additional parameter to the quality requirement. The goal level also is linked to the value: a different level results in a different value. So care must be taken not to separate the stakeholder value and the goal level. Additionally other aspects, such as frequency of usage impact the stakeholder value. This led to identifying that such aspects, such as the workload volumes ideally need explicitly capturing in the VIE table (and they were not in this case study's VIE table).

A further question was raised as to whether the two types of stakeholder value should be captured separately rather than being combined as in this table. This seems worthwhile, the only issue being display of the VIE table becomes rather unwieldy.

- Is it useful to identify stakeholder cost (as well as value)?

Regarding cost, one of the consultants commented that the additional effort incurred by IT was not being captured – which was an interesting observation as organizations often fail to understand the extent of the work that IT carry out (The IT department is often considered mainly as an unwelcome cost overhead). However it is also interesting in that it raises the question as to whether the stakeholder costs incurred by the individual requirements ought to also be captured alongside stakeholder value in the VIE table. There seems no reason why this couldn't be done. A further aspect that was also considered was whether the stakeholder value could be used to help allocate the system development costs – on the grounds that the stakeholders receiving high value could fund the IT investment. Given the actual system development costs and the actual operating costs depend on the chosen design solutions, care would be needed to reflect that when handling the costs.

An IE table currently captures the overall development costs and sometimes the operating costs associated with the individual proposed designs. These costs are currently captured within the main IE table and evaluated against the financial systems development budget (How do the costs of the proposed design solutions impact the overall budget?). They are also used in a simplified return on investment (ROI) calculation (The sum of percentage performance impacts for a proposed design solution divided by the systems development costs for the same design solution gives an idea of the cost effectiveness of the proposed design solution, and allows comparisons among different proposed design solutions).

A proposed extension to handle stakeholder costs would require the system development effort and operating costs associated with each requirement to be assessed and then apportioned over the stakeholders. In fact, all the system development costs would be set against IT, and the operating costs would be assessed for each stakeholder (including IT). So the cost of the selected design solutions would be apportioned across the requirements to IT. That would allow the individual requirements to be evaluated for their cost effectiveness. Finally, having assessed the stakeholder value and the associated stakeholder costs, the *pricing* to the different stakeholders could be allocated along the lines of the stakeholders obtaining the most benefits (stakeholder value) paying the most towards the system development and on-going support and maintenance.

While this proposed extension to handle stakeholder costs, as outlined in the last paragraph, seems worthwhile of trial, it shall be left as a recommendation for further research and not considered further in this research.

Proposition 4: The additional effort spent structuring the data using the proposed method helps identify and understand the system data and supports IT priority decision-making.

- How well are the functional requirements captured?

How the functions are represented in IE is an on-going question. One way of looking at the issue is to see the functionality as expressed though the designs that then impact the quality requirements. However, a crosscheck between functional requirement and functionality to be delivered is also a possibility. Certainly the notion that all systems development is solely about changing quality levels is not true. There could be a case for including some functionality in the requirements column of the VIE table, especially if new functionality was involved. However, for this system, the system changes were all concerned with improving quality levels and the VIE table worked without modification.

- Can the designs be mapped to the increments?

There was no issue with mapping the designs to the increments.

- Can the system development costs be captured?

The system development costs could be estimated against the designs and increments.

- Is return on investment (ROI) captured and how is it calculated?

See previous discussions.

- How well are the varying levels of abstraction of the requirements handled?

Given the system change involved improving two business processes, the requirements could be captured at the same level of abstraction. So no issues over abstraction were identified.

- How well are the requirements dependencies captured?

Care was taken that the quality requirements did not overlap in their scope to avoid the possibility of double counting. The use of the business process diagrams helped ensure there was separation. It really is the concern of the person creating the VIE table to ensure this. The issues are not unique to the VIE table; Keeney stresses the importance of ensuring separation of objectives (Keeney, 2013).

- How well are the design dependencies captured?

The design dependencies are captured in this VIE table through using bold arrows. This technique would not scale well. One mechanism that does assist though is if the design dependency is reflected in the sequencing of the increments. This doesn't help solve the problem for VIE tables when there are possible alternative implementation paths. (In terms of Planguage specification, such dependencies are captured by using a Dependency parameter.)

- Does the proposed method scale up to handle numerous requirements?

The VIE method handles numerous requirements by only selecting a subset of the requirements. Only the key requirements are chosen. There typically are only a handful of key requirements and this was found to be true for this case study.

- How well does the proposed method handle reprioritization?

If the quality requirements have changed then reprioritization involves recalculation of ROI incorporating the new/changed requirements. The rest of the data should remain the same. If there are new/changed designs, then only the calculations relating to them need to be carried out. So reprioritization does not involve major reworking.

- How well does the proposed method cater for capturing the actual impacts after implementation?

The estimates can be supplemented with the actual impacts. (The actual results help calibrate future estimation and should be used in future planning.)

- Is the proposed method overly complex?

The VIE table looks fairly complicated. However, one observation is that most of the data demanded by the method is fundamental to the system: the quality requirements, the designs, the stakeholders, the stakeholder value and the design costs. The only artificial data demanded by the method is the assessment of the impact of the designs on the quality requirements and then the subsequent translation into percentage impacts. This is seen as not excessive given the system understanding gained by using the method.

The amount of time that the consultants had to spent on the case study – given they had no previous knowledge of the method is not excessive (approximately, 6 hours). However, the researcher had to spend considerably more time understanding the system specification to be able to create the VIE table.

Proposition 5: If stakeholders are provided with a functioning IT prioritization method, then a better understanding of the planned project outcomes is the result.

- Does a better understanding of the system result?

Creating the VIE table did result for the researcher in an in-depth knowledge of the system case study. It was the process of creating the VIE table that lead to this understanding, rather than looking at the final ROI results. One benefit of the VIE table is that it presents an overview of the system plans and it allows questions to be asked of the data.

- Does the method produce useful results?

The ROI results confirmed the selected priority implementation order of the designs. The VIE table also revealed an alternative implementation order that had not been apparent before.

5.2.6 Additional evaluation of the use of the VIE table in Case Study 1

In addition to the above findings related to the method itself, a number of practical aspects were also assessed. These were usefulness, power of clarification, ease of use, effectiveness and time efficiency.

The results of this evaluation, which was carried out using a short questionnaire are given in Section B.1, Questionnaire completed by a participant for the Bank Loan System. The participant, a consultant for the system, rated the method between 7 and 9 points out of 10, which is a very positive rating.

Furthermore, the participant's opinion about the performance of the VIE table in this project was asked. Overall, this participant rated the performance of the VIE table very positively with very few aspects needing improvement. In the list of improvements, the participant listed the additional workload and a wish for automation, which would reduce the additional workload. He gave one ambiguous response for the third improvement, "Sometimes far from actual situation". As the VIE table only represents a snapshot of the project situation, he might be pointing at this fact. His overall comment was that the method was "A useful way of looking at developments and on-going improvements".

5.2.7 Main findings of Case Study 1

From the above, the following findings can be summarized.

The system specification originally produced by the consultants lead to the following observations and issues: (Note, explanation of the cross-referencing to the Findings given below is explained in the next chapter.)

- No prioritization method had been used prior to the case study (Finding 27)
- An incomplete set of quality requirements were stated. No metrics were provided for the quality requirements (Finding 4)
- Benefits were not explicitly discussed (inadequate IT investment justification) (Finding 1)
- There was lack of an evolutionary or agile or incremental/phased delivery approach. Such an approach could help reduce risk and assist early value delivery (Finding 2)
- There was inadequate data for establishing stakeholder value (Finding 1)
- There was a lack of consideration of the operational use of the system (Finding 3).

In the process of creating the VIE table certain observations were made:

- The business process diagram was helpful in identifying the requirements and the components of solutions (Finding 5)
- Stakeholder value was treated as a combined value in this case study and it was felt that this was losing information. So it was decided to separate the different dimensions of stakeholder value in subsequent case studies (Finding 16)
- The VIE table needs to capture explicit workload volumes to help with stakeholder value calculations – this was not done for this VIE table (Finding 15)
- It was noticed that it would be possible to capture stakeholder costs explicitly in a VIE table (Finding 9)

- Only a small selection of key quality requirements was necessary to arrive at a sensible prioritization (Finding 26).

Furthermore, the following concepts were validated within the limits of a single case study:

- Use of VIE helps understand the system data (Finding 10)
- Different stakeholders had different viewpoints (Finding 6)
- Stakeholder value could be estimated (Finding 17)
- Different stakeholders obtained different stakeholder value (Finding 7)
- The quality requirements had different stakeholder value (Finding 8).
- The VIE table doesn't easily capture alternative implementation paths. The issue is the expression of the design/increment interdependencies. It worked for this system, but would not for larger, more complex systems (Finding 14).
- The value-to-cost ratios differed from the quality-to-cost ratios (Finding 24).

Finally, the evaluation of the prioritization method and particularly the VIE table by the participant was generally positive. He found the VIE table useful, clarifying, easy to use, effective and efficient. He also rated the performance of the VIE table quite high.

5.3 Case Study 2: An IT Services Bid

5.3.1 Selection of the case study

The second case study comprises an IT service bid to a major high street retailer by a large IT service provider. The researcher was approached by one of the bid managers within the IT service provider who considered using a VIE table might be helpful to capture the details of a bid. In this context he thought it useful and beneficial to investigate the stakeholder value. As part of a bid, the IT services provider produces requirements, design constraints, solutions, implementation plans, costs, prices, risks and benefits, all of which match well with the contents of a VIE table. As service management is a major area for IT investment, this is an interesting case to study IT prioritization. Given the fact that the components of a bid map well to the elements of a VIE table, this case study can further validate the propositions and possibly provide new insights with regards to the use of the VIE table for the preparation of a bid. One aspect that seemed particularly appealing was that the bidding process caused IT staff to be very interested in understanding stakeholder value. The bid managers main aim of this study was to find out, if all the major bid details could be captured in one table to present an overview Such an overview would have to provide sufficiently detailed information to allow a better understanding of the bid and allow questions about the extent to which the IT service provision addresses customer needs and which service components have the greatest stakeholder value.

5.3.2 Background

The bid manager considered that the IT services bid documentation could be made more effective by including a VIE table as an improved overview of the bid. The bid documentation defines the contractual details and is the key definition of the IT services to be provided. It is used to review the bid, to sell the bid, and to plan service delivery. In the case of this IT service provider there is a policy of transparency with the customers leading to some 90% of the bid documentation to be shared with the customers.

According to the bid manager, the problem is that the bid process is “complex, chaotic and creative” and putting together a bid involves considerable staff effort amounting to a “heroic endeavor”. New bid team members are often completely daunted by the demands of the bid process, especially meeting the requirements for the bid documentation. Several attempts have been made to date to improve the situation with external consultants brought in to provide advice. However, the process and the documentation remained too complex; if anything, the bid documentation is now lengthier and more demanding of bid staff effort than ever before: the outline template used to produce the bid documentation is now some 50 pages long.

The questions posed for this case study are how well does the bid specify the IT services to be provided and specifically does the IT services offering, as documented, address the customers' requirements for IT services? It was decided to analyze the existing documentation of a major bid and on this basis to create a VIE table to answer the above questions. The bid manager approached his management and the customer management to obtain the relevant permissions. Both parties agreed and a non-disclosure agreement was put in place. There was one restriction requested by the customer, that the actual financial prices were not revealed. However, a rough indication of the size of the bid can be given by the number of calls from the customer company to the service desk of the service provider is approximately 4800 calls a month involving approximately 300 retail stores. In other words, the service provider was bidding for a major contract.

It is worthy of comment that the bid documentation analyzed was for a successful bid that was won by the services provider. Also, it is noteworthy that the quality of the bid documentation does not necessarily reflect the day-to-day quality of the actual services provision! While the bid documentation has its issues, the services provision runs efficiently. However, as shall be shown, better bid documentation could help improve services delivery by providing assistance with planning.

5.3.3 Data collection

The bid manager provided by email the set of documents comprising the bid. All the names of the organizations involved and the actual financial figures were removed apart from a percentage breakdown of the prices involved. The researcher read through the documentation and a meeting was held to discuss the documentation generally.

Following the meeting, the researcher then started to identify the stakeholders, potential quality requirements and solutions (designs). A further meeting was arranged to discuss those. The bid manager raised the question of capturing the functions and the researcher agreed to go back to the documentation and extract the functionality. This was quite a task as trawling through the documentation revealed a very considerable number of functions. This functionality had not been immediately apparent on initially reading the documentation. One observation is that the bid documentation does not sufficiently explain all the services provision activities and effort that is involved in supporting a customer. The identified functions were grouped under suitable headings and mapped to the solution components. A further meeting was held and it was agreed that the VIE table had to be kept to a manageable size. To reduce the size, a decision was taken to use the high-level solutions and focus on the customer viewpoint. On that basis, the agreed main stakeholders, quality requirements and the solutions were put into a VIE table and the impacts of the solutions on the requirements, and the requirements on stakeholders were marked, initially indicating an impact with a 'Y' (for 'Yes an impact').

The researcher then assessed the impacts of the solutions on achieving the requirements and input some estimated percentages with rationale. This VIE table was sent by email to the bid manager. The bid manager approved the estimated percentages. At this point, the researcher and the bid manager agreed that a useful initial overview had been captured and that several important problems had been identified in the bid documentation. For example, there were inconsistencies in the naming of the organizational units, one key metric was expressed at two different target levels, it was impossible to identify which organizational unit was responsible for one solution or if that solution was in the final bid, and overall the documentation was fairly tangled with key information having to be assembled at times from more than one document.

Having obtained an initial overview, the question was where to go next? There appeared a great deal of functionality that was non-negotiable and simply had to be provided. To resolve this issue, the researcher decided to write up the Planguage specification and to look specifically at the stakeholder value attached to the quality requirements. This proved useful in two ways. First, the quality requirements immediately revealed that they do have very varying stakeholder value – this is to be expected, but it was good to see this confirmed. Also, the different stakeholders were shown to have different stakeholder value – again not a surprise. The additional discovery, which was unexpected, was that the value dimensions of the quality requirements agreed really well with the list of prioritization factors identified originally from the literature: all the prioritization factors apart from strategy were present.

Regarding the stakeholder value, the estimated actual value figures could be calculated in many cases by making assumptions. However, the information for refresh hardware and integrate products simply was not available. So a decision was taken to leave the actual value calculations at that point and to consider what could be decided about priority from the VIE table content. This was done and the bid manager made a final evaluation of the VIE table.

5.3.4 Results

This section provides the detailed results of Case Study 2. The actual process that was carried out for Case Study 2 and the results of this process is described below.

1. The vision: The bid documents the proposed services to be delivered by the service provider covering approximately 300 stores of a high-street retailer. The corporate IT remains separately under the control of the in-house IT organization. The corporate IT management is the sponsor for the services delivery contract: it is paid from their budget. The proposal is for the services provider to take over services management of all the retail stores. This will include asset management, swapping out legacy kit, building systems for new hardware, maintaining the software remotely including virus protection, providing consumables for the stores' printers, and resolving hardware and software calls. All necessary governance and security procedures are to be followed. A service level agreement (SLA) is to be put in place, and the services provider will monitor customer satisfaction by survey and interview. There will be a single point of contact, a dedicated service desk, which is to be run by the services provider. The service desk will log the calls, and attempt to resolve software calls as first time fixes. Unresolved calls are passed promptly by the services desk to the appropriate responder group for them to resolve. Some responder groups are located within the service provider, some are located within the corporate IT of the customer and some are part of third party organizations. The service desk will retain responsibility for monitoring the resolution of all the calls and will flag up any issues in their progress. The vision is as follows:

> Vision: To provide services that meet the customer's needs.
> Rationale: The customer's IT products shall be pro-actively managed to help ensure they provide effective and cost-efficient support for the customer's retail business. That will allow the customer to focus on their business rather than fire-fighting technical issues with their IT.

2. The stakeholders: These stakeholders were identified: see Table 5.1. See also the Planguage specification in Appendix B.

Table 5.1: The stakeholder hierarchy for the IT Services Bid Case Study

Stakeholder Level 1	Stakeholder Level 2	Stakeholder Level 3
Service provider	Service delivery management	
	Service desk	
	Hardware engineering	
	Lifecycle services	
	New Build	
	Service provider resolver groups	Engineering resolver groups
		Software resolver groups
Customer	Retail store staff	
	Retail IT management	
	Retails senior management	
	Data centres and cloud services	
	Customer resolver groups	

3. The quality requirements: The executive summary of the bid documentation contains three pages that outline the benefits of the proposed services. There is only one metric provided: the intended level of cost reduction. This appears the most important metric. However, there is an SLA (service level agreement) specified, which is how the services delivery performance is assessed. There were also some PIs (performance indicators), which are targets set that are advisory, and not included within the SLA. All these metrics, cost reduction, SLA and PI were used to specify the initial quality requirements. See Figure 5.5.

There is reference to how customer satisfaction is measured, via CSAT (a scorecard) and CSIP (the customer satisfaction interview programme). Both together formed the basis for a metric in customer satisfaction, which like the cost reduction, had to be applied across all the services. Additional quality requirements are also specified for governance, security and business continuity planning to recognize their importance.

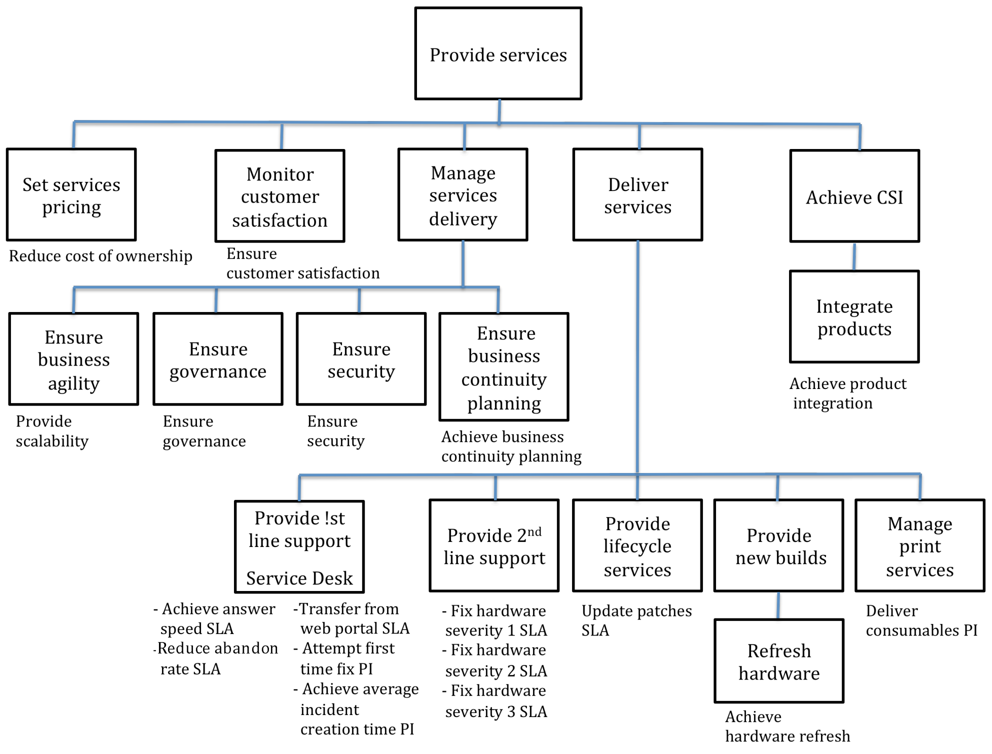
The remaining quality requirements were set up to cover some important services components described in the bid documentation: integration, hardware refresh and managed print services. Integration involves reducing the product portfolio to consolidate to the best software where there is product overlap. Hardware refresh concerns replacing legacy products, especially printers, with more resilience new products. The managed print services run to ensure the printer consumables are dispatched as needed and the cartridges returned for recycling.

Finally, the stakeholder value for each of the quality requirements was considered and added to the Planguage specification (see Appendix B).

4. The relevant scales of measure for the quality requirements: Certain scales of measure are given in the SLA and PI specifications. Other scales were chosen on the basis that they captured the essence of what the customer wanted to achieve from the services provision.

5. The levels on the scales of measure: Some of the target (goal) levels on the scales of measure are given (cost reduction, SLA and PI metrics). What was not known was the current (past) levels, so some impacts were assessed as informed guesses at how well the services provision would improve.

6. The potential designs: The solutions are present in the bid documentation, however there are missing data, which prevent calculation of the precise stakeholder value. While the service desk and hardware support are known elements for the service provider, the rate of hardware refresh and the rate of integration really need more specification. One aspect being that hardware refresh has the capability to significantly reduce the number of service calls. Another aspect being that integration will remove the need to support certain products, which has skill and cost implications. Overall though, proof of concept is present that the stakeholder



Provide services
Set services pricing
Reduce cost of ownership
Monitor customer satisfaction
Ensure customer satisfaction
Manage services delivery
Deliver services
Achieve CSI
Integrate products
Achieve product integration
Ensure business agility
Provide scalability
Ensure governance
Ensure governance
Ensure security
Ensure security
Ensure business continuity planning
Achieve business continuity planning
Provide !st line support Service Desk
- Achieve answer speed SLA
-Reduce abandon rate SLA
-Transfer from web portal SLA
- Attempt first time fix PI
- Achieve average incident creation time PI
Provide 2nd line support
- Fix hardware severity 1 SLA
- Fix hardware severity 2 SLA
- Fix hardware severity 3 SLA
Provide lifecycle services
Update patches SLA
Provide new builds
Refresh hardware
Achieve hardware refresh
Manage print services
Deliver consumables PI

Figure 5.5: Function requirements and quality requirements. The functions are shown using rectangles and the qualities are listed under their relevant function

value could usefully be derived: see the specification of stakeholder value in Appendix A.

7. Planguage specification: The Planguage specification produced in this context can be found in Appendix A.

8. The VIE table: On the basis of the Planguage specification the VIE table for Case Study 2 was created (see Table 5.2).

Table 5.2: VIE table for the IT Services Bid Case Study

									VALUE			REQUIREMENTS								
Services Delivery Management	Service Desk	Hardware Engineering	Lifecycle Services	New Builds	Software Resolver Groups	Customer Retail Staff	Customer Senior Management	Customer IT Management	VALUE-RELATED DATA	VALUE	VALUE DIMENSIONS	CUSTOMER REQUIREMENTS: **Functional Requirements** and Quality Requirements	CUSTOMER QUALITY REQUIREMENTS' MEASURES	Provision of Account and Service Management Team	Provision of Service Desk and trained staff	Provision of Engineering Services: Staff and Spares	Provision of PMO and Lifecycle Services for Ongoing Maintenance	Provision of Design and Development for New Products	Provision of Support for Data Centres and Networks	Provision of Support for Infrastructure as a service (IaaS)
						Staff in 300 stores								8	32	30	17	9	2	1
										Must have		**Set services pricing**		Y						
							Y	Y		GBP of 20-30% cost reduction	Cost reduction	Reduce cost of ownership	% cost reduction in cost of ongoing services Goal=20-30%	10%	5%	5%	20%	30%	0%	0%
Y										Cost of achieving any shortfall in fit	Fit - geographic operation, staff skills, staff resources, spares resources, customer skills									
										Must have		**Monitor customer satisfaction**		Y						
							Y	Y		Overview of the customer opinion of the service provided.	Opinion	Ensure customer satisfaction.	% customer satisfaction from CSAT and CSIP Goal=>80% <- Guess	5%	50%	50%	5%	5%	5%	5%
										Must have		**Ensure business agility**		Y						
								Y		Time and cost involved setting up new contracts avoided. Replaced by CCP meetings	Effort saving	Provide scalability	% ability to deliver required services provision including any ad hoc services to support changes in the retail estate within the agreed timescales Goal=95% <- Guess	10%	5%	30%	30%	30%	5%	5%
								Y		Financial cost of the technology upgrade	Cost									
Y										Income from carrying out technology upgrade and extra service calls	Financial Income									
Y										Exra cost of services provision (spares, staff}	Cost									
										Must have		**Ensure governance**		Y						
							Y	Y		Peace of mind	Legal	Ensure governance	% of specified governance measures in place when audited. Goal=95% <- Guess	50%	10%	10%	10%	10%	10%	10%
										Must have		**Ensure security**		Y						
										Peace of mind	Legal	Ensure security	% of specified security measures in place and up-to-date when audited. Goal=95% <- Guess	50%	10%	10%	10%	10%	10%	10%
										Must have		**Ensure business continuity planning**		Y						
							Y	Y		Peace of mind	Opinion	Achieve business continuity planning	% conformance with business continuity plans when audited. Goal=95% <- Guess	100%	10%					10%
										Must have		**Provide 1st line support (Service desk)**			Y					
						Y			4800 calls per month	Saving effort. Say 2 minutes per call saved = 25K	Effort saving	Achieve answer speed SLA	% of calls answered within 30 seconds Goal=90%		110%					
Y										Potential service credits	Risk (financial loss)									
						Y			4800 calls per month	Saving effort. Say 05 minutes per call saved = 125K	Effort saving	Reduce abandon rate SLA	% of calls answered and not abandoned Goal=95%		110%					
Y										Potential service credits	Risk (financial loss)									
						Y			1440 calls per month?	Saving effort. Say one hour saved per s/w incident = 112K	Effort saving	Attempt first time fix PI	% of calls resolved during first call to the service desk. Target=50%		110%					
Y										Potential service credits	Risk (financial loss)	Transfer from web portal SLA	Goal=90%		110%					
										Might impact on achieving SLA	None	Achieve average incident creation time PI	Time taken to transfer call details to the service desk Target=20 minutes		110%					
										Must have		**Provide 2nd line support**				Y				
							Y	Y	H/W	Business impact of failure	Risk	Fix hardware severity 1 SLA	% of hardware incidents of agreed severity being fixed within 4 hours Goal=90%	10%	10%	110%				
						Y				Will have to work round the faults	Extra effort									
Y										Potential service credits	Risk (financial loss)									
								Y	H/W	Business impact of failure	Risk	Fix hardware severity 2 SLA	% of hardware incidents of agreed severity being fixed within 8 hours Goal=90%		10%	110%				
						Y				Will have to work round the faults	Extra effort									
Y										Potential service credits	Risk (financial loss)									
						Y			H/W	Will have to work round the faults	Extra effort	Fix hardware severity 3 SLA	% of hardware incidents of agreed severity being fixed within 3 days Goal=90%		5%	110%				
Y										Potential service credits	Risk (financial loss)									
										Must have		**Provide Lifecycle services**					Y			
						Y				Reduced number of calls to log and progress	Effort saving	Update patches SLA	% of patching achieved within agreed patching window Goal=95% (Another measure could be the reduction in the number of duplicate incidents occurring)				110%		10%	5%
	Y									Reduced number of calls to log and progress	Effort saving									
					Y					Reduced number of calls to log and progress	Effort saving									
								Y		Reduced dependency	External dependency									
										Must have		**Refresh hardware**					Y			
						Y				Fewer hardware failures to log and progress	Effort saving	Achieve hardware refresh	% reduction in calls due to the replaced hardware products. Goal=10% <- Guess	10%			70%			
								Y		Fewer calls	Cost reduction									
	Y									Fewer calls	Effort saving									
		Y								Fewer calls	Effort saving									
Y										Fewer calls so reduction in call charges	Financial income reduction									
										Must have		**Manage Print Services**			Y	Y	Y			
						Y				Saves retail staff effort in reordering, etc.	Effort saving	Deliver consumables PI	% of consumables delivered the next working day Target=75%		5%	10%	85%			
								Y			Environmental									
										Must have		**Rationalize products**								
						Y				Fewer calls to log and progress	Effort saving	Achieve product rationalization	Cost reduction due to rationalization changes Goal=?	10%				70%		
								Y		Fewer calls so reduction in call charges	Cost reduction									
Y										Carrying out ad hoc upgrades	Financial income									
Y										Fewer calls so reduction in call charges	Financial income reduction									
Y										Fewer products to support	Financial income reduction									
													TOTAL QUALITY REQTS.	255%	670%	495%	340%	155%	40%	45%
												Ongoing costs (Pricing)		Fixed	Banded on calls per month	Fixed	Fixed	Fixed	Fixed	Fixed
												Ongoing costs (Percentage)		8%	32%	30%	17%	9%	2%	1%
												One-off costs								
													TOTAL QUALITY REQTS/COST RATIO	31.9	20.9	16.5	20	17.2	20	45

5.3.5 Findings

This section analyses the results and what was found out during the case study against the research propositions.

Proposition 1b: Current IT projects fail to capture value and priority information adequately in their system specifications.

- What prioritization methods have been used prior to the case study?

None.

- What system benefits and costs are identified in the original system documentation?

The system benefits were described at a high level in terms such as reducing cost of ownership, achieving business agility and single point of contact. If the target service levels are not counted, then only one metric was present, which was the required target percentage of cost reduction. Due to the financial data being too sensitive the financial figures had been removed, so the actual figure is not revealed. However, for the cost information, the framework for the charging was still present, which showed the breakdown of the charges was typically against five separate solution components. There was also a pie chart showing the relative cost ratios across all the solution components. (Note the bid documentation refers to charges and costs, not prices. It is shared documentation with the customer.)

- What stakeholder value is identified in the original system documentation?

Apart from the discussion of the benefits, there is no discussion of explicit stakeholder value. For example, there could have been some discussion as to how the service levels were decided.

Proposition 2: The use of metrics is beneficial to understanding the quality requirements and for supporting IT priority decision-making

- What quality requirements are identified prior to the case study?

There is the overall cost reduction target. Also measuring the customer satisfaction (CSAT and CSIP) is mentioned. The service levels are given with target percentages.

- What problems/benefits are introduced by the use of metrics?

The metrics for service delivery (the service levels) to respond to software and hardware incidents are already in place, so they were no problem. Setting targets for the other services provision was difficult only because the data was not provided. In some cases, the data was not available, for example even the retailer would be somewhat unsure about the rate at which the business is going to grow. In other areas, for example, hardware refresh, the lack of data was more concerning as it should have been discussed in greater depth as it impacted on the cost reduction.

- How well are the quality requirements captured?

As far as the bid documentation expresses the quality requirements, the quality requirements are captured. The main concern is the missing data and this goes further than simply missing data to enable the specification of quality requirements. There is some fundamental data that impacts the cost reduction in the next year and beyond, that is absent. This missing information affects both organizations. The creation of the VIE table identified this problem.

Proposition 3: Identifying stakeholders more explicitly and considering the impacts on stakeholders of the requirements in terms of value and cost is useful for IT priority decision-making

- How well are stakeholder viewpoints catered for?

For reasons of presentation size, the main VIE table only captured the customer viewpoints. There were no issues over there being distinct viewpoints.

- Do stakeholders show different impacts for stakeholder value?

Yes, the stakeholders do experience different stakeholder value.

- How well is stakeholder value captured, especially explicit stakeholder value?

It was identified while capturing the stakeholder value that the VIE table could be modified with two columns for value dimension and value, instead of the one total stakeholder value column, in order to better capture the value dimensions. Individual quality requirements can often have several value dimensions (for example, see provide scalability).

- Is it useful to identify stakeholder cost (as well as value)?

In this VIE table the stakeholder cost associated with the quality requirements was considered in that value was captured that represented the changes in cost and effort. The estimated value figures were not worked through completely because so many assumptions had to be made due to missing data. Some calculations were carried out, see reduce abandon rate SLA. There did not seem to be any issue with this. (It is worth noting that the VIE table captures the customer quality requirements and so doesn't cover any of the services provider's additional quality requirements. The basic costs for service provision are the percentage ratios, which are captured on the bottom right-hand side of the VIE table.)

Proposition 4: The additional effort spent structuring the data using the proposed method helps identify and understand the system data and supports IT priority decision-making.

- How well are the functional requirements captured?

An IE table does not capture the functional requirements. It has been suggested in the past that it could do so for crosschecking purposes. In this case study, the question of capturing functional requirements was raised specifically as it was difficult to understand how to model the basic mandatory services provision. That is, the customer pays to have a services provider available at all the required times to resolve any hardware or software problems, so there's certain functionality provided as a given. This was an important issue as the main aim was to capture an overview of the total services provision. So to resolve this, the functional requirements are documented in this case study's VIE table. They are shown in the requirements column, with a value in the value column of 'Must have'. Note not all the functions are present, only the functions with associated quality requirements. See Figure 5.5, *Function requirements and quality requirements*, which shows the main functions for this case study.

- Can the designs be mapped to the increments?

The on-going mandatory services provision carries on continuously, so that is not divided into increments. The ad hoc services certainly could be mapped to increments and there would be benefit in doing this in order to plan to achieve the cost reduction. The retail annual cycle also needs to be reflected as well, as implementations need ideally to occur at off-peak times.

- Can the system development costs be captured?

In this case study they could only be captured as ratio values as the actual figures were too sensitive. There were parts of the VIE table where some further sub-division of the costs would have been useful, for example the split in costs between asset management and software patching, and indeed all the other services, which were bundled into Lifecycle Services.

- Is return on investment (ROI) captured and how is it calculated?

Summing the percentage impacts of each solution, and then dividing by its percentage cost calculated the quality-to-cost ROI. However, for this VIE table, given the solutions were mandatory, this is really only of minor interest to see the contribution towards the customer quality requirements. Value-to-cost would be of interest, and could be calculated, but more data would be needed.

- How well are the varying levels of abstraction of the requirements handled?

Cost reduction and customer satisfaction are both high level measures that go across all the services provision. However, they are also both requirements with set target levels, so while noting their nature, it was still appropriate to include them as quality requirements. Care is just needed that there is no double counting of the cost reduction.

For the service levels, there were no issues over abstraction found. For capturing other quality requirements, namely CSI (continual service improvement), business agility, security and governance, more detail would have given a better understanding of what the customers wanted to achieve. As noted before, this is an area where the documentation (and therefore an understanding of the services requirements) could be strengthened. These quality requirements require some further decomposition.

- How well are the requirements interdependencies captured?

No problems were identified with requirements interdependencies.

- How well are the design interdependencies captured?

No problems were identified with design interdependencies.

- Does the proposed method scale up to handle numerous requirements?

There were issues in that all the functions and functional requirements could not be handled at one level down of decomposition, as there were too many. Automation would be needed to solve that aspect. The alternative would be to have multiple VIE tables, which might work as different services organizational units are involved and the tables could be split accordingly.

- How well does the proposed method handle reprioritization?

Adding additional rows and columns meant some recalculation within the table, but the existing data was still valid. So reprioritization was not an issue.

- How well does the proposed method cater for capturing the actual impacts after implementation?

Extra columns in the table would be needed to show the actual results against the estimates. A further level of decomposition of the data would be needed for some quality requirements, and for some solutions, to home in on where any under- or over-achievement occurred. However as a top-level overview, the VIE table would give an overview highlighting where any issues existed.

- Is the proposed method overly complex?

The VIE table captures an overview of the services offering in a condensed format. This overview helps understanding of the main elements of the services provision.

However, it is easy to make the VIE table too unwieldy. In this case study, supporting one lower level of decomposition would have captured a more informative picture. Ideally the table should be automated for support drilling down into the data. An automated application would also better support the capture and provision of the rationale information.

For services provision, a specialist application could be of use. For example, vetting bids might be made easier if each bid had a standardised VIE table overview.

Proposition 5: If stakeholders are provided with a functioning IT prioritization method, then a better understanding of the planned project outcomes is the result.

- Does a better understanding of the system result?

Yes, a much clearer picture emerges with the VIE table. The main solution components are immediately visible, and the key quality requirements capture the customer's main requirements. This overview saves considerable time compared to reading through the bid documentation.

- Does the method produce useful results?

Several major problems with the bid documentation were identified while creating a VIE table. Simply reading through the bid documentation was not as effective. For example, two different names were used for the same organizational group, and that only became apparent when trying to identify the stakeholders' functionality. Also there was duplication between a service level and a performance indicator and that was detected while compiling a list of the quality requirements.

In terms of providing an adequate overview, as already mentioned, the top level VIE table is useful to quickly see the major elements of the services provision. The major problem is the lack of sufficient data available to address some of the stakeholder value questions. For example, the basic information about the impact on the retail store of certain software and hardware failures is not apparent. In addition, the split between the number of software and the number of hardware calls is not provided. The speed at which the legacy systems and hardware can be replaced is not known, and yet this is seen by the researcher as being very significant in reducing the number of service calls and also in achieving the required level of cost reduction. By using the VIE table and analysing where the high stakeholder value lies, the high priority issues that need addressing are readily identified.

Regarding the priorities, the main priority that emerges is that it is not explained how the required cost reduction will be achieved. There is insufficient data to be sure that it will be achieved. So the recommendation would be to investigate the plans for *Refresh hardware* and *Integrate products* to determine the level of cost reduction that could be achieved from them.

5.3.6 Additional evaluation of the use of the VIE table in Case Study 2

In addition to the above findings related to the method itself, a number of practical aspects were also assessed. These were usefulness, power of clarification, ease of use, effectiveness and time efficiency.

The results of this evaluation, which was carried out using a short questionnaire are given in Section B.2, Questionnaire completed by a participant for the IT Services Bid. The participant, the Bid Manager rated the method between 7 and 9 points out of 10, which is a very positive rating.

5.3.7 Main findings of Case Study 2

The following findings can be summarized.

The original system specification has the following issues: (Note, explanation of the cross-referencing to the Findings given below is explained in the next chapter.)

- No prioritization method had been used prior to the case study (Finding 27)
- The benefits are discussed in words, but there is only one specific metric, so the IT investment justification is lacking (Finding 1)
- There was a lack of available data to support more in depth value analysis (Finding 1)
- Though there are quality requirements for cost reduction and the SLAs, there are no quality requirements covering the remaining services delivery. Target (goal) levels had to be guessed, for example in the areas of customer satisfaction, business agility, business continuity, refresh hardware and rationalize products (Finding 4)
- It was identified that further planning was required for Refresh hardware and Rationalize products to ensure meeting the cost reduction objective (Finding 11)
- Specifying the quality metrics gave more understanding of the system, for example, the SLA metrics (Finding 28)
- The bid documentation doesn't transparently show the true picture of the amount of work involved in delivering the services (Finding 3)

- The bid solution was not presented with sufficient clarity (Findings 10 and 18)
- Much of the description found in the bid documentation could be offloaded. It belongs in a standard manual of responsibilities and working practices (No Finding).

The creation of the VIE table and the subsequent prioritization process was helpful in the following way:

- Specifying the functions in detail was not required to create the VIE table (Finding 12)
- The VIE table does not scale up to cope with a large number of functions (Finding 20), but note Finding 12 above. If really necessary, automation could solve this issue.
- A simple means of displaying multiple stakeholder value dimensions against a quality requirement was discovered (Finding 19)
- Customer satisfaction was a high-level objective, which together with cost reduction belonged in a higher level VIE table. There is a possibility of double counting if care is not taken (Finding 13).

The following concepts were validated within the limits of a single case study:

- Several significant areas for improvement in the implementation planning were revealed by the exercise of creating a VIE table (Finding 11)
- Missing data was identified by the exercise of creating a VIE table (Findings 1 and 4)
- The stakeholders were shown to have differing viewpoints and differing stakeholder value (though the data to calculate the precise values was absent), and the quality requirements had differing stakeholder value (Findings 6–8)
- Identifying the stakeholder value of the quality requirements revealed an excellent mapping to the prioritization factors identified in the literature review (Finding 17)

- "A better understanding of the bid solution was obtained and the VIE table could be used as an overview of a customer offer or bid solution, as it provides a holistic view of requirements, solution components, costs and benefits." (Bid Manager) (Finding 29).

In conclusion, the practical application and creation of the VIE table was found to be useful, clarifying, easy to use, effective and efficient. The performance of the VIE table was rated high.

5.4 Case Study 3: A Research Project

5.4.1 Selection of the case study

The third case study is a European research project aiming to build a software framework to support functionality (apps) to improve the quality of life for people with Down's Syndrome (DS). Middlesex University is one of the participants and the opportunity arose to use the project as a case study. From the viewpoint of this research into setting priorities regarding IT investment, this project offers the chance to investigate stakeholder value where the high-level outcomes are likely to be measured in non-financial ways.

5.4.2 Background

The European Research Project involves collaboration among several partners across three main countries: Germany, Norway and the UK. In addition, Down's Syndrome (DS) charities in each of these countries are also involved providing advice and coordinating the contact with people with DS and their families.

The systems development is split among three sites in the three main countries: one has overall responsibility for the system framework, another for the user interface and the final one for the context-awareness of the system.

The project has an emphasis on provision of the basic underlying supporting platform to enable third party software providers to later deliver further software. However the project is required to produce some user functionality to trial the framework. Its main aim is to support the social inclusion of people with DS: specifically to assist with communication and supporting travel. Additional user functionality to support learning, work, health and managing money is also outlined, but is a lower priority.

The project involves phased delivery and is planned to have three phases, each of which deliver prototype software. This presents a challenge in that it provides for only three major review points. The danger being that without strong planning within the phases, it is easy to see the system development becoming uncoordinated. A further, even bigger, challenge is that many of the key requirements are lacking detailed refinement as to date little research has been carried out on the use of IT by people with DS: the user interface needs considerable empirical research to establish its precise requirements.

In its early months, the project decided that more knowledge was required about the user requirements before the system design began. A survey was carried out by questionnaire to capture information from the carers, and some interviews were conducted with people with DS. The questionnaire established data about the use of technology; however it did not provide any supporting background detail. So for example, it established that assistance was required with using various types of technology, but didn't explain when assistance was required or how often the technology was currently being used or for what purpose. This case study commenced at the point that the questionnaire results became available.

The researcher (the author) is not a member of the Research Project team. However, the project manager granted permission to carry out the case study, and access was granted to the relevant staff and documentation. As part of the case study, the researcher (the author) contributed to the development and prioritization of a list of system requirements, which are now in use by the Research Project. The main focus of the case study was the system requirements, especially the PU requirements, their resulting stakeholder value and the impacts of the functional deliverables. Detailed lower-level planning of the actual systems development was not carried out.

5.4.3 Data collection

From the existing system documentation, discussion concerning the requirements was at a fairly high level of abstraction. There was a lack of any testable objectives for the project. So the first step of the case study was to produce a list of system quality attributes with the aim of identifying some potential objectives for the project that could be used to direct and measure the project's progress.

At this point as the deadline set for the initial requirements was rapidly approaching, an initial set of requirements for the framework was drawn up by the project. The site responsible for context-awareness carried out most of this specification. The framework requirements list was circulated to the other partners in the project and generated some response from them. One response resulted in more explicit prioritization across the three prototypes being carried out within the case study. The initial division of functionality into the three prototypes was carried out mainly on the basis of development precedence and the stated preferences.

As the framework requirements list was quite high-level and fragmented, a more detailed user functional requirements list was drawn up (For brevity, this list is not included in this document). The aim of this list was two-fold: first to gain a better understanding of the requirements and second to be the basis for more detailed analysis of the priorities. To ensure the user functional requirements were in line with the project's framework requirements, the framework requirements' functional requirements list was cross-referenced against the user functional requirements. (These cross-references appear in the VIE tables, for example, "D1" and "FUN27".)

It became apparent to the case study researcher through conversation that the systems programming was integrated across the three sites and this raised the need for prioritization to coordinate the design and coding effort. Also that the amount of project time allocated for programming was probably limited when set against the project objectives, and this was another reason for better prioritization. From the viewpoint of research into setting priorities, the need for consideration of system development effort for programming was reinforced.

At this point, an initial VIE table was drawn up as further discussion in the next section explains.

5.4.4 Results

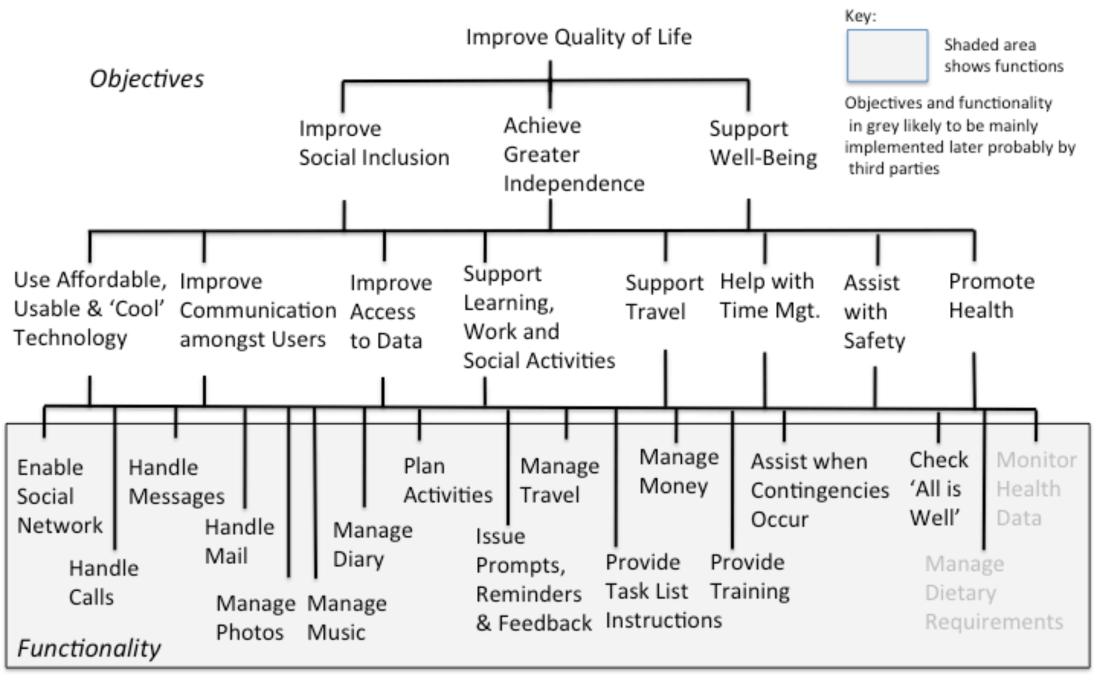
The results for Case Study 3 are as follows:

1. The vision:

Vision: Promoting the autonomy and independence of DS individuals.

The high-level primary user objectives are to improve social inclusion, help achieve greater independence and to support wellbeing, see Figure 5.6, *Primary user objectives and functionality*. The means objectives (Keeney, 1992) to support the high-level objectives include providing support for activities for school, college and work, support activities for leisure at home or in the wider community, support for health-related tasks, and improving IT inclusion by addressing accessibility. There are overlaps among these objectives and so Figure 5.6 shows a matrix structure rather than a hierarchy. The lower portion of Figure 5.6 outlines the functionality required to deliver the objectives.

In addition to the primary user objectives, there are the needs of the other stakeholders. From the viewpoint of designing the system, the other main stakeholders, apart from the primary user, are the IT stakeholders and their requirements to engineer the technical system framework. The resulting framework has to be of sufficiently high quality to adequately support the user apps.


Key:
Shaded area shows functions
Objectives and functionality in grey likely to be mainly implemented later probably by third parties
Objectives
Improve Quality of Life
Improve Social Inclusion
Achieve Greater Independence
Support Well-Being
Use Affordable, Usable & 'Cool' Technology
Improve Communication amongst Users
Improve Access to Data
Support Learning, Work and Social Activities
Support Travel
Help with Time Mgt.
Assist with Safety
Promote Health
Enable Social Network
Handle Calls
Handle Messages
Handle Mail
Manage Photos
Manage Music
Manage Diary
Plan Activities
Issue Prompts, Reminders & Feedback
Manage Travel
Manage Money
Provide Task List Instructions
Assist when Contingencies Occur
Provide Training
Check 'All is Well'
Monitor Health Data
Manage Dietary Requirements
Functionality
Figure 5.6: Primary user objectives and functionality

In addition, the system must also conform to the relevant quality standards, especially any ethical standards: these are specifically important given this system supports users with a various spectrum of disabilities. Finally, from a research viewpoint, there are objectives to advance knowledge and to ensure the visibility of the resulting research.

2. The stakeholders: The stakeholders identified to date include:

(Note: See also Figure 5.7)

- Primary users (PU) – people with Down's Syndrome (DS)
    - Children
    - Teenagers
    - Adults (19% of PU work and 23% of PU attend a day centre)

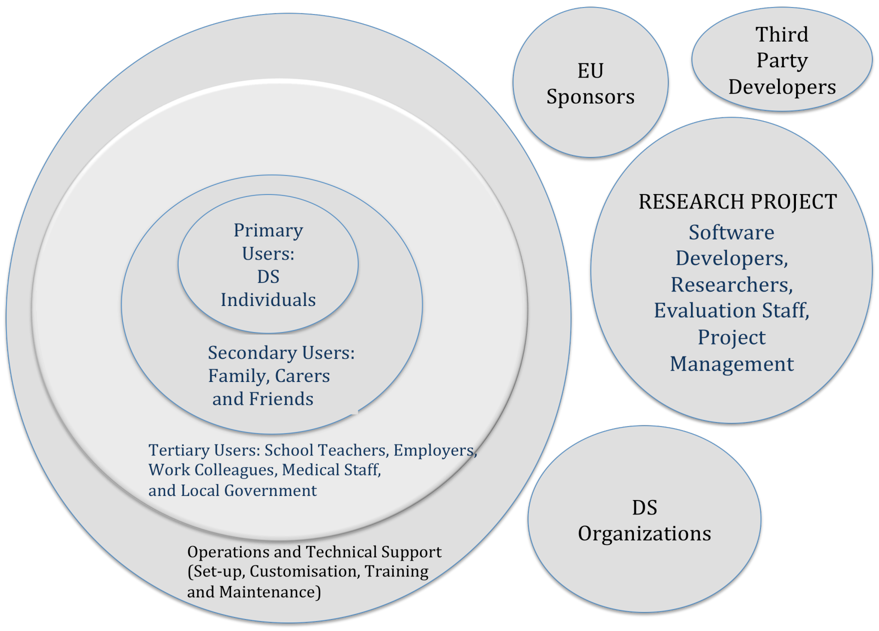

EU Sponsors
Third Party Developers
RESEARCH PROJECT
Software Developers, Researchers, Evaluation Staff, Project Management
Primary Users: DS Individuals
Secondary Users: Family, Carers and Friends
Tertiary Users: School Teachers, Employers, Work Colleagues, Medical Staff, and Local Government
Operations and Technical Support (Set-up, Customisation, Training and Maintenance)
DS Organizations

Figure 5.7: Main stakeholders for the Research Project

- Secondary users (SU) - carers
    - Family (for 85% of PU)
    - Care home (for 3% of PU)
    - Monitoring (as opposed to living alongside) (for 12% of PU)
- Tertiary users (TU)
    - Friends (Note: in their own right some could additionally be PU)
    - Teachers (including day centre staff) (99% of PU aged 10-18 are at school and 23% of adult PU attend a day centre)
    - Employers (19% of adult PU work)
    - Health-related staff (doctors, nurses, dentists, nutritionists, etc.)
- Down's Syndrome (DS) organizations
- Project management
- Project system developers
- Technical support

- Operations
- Research organizations
- Industrial partners
- EU project sponsors
- Legislation
- Third party developers.

3. The quality requirements: The quality requirements in the system documentation are at a high-level of abstraction and, apart from two (see later), are not quantified. They are also very ambitious. While being ambitious is essential for research, the researcher considered that identifying a minimum design solution that provided basic functionality for daily use by the primary users could help by providing a focus for some of the research activity. With that in mind, an initial 'top ten' set of quality requirements was drawn up and some scales of measure were specified.

The main aim of the set of 'top ten' requirements is to help project planning ensure useful features are delivered. In addition, the measures provide a means of keeping track of project progress and they help define project success. The initial ideas for the 'top ten' requirements relate to key aspects of Prototype 1, and they include:

- Supporting more primary users to make mobile calls unaided
- Helping primary users plan in advance local journeys
- Supporting primary users send messages unaided
- Improve the location tracking of primary users when outdoors
- Enable the monitoring of the emotional wellbeing of the primary users
- Ensure the setup time for a system user is practical
- Ensure the system reliability is sufficient
- Monitor the number, type and severity of system defects
- Ensure the awareness of third party developers
- Measure customer satisfaction with the system.

Different or more measures than the initial 'top ten' suggested above could be specified. Once any target measure is achieved, an additional replacement measure should be considered. Suggestions for additional measures include:

- Reduction in the amount of time spent by secondary users assisting primary users organize their school/work/learning
- Reduction in the amount of time spent by secondary users helping primary users with usage of technology
- Increase in the frequency of productive use of technology by the primary user
- Reduction in time spent by tertiary user in supporting primary user to complete tasks (reduction in repetition of instructions, prompting over the next step, etc.).

4. The relevant scales of measure for the quality requirements: For each quality requirement, a scale of measure is selected that best captures the intended improvement.

Measurement of progress will require either interviews or a questionnaire, which for the prototype testing involving a limited number of users is acceptable. It could be that a more sensitive scale of measure is needed to capture the various degrees of support needed by the primary users.

5. The levels on the scales of measure: The results of the questionnaire survey are used whenever possible to set the current (past) levels. All figures are rounded to the nearest whole number. The project decided to set the age group for the primary users as 14 years and above. This was because the main aim is to support independence and social inclusion and the activities of people aged 14 and over best matched with this. The questionnaire had collected information in two age groups: 10-18 years and 19 years or over. These figures are used for the past levels. For the goal levels, guesses were made, based on the past levels adjusted for the different age criteria.

6. The potential designs: The potential designs are taken as the initial prototype sets selected in the initial prioritization using development precedence and the expressed preferences. The cross-reference to the framework requirements specification is given against each design or functional specification.

7. Planguage specification: A Planguage specification for Prototype 1 of the Research Project was drawn up for the selected stakeholders and quality requirements using the information discussed above. This specification is given in Appendix A.

8. The VIE table and its calculations: A VIE table was created using the Planguage specification for the Research Project. Its scope covered the initial top ten quality requirements, and the designs and functionality selected for Prototype 1. A relevant subset of the stakeholders was selected. A decision had been made to focus on the primary users' quality requirements, so that dictated that the VIE table did so too. The quality requirements for the researchers, or the system engineering quality attributes (such as privacy, security, user management, device management and information management), were not considered further. This in turn dictated that the stakeholder value also mainly related to the primary users. A further decision was taken to only cover Prototype 1 as otherwise the VIE table became somewhat unwieldy. However, separate VIE tables could have been drawn up for Prototypes 2 and 3.

Stakeholder value was assessed for each quality requirement by considering the likely benefits and then which stakeholders would obtain them. Then for each benefit, the type of value to the stakeholder was considered. The overall objectives for the primary users are to improve social inclusion, achieve greater independence and support wellbeing, so these formed the basis of the initial set of types of stakeholder value. Note this stakeholder value is the contribution to the next level up of the objectives. The final set of types of stakeholder value after considering all the quality requirements is as follows:

- Social inclusion
- Greater independence
- Wellbeing – self-confidence
- Wellbeing – peace on mind
- Wellbeing – safety
- Efficiency/time saving
- Cost
- Promotion of new ideas.

A ‘Y’ was placed in the table where there is considered to be stakeholder value. The stakeholder value is captured in this VIE table using several columns, as opposed to rows which were used in Case Study 2, as it allows easier assessment of the total amounts of the different types of stakeholder value.

Frequency of use is also captured as a separate column. Data is not available about the frequency of use associated with the functionality related to the quality requirements so various assumptions are made as follows:

- The primary user makes 10 mobile calls a day
- The primary user plans in advance a journey once a week
- The primary user sends 10 messages a day
- The secondary user checks on the primary user’s location 4 times a day
- The secondary user checks all is well with the primary user 4 times a day
- A user is setup once only
- The system breaks once every two weeks
- There are less than 5 incidents of severity B each week
- The number of third party developers expressing interest in the system will be about 10
- Satisfaction with the system shall be measured during prototype evaluation.

The impact of the Prototype 1 designs and functionality on the quality requirements was assessed: see Table 5.3.

Calculations were then carried out to assess if considering stakeholder value produced different results. To do this, the number of benefits/stakeholder value dimensions was totalled for each quality requirement, and then multiplied by the frequency of use.

At this point it became clear that only the first six quality requirements lent themselves to this calculation. The reasons for discarding the last four from the calculation are as follows:

- System reliability is a fundamental system property and will need to be high. But it is a complete unknown at present.
-

Table 5.3: VIE table for the Research Project Case Study for Prototype 1

STAKEHOLDERS								STAKEHOLDER VALUE											DESIGN SOLUTIONS AND FUNCTIONAL DELIVERABLES FOR PROTOTYPE 1 RELEASE														
																					P1	P1	P1	P1	P1	P1	P1	P1	P1	P1	P1	P1	P1
Down's Syndrome organizations	Researchers	Systems support	Tertiary users - friends	Tertiary Users - teachers	Secondary Users - carers (includes family)	Primary Users - Age 14+	**BENEFITS**	Functionality	Social inclusion	Greater independence	Wellbeing - self-confidence, motivation, self-worth, achievement	Wellbeing - peace of mind	Wellbeing - safety	Efficiency - time saving	Cost	Promotion of new ideas	**FREQUENCY OF USE**	**SYSTEM QUALITY REQUIREMENTS - 'HOW WELL' THE SYSTEM MUST PERFORM**	Use Tellu's SmartTracker Platform as Base	Use UniversAAL Platform as Base	Choice of H/W and S/W for Robustness D1	Support for time representation D7	Interface options of icons, symbols and animation D2	Interface options for the aesthetics - colour, font, contrast D3	Interface heuristics (Number of clicks) D4	Customizable interface options FUN9	Accurate outdoors location monitoring FUN27 / FUN14 / FUN5 / FUN15a	Support for outdoor navigation FUN6a	Carer is able to contact primary user FUN28 FUN2	Able to monitor emotional state by user request FUN2	Providing reminders FUN7	Support Context Awareness - time, battery, location, connectivity, volume of sound FUN14	System is reactive by default FUN8
				Y	Y	Y	able to contact	Y									Assume makes 10 calls a day	Mobile calls % of defined primary users able to make mobile/smartphone calls unaided															
						Y	use of technology		Y									Past [Age 10-18]: 28															
						Y	use of technology				Y							Past [Age 19 or over]: 63															
						Y	able to contact		Y									Past [Age 14 or over]: 50 <- Guess. Target [Age 14 or over]: 75			5	0	50	50	70	70	0	0	0	0	0	30	0
						Y	able to contact				Y																						
						Y	able to contact			Y																							
						Y	able to contact					Y																					
					Y		able to contact					Y																					
						Y	able to contact						Y																				
					Y		support organizing							Y																			
			Y		Y	Y	support travel	Y									Assume uses once a week	Local journeys % of primary users able to plan in advance local journeys unaided															
						Y	support activities		Y									Past [Age 10-18]: 0															
						Y	use of technology		Y									Past [Age 19 or over]: 7															
						Y	use of technology				Y							Past [Age 14 or over]: 5 <- Guess. Target [Age 14 or over]: >40			0	40	50	50	70	70	0	0	0	0	0	20	0
						Y	support travel			Y																							
						Y	support travel						Y																				
						Y	time management			Y																							
						Y	access to data			Y																							
					Y		support organizing							Y																			
			Y		Y	Y	able to contact	Y									Assume sends 10 messages a day	Messages % of primary users sending messages unaided (via use of apps)															
						Y	able to contact			Y								Past [Age 10-18, SMS/email]: 15/8															
						Y	able to contact				Y							Past [Age 19 or over, SMS/email]: 37/25															
						Y	able to contact		Y									Past [Age 14 or over]: 25 <- Guess. Target [Age 14 or over]: >50			5	0	50	50	70	70	0	0	0	20	20	20	0
						Y	use of technology		Y																								
						Y	use of technology				Y																						
						Y	support activities		Y																								
					Y		support organizing							Y																			
					Y		able to locate	Y									Assume used 4 times a day	Location % of time away from home that location can be reasonably established															
						Y	support travel						Y					Past [Aged 14 or over]: 0%															
						Y	support travel			Y								Target [Age 14 or over]: >80% <- Guess			20	0	0	0	0	20	100	0	10	0	0	20	0
					Y		able to locate					Y																					
					Y		able to locate							Y																			
				Y			reporting effort							Y																			
					Y		able to check	Y									If auto then continuous. Else on demand	Check all is well Time in minutes spent weekly by the secondary user monitoring the emotional state of the defined [primary user]															
						Y	faster reassurance					Y						Past [Aged 10-18]: 120 <- Guess															
						Y	able to contact			Y								Past [Aged 19 or over]: 90 <- Guess															
					Y		able to check					Y						Target [Age 14 or over]: <60 <- Guess			0	0	0	0	0	50	0	0	20	70	10	10	0
					Y		able to check							Y																			
																	Once for every system user	Setup time Average time in minutes to set up the technology for a primary user															
		Y					speed to setup							Y				Past [All users]: 60 minutes <- Rough Guess															
					Y		speed to setup							Y				Target [All users]: <30 <- Guess			0	0	N10	N10	20	60	0	0	0	0	0	N10	0
																	Ongoing - monitor monthly	System reliability NF12 (P2) MTBF in hours for system network (especially during Pilot 1 and 2). (Impact on end-users depends on the number of breaks.)															
		Y					maint. support effort							Y				Past [SmartTracker, June 2014]: ?															
		Y					maint. support effort								Y			Target [SmartTracker, June 2015]: 300 <- Guess			20	0	0	0	0	0	0	0	0	0	0	0	0
					Y		system available					Y																					
						Y	system available					Y																					
						Y	system available						Y																				
																	Ongoing - monitor weekly	Incidents Number of incidents over year by type by priority reported. (Impact depends on number of times that the end-users experience the problems, and of course their type and severity.)															
		Y					maint. support effort							Y				Past [SmartTracker, All, Priority B, June 2014]: ?															
		Y					maint. support effort								Y			Target [Smart Tracker, All, Priority B, June 2015]: <5 <- Guess			0	0	5	5	5	5	0	0	0	0	0	5	0
					Y		system works					Y																					
						Y	system works					Y																					
						Y	system works						Y																				
																	Measure monthly	Third party developers Number of third party developers participating in or following Poseidon on social media.															
Y							visibility									Y		Past [Poseidon, June 2014]: 0 <- Guess															
	Y						visibility									Y		Target [Poseidon, June 2015]: 10 <- Guess			5	5	10	10	10	20	5	5	5	5	10	20	0
						Y	potentially more apps		Y																								
																	Assume measured after each prototype	Satisfaction with the system % of secondary users observing a positive impact on a primary user from the use of Poseidon technology to support tasks															
							Overall system rating (Actually two levels higher in the objectives hierarchy)											Past [All primary users, Laptop/Tablet/smartphone use]: 82/85/56															
																		Target [Age 14 or over]: >50%			10	20	10	10	40	60	40	40	40	40	60	30	0
																		DEVELOPMENT COSTS															

- Incidents are also a complete unknown also. The impact of the incidents will depend on their type and severity and the frequency of hitting them.
- The interest and involvement of third party developers will be key, but its impact on value is less direct and less understood.
- System satisfaction is already a high-level system measure, which measures across the entire system.

However, for the first six quality requirements, the calculations held and the results were captured in a second VIE table, see Table 5.3. The next step for these six quality requirements was to assess the stakeholder value delivered by each design solution/functional deliverable. For example, Customisable interface options FUN9 is estimated to impact Messages delivering 70% of the required improvement. So 70% of the stakeholder value can be claimed. The total stakeholder value for Messages is 80, so 70% is 56. Once all the stakeholder value calculations had been completed, the columns were totalled to show the stakeholder value total for each of the design solutions/functional deliverables.

As Local journeys is assumed to be used only once a week as opposed to daily, its contribution was divided by 7. Also as Setup time is assumed to be carried out only once, it could be ignored in these calculations.

The rows and columns of Table 5.3 were rearranged to put the quality requirements and design solutions/functional deliverables in order of total stakeholder value. The quality totals and the stakeholder value totals could then be compared looking for any differences.

5.4.5 Findings

This section evaluates the results and what was found out during the case study against the research propositions.

Proposition 1: Current IT projects fail to capture value and priority information adequately in their system specifications.

- What prioritization methods have been used prior to the case study?

None.

- What system benefits and costs are identified in the original system documentation?

In terms of specific quantified objectives, there are five identified:

- 50% of secondary users should consider the product makes the primary users more independent and autonomous
- More than 50% of users should like to use the product
- The number of developers participating in project social media should be measured
- The number of companies providing services created by the project should be measured
- The number of apps/services created by the project and provided in different countries should be measured.

Quantified information was collected on the profiles of the primary users in terms of their current use of technology and also on what aspects/features were considered "very important" for the system. "Very important" aspects included: adaptable to individual needs (61%), robust (56%), use icons and symbols (34%) and use strong contrasts (19%). "Very helpful" functionality included: being able to determine a primary user had reached their destination (78%), locating a specific person with DS (77%) and setting an alert to do something (63%). This information concerns the required functionality though, and doesn't discuss the likely resulting system benefits.

Table 5.4: Second VIE table for Prototype 1 showing a subset with some extensions

STAKEHOLDERS								STAKEHOLDER VALUE											DESIGN SOLUTIONS AND FUNCTIONAL DELIVERABLES FOR PROTOTYPE 1 RELEASE														
																					P1	P1	P1	P1	P1	P1	P1	P1	P1	P1	P1	P1	P1
Down's Syndrome organizations	Researchers	Systems support	Tertiary users - friends	Tertiary Users - teachers	Secondary Users - carers (includes family)	Primary Users - Age 14+	BENEFITS	Functionality	Social inclusion	Greater independence	Wellbeing - self-confidence, motivation, self-worth, achievement	Wellbeing - peace of mind	Wellbeing - safety	Efficiency - time saving	Cost	Promotion of new ideas	FREQUENCY OF USE	SYSTEM QUALITY REQUIREMENTS - 'HOW WELL' THE SYSTEM MUST PERFORM	Use Tellu's SmartTracker Platform as Base	Use UniversAAL Platform as Base	Customizable interface options FUN9	Interface heuristics (Number of clicks) D4	Interface options of icons, symbols and animation D2	Interface options for the aesthetics - colour, font, contrast D3	Support Context Awareness - time, battery, location, connectivity, volume of sound FUN14	Accurate outdoors location monitoring FUN27 / FUN14 / FUN5 / FUN15a	Able to monitor emotional state by user request FUN2	Support for time representation D7	Providing reminders FUN7	Choice of H/W and S/W for Robustness D1	Carer is able to contact primary user FUN28 FUN2	Support for outdoor navigation FUN6a	System is reactive by default FUN8
				Y	Y	Y	able to contact	Y									Assume makes 10 calls a day	**Mobile calls** % of defined primary users able to make mobile/smartphone calls unaided															
						Y	use of technology		Y									Past [Age 10-18]: 28															
						Y	use of technology				Y						10 Ys for Value	Past [Age 19 or over]: 63															
						Y	able to contact		Y									Past [Age 14 or over]: 50 <- Guess. Target [Age 14 or over]: 75			70	70	50	50	30	0	0	0	0	5	0	0	0
						Y	able to contact				Y						10 x 10 = 100																
						Y	able to contact			Y																							
						Y	able to contact					Y						Adjusted for value			70	70	50	50	30	0	0	0	0	5	0	0	0
					Y		able to contact					Y																					
						Y	able to contact						Y																				
					Y		support organizing							Y																			
			Y		Y	Y	able to contact	Y									Assume sends 10 messages a day	**Messages** % of primary users sending messages unaided (via use of apps)															
						Y	able to contact			Y								Past [Age 10-18, SMS/email]: 15/8															
						Y	able to contact				Y						8 Ys for Value	Past [Age 19 or over, SMS/email]: 37/25															
						Y	able to contact		Y									Past [Age 14 or over]: 25 <- Guess. Target [Age 14 or over]: >50			70	70	50	50	20	0	20	0	20	5	0	0	0
						Y	use of technology		Y								8 x 10 = 80																
						Y	use of technology				Y																						
						Y	support activities		Y									Adjusted for value			56	56	40	40	16	0	16	0	16	4	0	0	0
					Y		support organizing							Y																			
					Y		able to locate	Y									Assume used 4 times a day	**Location** % of time away from home that location can be reasonably established															
						Y	support travel						Y					Past [Aged 14 or over]: 0%															
						Y	support travel			Y							6 Ys for Value	Target [Age 14 or over]: >80% <- Guess			20	0	0	0	20	100	0	0	0	20	10	0	0
					Y		able to locate					Y																					
					Y		able to locate							Y			6 x 4 = 24																
				Y			reporting effort							Y				Adjusted for value			5	0	0	0	5	24	0	0	0	5	2	0	0
					Y		able to check	Y									Assume used 4 times a day	**Check all is well** Time in minutes spent weekly by the secondary user monitoring the emotional state of the defined [primary user]															
						Y	faster reassurance					Y						Past [Aged 10-18]: 120 <- Guess															
						Y	able to contact			Y							5 Ys for Value	Past [Aged 19 or over]: 90 <- Guess															
					Y		able to check					Y						Target [Age 14 or over]: <60 <- Guess			50	0	0	0	10	0	70	0	10	0	20	0	0
					Y		able to check							Y			5 x 4 = 20																
																		Adjusted for value			10	0	0	0	2	0	14	0	2	0	4	0	0
			Y		Y	Y	support travel	Y									Assume uses once a week	**Local journeys** % of primary users able to plan in advance local journeys unaided															
						Y	support activities		Y									Past [Age 10-18]: 0															
						Y	use of technology		Y								9 Ys for Value	Past [Age 19 or over]: 7															
						Y	use of technology				Y							Past [Age 14 or over]: 5 <- Guess. Target [Age 14 or over]: >40			70	70	50	50	20	0	0	40	0	0	0	0	0
						Y	support travel			Y							9 x 1 = 9																
						Y	support travel						Y																				
						Y	time management			Y								Adjusted for value - note used only once a week			1	1	1	1	0	0	0	0	0	0	0	0	0
						Y	access to data			Y																							
					Y		support organizing							Y																			
		Y					speed to setup							Y			Once for every system user	**Setup time** Average time in minutes to set up the technology for a primary user															
					Y		speed to setup							Y				Past [All users]: 60 minutes <- Guess															
																	2 Ys	Target [All users]: <30 <- Guess			0	0	N10	N10	20	60	0	0	0	0	0	N10	0
																	2 x 1 = 2																
																		As used once only, ignore in the value calculations															
																		Total for Quality			280	210	140	140	120	160	90	40	30	30	30	N10	0
																		Total for Value			142	127	91	91	55	25	30	4	18	14	6	0	0

Costs are only identified at high level, there is no breakdown to lower-level functionality.

- *What stakeholder value is identified in the original system documentation?*

Discussion of the stakeholder value is only in high-level terms, such as social inclusion, adaptation to the individual, smart environments supporting people with DS, technical solutions to be portable across different technology, and provision of a development environment for the future. As this project is aiming to support people with DS, rather than say, improve a business process, it is understandable that stakeholder value is not quantified in such terms as time saved or cost reduction.

Proposition 2: The use of metrics is beneficial to understanding the quality requirements and for supporting IT priority decision-making.

- What quality requirements are identified prior to the case study?

System reliability, robustness, safety, usefulness and interoperability across different platforms are identified. However, there are no quantified levels discussed.

- What problems/benefits are introduced by the use of metrics?

There was no major issue over the use of metrics. The questionnaire about the system user requirements completed by the secondary users identified many of the required current (past) levels for the quality requirements. As discussed before, there was a slight problem because the age ranges mismatched – the questionnaire had captured data for age 10 to 18 and age 19 and over, but the project after discussion had decided age 14 or 15 and over was their target group for the system – so some adjustment was needed. Most of the goal levels were informed guesses based on the current level information.

There was no data regarding the system engineering attributes of the system, such as reliability, performance or security.

- How well are the quality requirements captured?

As discussed previously, the VIE table focused on the primary users' quality requirements, so quality requirements for the other stakeholders and for the system engineering aspects of the system were omitted. An initial 'top ten' set of quality requirements were selected that spanned the areas that effect the primary users' system experiences. The selection reflected the aspects raised in the system documentation.

Proposition 3: Identifying stakeholders more explicitly and considering the impacts on stakeholders of the requirements in terms of value and cost is useful for IT priority decision-making.

- How well are stakeholder viewpoints catered for?

The stakeholder viewpoints – allowing for the deliberate focus on the primary users – worked well. It was easy to establish which stakeholders a given quality requirement was going to impact. Mapping to the selected stakeholder value was also not difficult. The only issue was the time that it took and the discipline. This was an area where an application tool could help.

- Do stakeholders show different impacts for stakeholder value?

Yes – though this was damped down in this table due to the focus on the primary user.

- How well is stakeholder value captured, especially explicit stakeholder value?

Explicit stakeholder value was more difficult for this system given it was about improving accessibility. The value covered aspects such as social inclusion, peace of mind and safety. The decision taken for this VIE table was to simply count up the number of benefits that could be identified. Further refinement of saying that one benefit counted more than another would be possible, but here all the benefits were treated as equal.

There were certain areas of stakeholder value where the benefits could have been translated into financial values. For example, the teachers checking the locations of the primary users would take less time and effort once this system was implemented. However, the social benefits outweigh such considerations.

- Is it useful to identify stakeholder cost (as well as value)?

This was not carried out for this case study. Investigating the operational and support costs for running the system would be of interest. Given the vulnerable nature of the primary users to system problems, more thought is needed in the area of support. Some of the issues can be tackled by building instructions (additional functionality) into the handsets.

Proposition 4: The additional effort spent structuring the data using the proposed method helps identify and understand the system data and supports IT priority decision-making.

- How well are the functional requirements captured?

The VIE table used the design solutions/functional deliverables as expressed by the project. This reflected the weaknesses in the initial requirements specification of the system. Given the phased delivery approach, more detailed functional specification would have been beneficial. From the viewpoint of the VIE table, delivery of functionality is driven by the improvements in the quality requirements and by the delivery of the design solutions. Having a detailed list of the functional requirements would be helpful to a project, and crosschecking the deliverables against it would be a useful exercise. The focus of the VIE table is on identifying which designs (with their associated functionality) are going to be most effective in delivering the requirements. The VIE table is not about checking functional completeness as such. Having said that, the table will identify if there are quality requirements that are not addressed or if there are designs that seem not to be delivering required quality improvements. For example, the functional deliverable, *Support for outdoor navigation (FUN 6a)* is seen not to deliver to the initial set of quality requirements. Either it is excess functionality for Prototype 1 or there is some missing quality requirement, and that would be something for a system designer to consider. It maybe preparatory functionality for Prototype 2, but its delivery in the first instance is seen as not crucial for Prototype 1.

- *Can the designs be mapped to the increments?*

The design solutions/functional deliverables were initially allocated to the prototypes using the preferences expressed in the documentation. The overall management structure of the project did not lend itself to more agile delivery. However, as shown by the VIE table constructed, it would be possible to conduct the project in an evolutionary or agile manner. For example, from the VIE table the highest priority designs involve developing the primary user interface and that could be worked on early and separately.

- *Can the system development costs be captured?*

The system development costs were only available at high-level, they were not broken down to delivering the functionality.

- *Is return on investment (ROI) captured and how is it calculated?*

Given the lack of cost data, the calculations of priority lacked the benefit-to-cost ratios. If the data had been available, the calculations could have been carried out.

- *How well are the varying levels of abstraction of the requirements handled?*

One of the quality requirements, *Satisfaction with the system* was determined during processing to be a high level objective: all the other quality requirements contributed towards achieving it. Only when the stakeholder value considerations were added, did this problem emerge. The main issue is the aspect of double counting.

- How well are the requirements dependencies captured?

The quality requirement, *Satisfaction with the system,* due to being at a higher level of abstraction, has dependency relationships with the other lower-level quality requirements. Handling this quality requirement in a separate VIE table for the higher-level objectives would be the solution. For the other quality requirements, there was no issue over dependencies.

- How well are the design dependencies captured?

For Prototype 1, there were no design dependencies.

- Does the proposed method scale up to handle numerous requirements?

The method captured the 'top ten' quality requirements as an initial set to aim at meeting. Additional quality requirements could be added as needed. The other means used to tackle scalability was to set the scope as Prototype 1 only – this reduced the number of design solutions/functional deliverables that needed to be handled.

- How well does the proposed method handle reprioritization?

Adding in additional requirements or designs would mean determining the impact data for the new items and then recalculating the totals. There would be no need to revisit the impact data for the older items.

- How well does the proposed method cater for capturing the actual impacts after implementation?

Either the estimated values are replaced with the actual values or extra columns are added to the VIE table. The latter would be preferred if wanting to reflect on the accuracy of estimation, but the former would give a simpler presentation.

- Is the proposed method overly complex?

The proposed method only asks for information that should be taken into account, so from that perspective, it is not overly complex. However, handling stakeholder value in this case study was time-consuming. To a certain extent, this problem was due to the fact that this was the first time that stakeholder value had been approached in this way. If the stakeholder value framework was better established, and especially if supported by an application tool, then the process would become easier.

Proposition 5: If stakeholders are provided with a functioning IT prioritization method, then a better understanding of the planned project outcomes is the result.

- Does a better understanding of the system result?

Definitely by going through this process, areas where data is lacking are identified. For example, the lack of detailed costs became apparent.

- Does the method produce useful results?

Yes. The priority order for consideration of the design solutions/functional deliverables was identified, and highlighted the primary user interface as being of highest importance. The cost data was lacking, but from knowledge of the overall system, this feels the right outcome. The table also identified issues in the area of the functionality to support outdoor navigation, as it was not contributing to the goals for Prototype 1.

5.4.6 Main findings of Case Study 3

The following findings can be summarized.

For the original Research Project system specification, there are weaknesses in the requirements specification. The project commenced without a detailed requirements specification and while the work on the user requirements definitely helped, there are still gaps in the system specification, which are likely to impact the project planning. While the research activities are elastic given their nature, delivery of the working functionality is not, and the project resources are limited.

Regarding the original system specification:
(Note, explanation of the cross-referencing to the Findings given below is explained in the next chapter.)

- No prioritization method had been used prior to the case study (Finding 27).
- There was a lack of sufficient metrics data (Finding 4).
- There was a lack of detailed discussion of the expected stakeholder value (Finding 1).

- A greater focus on the project's deliverables would be helpful to ensure successful delivery of stakeholder value in the field (Findings 11 and 2)
- There was insufficient consideration of operational aspects such as support and system availability (Finding 3).

The creation of the VIE table identified several issues and observations:

- Extracting information from the user questionnaires into the quality metrics was helpful to understanding the levels of ambition of the project (Finding 28).
- To enable a VIE table to be created for the Research Project, a simple mechanism for handing stakeholder value covering less tangible dimensions had to be devised (Finding 21)
- Addition prioritization factors are identified in the area of social inclusion (Finding 22).
- The 'top ten' quality requirements were used. This coped with scaling-up issues (Finding 26).
- Specifying the functions in detail was not required to create the VIE table (Finding 12).
- An issue of mixing quality requirements at different levels of abstraction is identified, which occurs when investigating stakeholder value in more depth. *Satisfaction with the system* is at a higher level of abstraction (Finding 13).
- An issue regarding stakeholder value was identified in that some quality requirements lend themselves to workload volume or frequency of use calculations to determine the stakeholder value. Others, such as system reliability, do not. For system reliability, the impact on the primary stakeholder was calculated and found to be negligible. Further research is needed on this aspect (Finding 23).
- Two functional deliverables were identified within Prototype 1 that had no significant impact on the 'top ten' quality requirements. This needs resolving, either the functionality should be removed or maybe it reflects unspecified quality requirements (Finding 25).

The following concepts were validated to the extent possible for a single case study:

- Using VIE helps capture the key components of the system and shows where there are gaps in the data (Finding 10).
- The stakeholders had differing viewpoints and differing stakeholder value, and the quality requirements had differing stakeholder value (Findings 6-8).
- Using the stakeholder value information was shown to produce different results than when using the quality requirements alone. Value-to-cost ratios differed from quality-to-cost ratios (Finding 24).
- Using the method placed a focus on the project deliverables that was lacking in the existing system documentation (Finding 2).

Evaluation of this case study by a participant could not be carried out. The project manager did volunteer by email the information that he considered the 'top ten' objectives "are very relevant" and that he liked the specification of stakeholder value. Further, he found the work carried out on the case study was a "great support for the project" and "valuable". See Appendix B.

Chapter 6

Aggregate Findings

6.1 Introduction

In the previous chapter, three individual case studies were described and the findings of each case study presented. This chapter presents and discusses the aggregate findings across the three case studies. By comparing the findings across the three case studies, it will be possible to separate findings that are more specific from those that are of a more general nature. On this basis, generalizations can be made as to which propositions were supported by the case studies and which ones were not.

6.2 Aggregate findings

Below the findings related to each proposition are discussed. The applicability of each finding is considered across all three case studies to determine, if the finding supports the proposition. It is then determined whether each proposition is supported by its findings or not. For this purpose, the findings from the case studies are listed in Figure 6.1, Findings from the case studies, as they will be referred to in the following discussion.

Proposition 1: Current IT projects fail to capture value and priority information adequately in their system specifications

In all three case studies, there are issues with the justifications for IT investment in terms of identification of value and subsequent prioritization. Boehm's observation (Boehm, 2003) that software engineering is too often conducted in a value-free basis holds also true for the three case studies. In the Bank Loan System specification, there is no discussion of the business benefits. For the IT Services Bid specification, the business benefits are outlined in the executive summary, but there is no assessment of their value to the organization. For the Research Project, value is more difficult to assess given it is more connected to social value than financial value, but there ought to have been more discussion of the social value and there are associated effort savings that could have been identified and investigated in more depth. Certainly in all three cases, more value data was available, if it had been requested (**Finding 1**).

Associated with the lack of value data is an observation that all of the three system specifications lacked in some way a focus on delivering value (Finding 2): not only is there a lack of detail regarding the benefits to be obtained, but there is also insufficient attention to the time scale when these benefits will be delivered. For the Bank Loan System, the approach is implicitly waterfall: there is no discussion of incremental or phased delivery (The case study revealed that phased value delivery was possible and helped reduce risk). For the IT Services Bid, there was no discussion of any specific timetable for implementing changes, such as upgrading the legacy products, even though achieving the main objective of cost reduction depends to some extent on this occurring in order to reduce the number of service calls. The Research Project does have a plan to carry out its implementation over three prototypes. However, no allocation of requirements or designs to the different prototype stages is made in the original specification. This lack of focus on value delivery risks the delivery of any value. Boehm considers that value delivery should be tracked (2006). When he attacks the earned value method (EVM), he says that while EVM tracks a project's progress to meeting its plan, "it has absolutely nothing to say about the actual value being earned for the organization by the project's results" and he states "it would be preferable to have techniques which support monitoring and control of the actual value to be earned by the project's results". Further Boehm and Jain (2006) state that this is even more important "in an era of increasing rates of

List of Findings from the Case Studies

Finding 1: Value data, including justification for IT investment is failing to be captured.
Finding 2: There is a lack of focus on value delivery.
Finding 3: There is a lack of consideration of the operational use of the system.
Finding 4: Because of insufficient data in the system specification, extra effort has to be spent to find information so that the metrics for the quality requirements can be sufficiently specified.
Finding 5: Business process models (or functions at appropriate level of abstraction) help to identify quality requirements.
Finding 6: The stakeholders show differing viewpoints.
Finding 7: The stakeholders show differing stakeholder value.
Finding 8: The quality requirements are shown to have differing stakeholder value.
Finding 9: (Observation only) Stakeholder costs could be allocated against stakeholders in a similar manner to the way that stakeholder value is allocated.
Finding 10: Use of the VIE method helps identify, organize and clarify the system data.
Finding 11: Use of the VIE table helps identify where implementation plans are missing.
Finding 12: Effective initial system planning can be carried out using the key functions. While specifying the functionality in detail provides useful reassurance, it is not essential. See also 26.
Finding 13: Care is needed with high-level quality requirements that double counting is avoided.
Finding 14: Displaying design and/or increment interdependencies in a VIE table needs further work as at present only works for small systems.
Finding 15: The VIE table needs to capture workload volumes/frequency of use to help with stakeholder value calculations.
Finding 16: The different dimensions of stakeholder value should be captured separately in a VIE table.
Finding 17: Calculation of stakeholder value using the prioritization factors identified from the literature as the dimensions of value works well.
Finding 18: Extra effort is required because system data is often embedded in current system specifications.
Finding 19: Stakeholder value requires at least three columns in a VIE table: one for the value dimension, and for each stakeholder, one for workload volume/frequency of use and one for the calculated value.
Finding 20: Observation - There can be a problem with scaling up the number of functions being handled in a VIE table.
Finding 21: A simple technique for measurement of stakeholder value for social value is outlined.
Finding 22: Additional prioritization factors in the area of social values can be identified.
Finding 23: Care needed over stakeholder value comparisons if only some quality requirements have frequency of use data.
Finding 24: Value-to-cost ratios do not give the same results as quality-to-cost ratios.
Finding 25: An IE table (and hence a VIE table) identifies non-impacting functional deliverable(s).
Finding 26: Using a 'top ten' set of quality requirements can be a useful technique. It addresses scaling-up. See also 12.
Finding 27: Prioritization methods are not used.
Finding 28: Specifying the quality requirements as metrics clarifies the levels of ambition.
Finding 29: Participants achieve a better understanding of the planned project results.
Finding 30: The VIE table is seen as useful for planning projects.

Figure 6.1: Findings from the case studies

change in market, technology, organizational, and environmental conditions" and that projects need to manage "adaptively towards evolving value realization", else they are likely to become non-competitive.

Similarly to the collection of value data, prior to the case study, no prioritization method was used in any of the projects (**Finding 27**). This lack of use of a prioritization method agrees with Lehtola (2006), who reports the prioritization methods "are not widely used". The issue is that lack of the use of a prioritization method implies that prioritization data is not considered (that is, not specifically collected and not assessed) and there is then no derived priority order for implementation (no derived explicit priorities or prioritization data supporting priority decision-making). At best, there will be informal, non-aggregated stakeholder viewpoints on priorities based on implicit ad hoc evaluation and intuition.

Proposition 2: The use of metrics is beneficial to understanding the quality requirements and for supporting IT priority decision-making

In all three case studies, specifying the quality requirements as metrics clarified the levels of ambition (**Finding 28**) or raised further questions about the estimated levels.

The practicality of the use of metrics is discussed in (Hubbard, 2007). Further, the requirement for having some kind of method to utilize the metrics is also recognised: "In practice, an approach to measurement that explicitly links high-level business goals and measurement data is needed." (Basili et al., 2009)

Proposition 3: Identifying stakeholders more explicitly and considering the impacts on stakeholders of the requirements in terms of value and cost is useful for IT priority decision-making

For all three case studies, the stakeholders show differing viewpoints (**Finding 6**) and differing stakeholder value (**Finding 7**). In turn, the quality requirements are found to have different stakeholder values (**Finding 8**). Given the quality requirements are of differing stakeholder value, this reflects in the VIE table calculations for value-to-cost ratios. In contrast, the IE method handles all quality requirements as of equal value in its quality-to-cost ratio calculations. The three findings, Findings 6 to 8, validate as proof of concept the addition of stakeholders and stakeholder value to create the VIE method and the method's calculation of value to cost ratios. However, at this stage, it is too great a claim to say that VIE completely solves the issue. The addition of the value information in the VIE method goes some way to correctly handling the varying importance of the quality requirements regarding value, but more research and fuller treatment of stakeholder value is needed before a claim to have corrected the situation can be made.

A VIE table captures cost in the same way as an IE table, but also adds the possibility of allocating the costs against stakeholders (**Finding 9**). To date the stakeholder costs have not been assessed, the possibility of their use has just been noted (in the Bank Loan case study). The idea of capturing both the allocation of costs and benefits received appears a useful idea. The VIE method is capable of utilizing such data by capturing such costs alongside stakeholder benefits (that is, as treating costs as additional dimensions of value: cost is already identified as a prioritization factor.) Certainly the present value-neutral approach of asking stakeholders for their requirements without capturing adequately the associated benefits and costs needs challenging, and a method such as VIE that utilizes the data would assist this.

However, in the case studies the opportunity to investigate such cost considerations was not present due to the fact that the cost data was either too sensitive (as in the Bank Loan System and the IT Services Bid) or not available (as in the Research Project).

Proposition 4: The additional effort spent structuring the data using the proposed method helps identify and understand the system data and supports IT priority decision-making

Extracting the data from the system specifications in order to create the VIE tables, in all three case studies led to a better understanding of the system data (**Finding 10**). Given the way that information is captured in the original system specifications, substantial work was required at times. This was true for all three case studies, especially for the IT Services Bid and the Research Project, which are larger systems with considerable system specification documentation. The required information is often embedded (**Finding 18**). For example, some functionality is captured in a scenario in the case of the Research Project.

Missing data: Use of the VIE method identifies where key system data is lacking. The documentation for all three case studies proves inadequate in supplying the necessary information to specify all the metrics for the quality requirements (**Finding 4**):

- For the Bank Loan case study, further investigation of the business processes provided the quality requirements, and the past and goal information was then obtained from the consultants
- For the IT Services Bid case study, the SLA-related metrics were present in the supplied documentation. There was also the overarching, cost reduction objective and mention of the need for measuring customer satisfaction by interviews and surveys. However, the other metrics needed to be derived from considering how the solution components could be measured. The current (past) levels were completely unknown – possibly as there was limited access to the retail customer's data. So for these additional metrics, both the current (past) and target (goal) levels were guesses, even though some had a key contribution to make towards achieving the required cost reduction
- For the Research Project, there is data collected during the requirements specification providing past levels for the primary users, and the goal levels have to be estimated from those levels. However, for certain key areas such as system reliability, there is no current (past) or target (goal) data available. The quality requirements were derived by considering the features and functionality that the secondary users considered "very important" or "very helpful" and that would make a significant impact on the lives of people with DS and their families.

The lack of information to support metrics is not surprising given the majority of existing systems development methods fail to utilize quantitative data, and many system developers fail to quantify quality requirements (Gilb, 2005a). Using the VIE method makes the gaps in the metrics documentation become clearly visible.

However, the onus is on the person developing the VIE table to select the appropriate quality requirements.

Missing implementation plans: In addition, the implementation plans are missing (**Finding 11**). This links with Finding 2, the lack of focus on value delivery and with Finding 3, the lack of consideration of operational use. The proposed design solutions setting out how the systems are to be implemented fail to address how the system is to be delivered. For the Bank Loan System, only the software components are discussed. For the IT Services Bid, there is discussion of the service desk, service management monitoring and control, and also transition plans, but there is no discussion of the plans for replacing the legacy products. For the Research Project, it is identified that there would be three prototypes, but minimal discussion of their content. In addition, there is no discussion of the transition into use in the field.

A VIE table requires what can be considered fundamental system data addressing the stakeholders, the requirements, the designs, the costs and the stakeholder value, so any such data found missing is a concern. There is incomplete data in all three case studies: the missing data includes justification for IT investment (Finding 1), stakeholder value (also Finding 1), metrics for the quality requirements (Finding 4), implementation plans (Finding 11) and consideration of operational use (Finding 3).

Interdependencies: Requirement interdependencies are identified in two of the **three** case studies. In the Bank Loan System, in order to prevent double counting, care was taken to avoid any interdependencies. To achieve this, a business process model was drawn up and this helped ensure the separation of the quality requirements. For the IT Services Bid, a functional hierarchy was drawn up and quality requirements were specified with regard to that. However, at a later stage when considering the stakeholder value, the *Satisfaction with the system* quality requirement was found to overlap with other quality requirements (**Finding 13**). For the Research Project, the *Satisfaction with the system* quality requirement was found to behave in a similar way. The issue is that customer satisfaction is a complex quality requirement that aggregates various different dimensions of value as decided by the customer. It ideally needs to be handled in a higher strategic-level IE table, though with care as seen in the two case studies, it can be used in the lower level VIE tables.

Designs making no impacts: For the Research Project case study, two functional deliverables were identified, *FUN6a* and *FUN8* that made no impact on any of the specified quality requirements (**Finding 25**). This is actually a feature arising from the IE method that it helps identify such designs. Either the design is not required or there is at least one missing quality requirement. The other two case studies did not have this issue.

Display of stakeholder value: Over the course of the three case studies, how stakeholder value was displayed was modified. In the Bank Loan case study it was recognized that aggregating the stakeholder value for a quality requirement to a single figure was losing information and so it was decided to try to capture the different dimensions of stakeholder value separately (**Finding 16**). It was also identified that the workload volumes or frequency of use needed to be captured in separate columns (**Finding 15**) to make the data explicit and to use it in the different stakeholders' stakeholder value calculations. For the IT Services Bid case study, the stakeholder value was captured using at least three columns: the stakeholder value dimension (also known as prioritization factor), and then for each stakeholder, the workload volume/frequency of use and the calculated stakeholder value (**Finding 19**).

For the Research Project, a further adaptation to how the stakeholder value data is captured was made: stakeholder value is captured as numerous columns, one for workload volume/frequency of use, one for benefits, and numerous columns for the different dimensions of stakeholder value, see Table 5.3. It was found difficult to assign numerical value to the stakeholder value, so a simple 'Y' for 'Yes, there is value' was placed in the relevance cells. To put a value on the stakeholder value for a quality requirement, the number of 'Y's for the quality requirement were added up and multiplied by the frequency of use. This provided a simple relative measure for the quality requirements of stakeholder value (**Finding 21**).

One issue was identified with using the workload volumes/frequency of use to multiply the 'Y's. Care has to be taken that a valid comparison is being made. Some quality requirements were of a different nature and thought is needed (**Finding 23**). For example, one quality requirement was to do with sending messages and that clearly would have an average frequency of use, while another system reliability was an on-going measure (MTBF). However, by converting the system reliability to represent the number of times that a system break was likely to occur in a given time period, a comparison could be made. More research is needed to work through different types of quality requirements.

Stakeholder value and prioritization factors: In the Bank Loan case study, two prioritization factors were used for stakeholder value, financial gain through additional sales of loan products and financial savings resulting from the effort savings of staff time.

For the IT Services Bid, several different dimensions of stakeholder value were found to exist. There is an extremely good mapping from the VIE table's dimensions of stakeholder value to the prioritization factors from the literature – almost one-to-one across all the prioritization factors. That there is so good a match was a surprise finding. It was only as the VIE table was completed that the mapping became apparent and the finding emerged: it had not been anticipated, but was most welcome as it verified the list of prioritization factors derived from the literature and was proof of concept of their relationship to stakeholder value (**Finding 17**).

For the Research Project, the stakeholder value was different because it involved social values, such as social inclusion. A list of different dimensions of stakeholder value for the social values was drawn up and this was found to work within the VIE table (**Finding 22**).

Calculation of value to cost ratios: A final finding regarding stakeholder value is that the value-to-cost ratios were found to differ from quality-to-cost ratios (**Finding 24**) for two of the case studies. The other case study, the IT Services Bid did not have sufficient data to allow such calculations. The fact that the ratios differ is proof of concept that using stakeholder value is necessary.

Proposition 5: If stakeholders are provided with a functioning IT prioritization method, then a better understanding of the planned project outcomes is the result.

Creating the VIE tables did help provide a better understanding of the requirements and the design solutions, and of the proposed implementation plans. It enabled gaps in the system documentation and likely shortfalls in delivering to the requirements to be detected.

Especially important from the viewpoint of this research, the VIE method enabled prioritization questions to be addressed. For both the Bank Loan System and the Research Project case studies, the stakeholder value contributions of the quality requirements and the design solutions are assessed in their relevant VIE tables (**Finding 29**). As has been discussed previously, only lack of data prevented this being the case for the IT services case study.

Further the VIE method was evaluated by the participants in the Bank Loan and the IT Services Bid projects (Case Studies 1 and 2) as being useful for planning projects (**Finding 30**). Their views were captured by questionnaire: see Appendix B. The third case study, the Research Project was not evaluated. However, the project manager volunteered by email that he found the information from the case study useful and that he liked the capture of the stakeholder value: see Appendix B.

Additional Findings: There were also some additional findings identified that did not fit under any of the propositions.

Underlying issues leading to a lack of focus on delivery: Underlying the lack of focus on value delivery is also a lack of investigation of the operational use of the system (**Finding 3**). Estimating the stakeholder value to be obtained requires information about the operational use of the system. For the Bank Loan System, there is a lack of information, including the number of loan requests likely to be processed, staff numbers and the frequency of updating the rules. Changes in working practices are involved for the back office staff and also for the staff amending the decision rules, and these need drafting and working on. The IT Services Bid is considerably better in documenting the operational information, which is to be expected as there is an on-going commitment over the service delivery. However, there is insufficient attention paid to understanding the rate of hardware refresh and the rationalization of legacy products, and their impact on the operational use. For the Research Project, there are several aspects such as provision of support and service availability that require detailed consideration.

Support for quality requirements and their metrics: To identify the quality requirements and their metrics drawing up a business process model was found helpful. It was also observed that a diagram of the main functions for a system could provide support in much the same way as a business process model (**Finding 5**).

A business process model was drawn up for the Bank Loan System (Figure 5.2) and it proved extremely useful. In addition, the diagrams providing an overview of the functionality were used (Figures 5.1, 5.5 and 5.6). In all three case studies, these were seen as helping ensure adequate coverage of the system by the quality requirements.

Scaling functionality: Functionality in the system documentation is captured at a fairly high level of abstraction. For the Bank Loan System, the business processes were drawn up as part of the case study, which made the functionality more explicit. For the IT Services Bid and for the Research Project, the functionality is scattered across the documentation. An exercise to list all the functionality was carried out in both these case studies. For the IT Services Bid, it was because the bid manager requested the exercise be carried out. Identifying the functions did help to understand the systems' scope. It allowed the researcher to better assess the VIE tables in relation to the system functionality and to check that no important aspects of the systems had been overlooked. However, see later, the researcher observed that the very detailed functionality contributed little to creating the VIE table. The bid manager subsequently placed each of the functional requirements associated with a quality requirement in the requirements column, and he labeled them as 'Must Have' as the functionality was not optional. (This is proof of concept for the earlier discussion in Chapter 3 about the relationship of quality attributes and functions.)

The number of functions present in the IT Services Bid documentation was very large (over 100 functions), in fact so large that once this was realized, the decision was taken to only use the top-level functions as the VIE table could not scale to contain them all (**Finding 20**). For the Research Project, the user functionality was drawn up to help identify the user requirements. It was found helpful in splitting the functionality among the three prototypes. The case study focused on the design solutions for Prototype 1.

Limits to the scalability of the design solutions: With regard to mapping the designs, there was no problem in allocating designs to increments. For the Bank Loan System with four increments displaying the alternative implementation paths using the VIE table did work: the increment interdependencies could be shown using black arrows. However, this is on the limit of usability, it is not going to scale up for larger, more complex systems with a greater number of interdependencies (**Finding**

14). It was observed that for the IT Services Bid and the Research Project, there were no design or increment interdependencies identified. This is considered due to the designs being at a high-level of abstraction.

Select the 'top ten' quality requirements: The Research Project successfully used the technique of identifying the 'top ten' quality requirements from the system documentation (**Finding 26**).

Identifying the higher level functional requirements only works: As mentioned above, it was found that documenting all the functionality in detail was not essential for initial system planning. In fact, the VIE tables were created using only the key quality requirements (derived with reference to the functionality) and the proposed designs. In all three case studies, this was found to work and indicates that effective system planning can be carried out without needing all the lower level functions being identified (**Finding 12**). (Note Finding 26 applies to quality requirements and Finding 12 to the functional requirements.)

6.3 Main findings

In many respects, the use of the VIE method in the case studies helped address many of the system issues identified. The findings 1, 6, 7, 8, 9, 13, 15, 16, 17, 19, 21, 22, 23 and 24 indicate that the VIE method has made a contribution towards an improved prioritization of IT by focusing on stakeholder value. The two tables below summarize the findings. The first table maps the findings to the system concepts and the case studies, which support the findings. The second table maps the propositions and findings to the case studies, which support them.

Table 6.1: Mapping of the findings to the system concepts and case studies

Coverage of the System Concept	**Finding**	**Case Study**		
		1: Bank Loan	**2: IT Services Bid**	**3: Research Project**
Objectives/quality requirements	8	Y	Y	Y
	13	n/a	Y	Y
	26	Y	Y	Y
Quality metrics	4	Y	Y	Y
	28	Y	Y	Y
Functionality	20	n/a	O	n/a
Functional requirements	25	n/a	n/a	Y
	12	Y	Y	Y
Designs/increments	14	O	n/a	n/a
Implementation plans	11	Y	Y	Y
System results – operational use	3	Y	Y	Y
	29	Y	missing data	Y
	2	Y	Y	Y
Stakeholder viewpoints	6	Y	Y	Y
Stakeholder value	1	Y	Y	Y
	7	Y	Y	Y
	15	Y	Y	Y
	16	Y	Y	Y
	19	Y	Y	Y
	21	n/a	n/a	Y
	23	n/a	n/a	Y
Stakeholder cost	9	O	a	a
Prioritization factors	17	Y	Y	missing data
	22	n/a	n/a	Y
Quality-to-cost ratios versus value-to-cost ratios (measures of IT investment potential, similar to ROI)	24	Y	missing data	Y
Practical guidance: Use of BPM or functions at appropriate abstraction level to identify quality requirements	5	Y	Y	Y
Practical guidance: Effort is required to extract the system data as often it is embedded in specifications	18	Y	Y	Y
Practical guidance: Use of the VIE method helps identify, organize and clarify the system data	10	Y	Y	Y
Prioritization methods are not used	27	Y	Y	Y
Participants find the VIE method useful	30	Y	Y	Not evaluated
Key: Y = yes n/a = not applicable a = applicable, but not considered O = observation only				

Table 6.2: Support for the propositions

Propositions	**Supported by Finding No.**	**Supported by Case Study No.**	**Comments**
Proposition 1: Current IT projects fail to capture value and priority information adequately in their system specifications	**1** Value data, including justification for IT investment is failing to be captured.	**1, 2, 3**	
	2 There is a lack of focus on value delivery	**1, 2, 3**	
	27 Prioritization methods are not used	**1, 2, 3**	
Proposition 2: The use of metrics is beneficial to understanding the quality requirements and for supporting IT priority decision-making	**28** Specifying the quality requirements as metrics clarified the levels of ambition	**1, 2, 3**	
Proposition 3: Identifying stakeholders more explicitly and considering the impacts on stakeholders of the requirements in terms of value and cost is useful for IT priority decision-making	**6** The stakeholders show differing viewpoints	**1, 2, 3**	
	7 The stakeholders show differing stakeholder value	**1, 2, 3**	
	8 The quality requirements are shown to have differing stakeholder value	**1, 2, 3**	
	9 Stakeholder costs could be allocated against stakeholders in a similar manner to the way that stakeholder value is allocated	**1**	**Observation only made during CS 1**
Proposition 4: The additional effort spent structuring the data using the proposed method helps identify and understand the system data and supports IT priority decision-making	**4** Because of insufficient data in the system specification extra effort has to be spent to find this information so that the metrics for the quality requirements can be sufficiently specified.	**1, 2, 3**	
	10 Use of the VIE method helps identify, organize and clarify the system data	**1, 2, 3**	
	11 Use of the VIE table helps identify where implementation plans are missing	**1, 2, 3**	

	13 Care is needed with high-level quality requirements so that double counting is avoided.	**2, 3**	Not applicable to CS 1 because it was controlled
	15 The VIE table needs to capture workload volumes/frequency of use to help with stakeholder value calculations.	**1, 2, 3**	
	16 The different dimensions of stakeholder value should be captured separately in the VIE table (rather than being combined together into a single value as in Case Study1).	**1, 2, 3**	
	17 Calculation of stakeholder value using the prioritization factors identified from the literature as the dimensions of value works well.	**1, 2**	**Missing data for CS 3**
	18 Effort required because system data is often embedded within system specifications	**1, 2, 3**	
	19 Stakeholder value requires at least three columns in a VIE table: one for the value dimension, one for the associated value and one for workload volume/frequency of use	**1, 2, 3**	**Identified during 1**
	21 A simple technique for measurement of stakeholder value for social value is outlined	**3**	Special case, not applicable to CS 1 and 2
	22 Additional prioritization factors in the area of social values can be identified	**3**	Special case, not applicable to CS 1 and 2
	23 Care needed over stakeholder value comparisons if only some quality requirements have frequency of use data	**3**	Special case, not applicable to CS 1 and 2
	24 Value-to-cost ratios do not give the same results as quality-to-cost ratios	**1, 3**	**Missing data for CS 2**
	25 An IE table (and hence a VIE table) identifies some non-impacting functional deliverable(s)	**3**	Not applicable to CS 1 and 2

Proposition 5: If stakeholders are provided with a functioning IT prioritization method, then a better understanding of the planned project outcomes is the result.	**29** Participants achieve a better understanding of the planned project results	**1, 2**	**Not evaluated for CS 3**
	30The VIE table method is seen as useful for planning projects (Evaluation by Questionnaire)	**1, 2**	**Not evaluated for CS 3**
Findings not related to any proposition	**3** There is a lack of consideration of the operational use of the system	**1, 2, 3**	
	5 Business process models (BPM) or functions at appropriate abstraction level help with identifying quality requirements	**1, 2, 3**	**BPM for 1 only. Functional diagrams for all**
	20 There can be a problem with scaling up the number of functions being handled in a VIE table.	**Observation only made during 2**	Scalability issue
	14 Displaying design and/or increment interdependencies in a VIE table needs further work as at present only works for small systems	**Observation made only during 1**	Scalability issue
	26 Using a 'top ten' set of quality requirements can be a useful technique. It is one way of addressing scaling-up issues. See also 12	**1, 2, 3**	Scalability solution
	12 Effective initial system planning can be carried out using the higher-level functions. See also 26	**1, 2, 3**	Scalability solution

As summarized in the table above, all propositions have been supported by the findings in the case studies. However, the degree of support for the propositions varies.

Proposition 1: Current IT projects fail to capture value and priority information adequately in their system specifications

There is strong evidence across all three case studies that current projects indeed **fail to collect stakeholder value data.** All three case studies show that there is very little if any focus on stakeholder value. Similarly, prioritization methods are not used. If prioritization is carried out, it is on an ad hoc and informal basis, which means that the criteria for the decisions remain implicit and the prioritization, which is taking place, is rather arbitrary.

Proposition 2: The use of metrics is beneficial to understanding the quality requirements and for supporting IT priority decision-making

This proposition about the benefits of using metrics is a rather fundamental one. Finding 28 shows how the **use of metrics is beneficial to understanding** the quality requirements by clarifying the level of ambition, which in turn enables and supports management in estimating concrete dimensions of stakeholder value. Once this has happened stakeholder value should then be used as the basis for any decisions regarding IT prioritization.

Proposition 3: Identifying stakeholders more explicitly and considering the impacts on stakeholders of the requirements in terms of value and cost is useful for IT priority decision-making

Proposition 3 is based on the fact that each IT system has **multiple stakeholders, who have different viewpoints** (Finding 6) with different expectations and different needs leading to different quality requirements. Each quality requirement delivers a specific stakeholder value to each stakeholder (Finding 8). Therefore, each stakeholder potentially receives different stakeholder value from the system (Finding 7).

Finding 9 expresses the fact that development planning could include more information about costs in relation to the benefits received by different stakeholders, which in turn could provide the basis for distributing the costs over the stakeholders receiving these benefits. This issue was discovered in Case Study 1, where all stakeholders received benefits, except for the system developers. It was observed that the system developers incurred most of the costs. This finding spurred a number of discussions and ideas. However, this issue could not be followed up (as costs were a sensitive issue or unavailable). It could form part of further work to be carried out after this research is completed.

Proposition 4: The additional effort spent structuring the data using the proposed method helps identify and understand the system data and supports IT priority decision-making.

Proposition 4 concerns the effort spent **structuring the data** needed to support the proposed method (Finding 4). The proposition is that the effort involved is more than compensated by the benefit obtained from an increased understanding of the system (Findings 4, 10, 11, 13, 15, 16, 17, 18 and 19) and support for improved priority decision-making (Finding 24).

The special case leading to the Findings 21, 22, 23 and 25 involved inclusive design, which was not considered in the other two case studies. These findings point to issues and solutions on how to include inclusive design into the list of quality requirements and which stakeholder value could be associated with it. This is seen as leading to the need for further research.

Proposition 5: **If stakeholders are provided with a functioning IT prioritization method, then a better understanding of the planned project outcomes is the result.**

Proposition 5 focuses on the fact that, if stakeholders **use the extended prioritization method** (VIE) (i.e. if they go through the process of extracting the relevant information, putting it in the table and calculating the value to cost ratio), then they will have a better understanding why specific implementation plans (or increments) and thereby specific estimated results should be given priority over others (Finding 29). In other words by carrying out prioritization using the VIE method, an implementation plan emerges, which in turn is very useful for the project planning (Finding 30).

Finally, the additional findings listed at the end of the above table, flag issues with the **scalability** of the VIE table. This was remedied by a reduction of scope and/or selection of only the top 10 key quality requirements. These solutions were confirmed to be working in all the case studies.

6.4 Conclusions

Summarizing the findings above, one can say that within the limitations of three case studies, the proposed method (VIE) has been proven to be working. Without the proposed method, data about stakeholder value is not normally collected and no systematic IT prioritization is carried out. If people use the proposed method, the required effort of structuring data and carrying out the extended prioritization method, in the form of creating a VIE table, increases the understanding of the system at hand and makes the required prioritization transparent. To support scalability of the VIE table it is important to select key quality requirements only.

Chapter 7

Discussion and Conclusions

7.1 Introduction

In the previous chapter the main findings of this research have been presented. This chapter summarizes these main findings and discusses the limitations of the research and the findings. It also identifies the contributions of this research to the academic and practical fields and points out the original contribution to knowledge. Last, but not least, suggestions for future research are made.

7.2 Summary of the research findings

7.2.1 The research question and propositions

In the introductory chapter the initial research question was formulated:

> **"How can better support be provided for priority decision-making regarding IT investment?"**

After consulting the body of literature related to this topic, the research question was refined in the following way:

> **"Within Gilb's Planguage method, how can the current provision to support IT priority decision-making (namely Impact Estimation) be improved?"**

The key reasons for adopting Impact Estimation as the candidate IT prioritization method were its use of metrics and its integrated coverage of the system concepts across the entire system development lifecycle.

Based on the knowledge provided by the large body of literature, the following propositions were formulated:

- **Proposition 1**: Current IT projects fail to capture value and priority information adequately in their system specifications
- **Proposition 2**: The use of metrics is beneficial to understanding the quality requirements and for supporting IT priority decision-making
- **Proposition 3**: Identifying stakeholders more explicitly and considering the impacts on stakeholders of the requirements in terms of value and cost is useful for IT priority decision-making
- **Proposition 4**: The additional effort spent structuring the data using the proposed method helps identify and understand the system data and supports IT priority decision-making
- **Proposition 5:** If stakeholders are provided with a functioning IT prioritization method, then a better understanding of the planned project outcomes is the result.

The above propositions cover the most important aspects of this research These aspects include: the manner in which the concepts of priority and value are currently utilized (Proposition 1), the use of metrics as a proposed technique to improve support for expressing value and through that priority (Proposition 2), the need to cater for multiple stakeholder viewpoints and evaluate stakeholder value against them (Proposition 3), the efficiency, effectiveness and usability of the proposed improved prioritization method (Proposition 4), and finally, the understanding of the system, its planned system improvements and the likely benefits through the use of the proposed method (Proposition 5).

7.2.2 Development of the VIE method

While Impact Estimation (IE) was selected on the basis of its strengths in handling metrics and providing integrated coverage for the system concepts across the system development lifecycle, analyzing IE against the results of the literature review identifies several shortfalls in the IE method. The two main shortfalls are related to the system concepts: its lack of explicit support for multiple stakeholder viewpoints and for explicit stakeholder value. While Planguage specifications can capture this stakeholder data, the IE table fails to present and use this stakeholder data, which is seen as key in determining implementation priorities. To address this, IE is specifically extended to cater for stakeholder value for multiple stakeholders.

The VIE method adds to IE by capturing multiple stakeholder columns to the left-hand side of the IE table. These stakeholder columns are used to capture stakeholder value against the quality requirements for the different stakeholders. The objectives of doing this are to identify which requirements have highest stakeholder value, which stakeholders obtain stakeholder value, which designs have high stakeholder value to cost ratios and the potential implementation order of the proposed designs.

7.2.3 Findings

Three case studies were carried out to support or reject the propositions formulated earlier. In Chapter 5, the case studies and their findings were described. In Chapter 6 these findings were compared across case studies to arrive at a more generalised set of findings. This is possible for two reasons. Firstly, the case studies were carried out to prove the concept of the proposed method. Secondly, the case studies were different and covered slightly different ground, thereby proving that the method can be used in more than one context and still delivers relevant data for the prioritization of IT investment.

The more generalised findings were as follows:

Proposition 1: Current IT projects fail to capture value and priority information adequately in their system specifications.

All three case studies suggest strongly that current IT projects fail to collect stakeholder value data. The original system specifications also show that there is very little focus on explicit stakeholder value in general. Furthermore, prioritization methods are not routinely adopted. Most people seem to use their implicit understanding of stakeholder value to carry out an informal prioritization of the requirements, ignoring many of the other system concepts. This leads to a lack of transparency and an arbitrary prioritization process.

Proposition 2: The use of metrics is beneficial to understanding the quality requirements and for supporting IT priority decision-making.

In all three case studies, there was originally a substantial lack of metrics, if any, supporting the quality requirements. By specifying metrics and setting the target levels, the stakeholders were able to understand the amount of required change and could assess its impact. Given the required system changes were numerically specified, and related to the real world, this in turn (providing additional data such as the workloads involved were known), supported calculations of stakeholder value. Prioritization on the basis of explicit stakeholder value became possible.

Proposition 3: Identifying stakeholders more explicitly and considering the impacts on stakeholders of the requirements in terms of value and cost is useful for IT priority decision-making.

Multiple stakeholders, each with a different viewpoint, were present in all three case studies. The different viewpoints gave rise to different interests in different quality requirements and hence different stakeholder value. Likewise estimated stakeholder costs arising for the system changes varied by stakeholder. The notion of distributing the costs over the stakeholders depending on the stakeholder value they were likely to receive appears attractive. It was noticed during the first case study that the developers received no stakeholder value, but that they did incur costs. This raised debate on whether stakeholder cost ought to be allocated within the VIE table. This was seen as something to be investigated to the future.

Proposition 4: The additional effort spent structuring the data using the proposed method helps identify and understand the system data and supports IT priority decision-making.

A considerable amount of effort was required to analyse and extract the data from the system specifications into a VIE table. However, the insights gained in terms of organizing the key system data and identifying missing data are considered overall to be worthwhile. As a result of the process of creating the VIE table, a clear picture of the systems' components emerged and several aspects, that threatened the success of the proposed systems, were identified. In addition, the completed VIE table gave an overview of the system components and enabled value to cost ratios to be calculated, which assisted prioritization. The third case study included an initial attempt to structure stakeholder value data for social inclusion quality requirements. A simple solution was proposed which worked, but needs further trialing.

It is noted that it is not only effort that is required to create a VIE table, training in the VIE method is required as well. Specifically to be able to develop a VIE table an analyst needs to be able to specify the quantified quality requirements (the quality metrics). The key aspect is to be able to identify the potential scales of measure. Then it is up to the stakeholders to specify the current and target levels for the areas in which they have expertise. The required quality metrics can span both the business and the IT aspects of a system. In addition, especially if evolutionary or agile system development methods are in use, training is likely needed on how to decompose a system into its component parts. The analyst must be able to decompose the proposed system to be able to identify a set of proposed system deliverables to bring about the required system changes.

Proposition 5: If stakeholders are provided with a functioning IT prioritization method, then a better understanding of the planned project outcomes is the result.

This research proposes a prioritization method that utilizes a wide set of system concepts. The mapping of these system concepts to the PDSA cycle and then from the PDSA cycle to the VIE table has been outlined. Moreover, the use of real world metrics with absolute scale data ties the captured system data to tangible metrics enabling simple arithmetic to arrive at stakeholder value. The VIE prioritization method is constructed from fundamental system concepts.

By extracting the system data and building a VIE table, an analyst gains considerable insight into the system components and their mapping to the system concepts. The analyst's acceptance of the VIE table results is not difficult given the underlying method to achieve them. The final calculations, the value to cost ratios point out a proposed implementation plan. These calculations are derived and underpinned by the explicit stakeholder value data and all the other data in the VIE table. The VIE table provides a useful input into project planning.

Additional findings: Finally, the additional findings (listed at the end of the above table), flagged an issue with the scalability of the VIE table in Case Study 1, which was remedied by a reduction of scope and selection of only the top 10 key quality requirements. These solutions were also confirmed to work in Case Studies 2 and 3.

Summarizing the key findings of this study, the proposed method has been proven to be working. Data about stakeholder value is not routinely collected and often an informal IT prioritization leads to rather arbitrary results. When the proposed method is used, the effort of structuring data and carrying out the extended prioritization method increases the understanding of the system to be developed and makes the prioritization transparent and well grounded in stakeholder value. Scalability can be achieved by selecting the key quality requirements only.

7.3 Contribution to the practitioner and academic fields

The findings of this research are relevant to a number of practitioner fields and academic fields:

With regards to the practitioner field of project management and specifically project planning, the findings of the research can eventually lead to improved project planning. The improved prioritization method, could lead to benefits including improved communication between developers and the business, improved quality of the system to be delivered and an improved delivery of the business benefits of a system. Furthermore, given that a considerable number of IT projects still fail, this method could contribute to an improvement in project success rates.

Improving the communication between the developers and the business is considered a much needed aspect within systems development (Moisiadis, 2002). All too often, non-technical people are somewhat afraid of the technical details and they don't understand what information they should be imparting to the developers. In turn, the developers are often loathe to spend the required time to learn about the business, and anyway are more interested in their technical work than discussing the business aspects. VIE offers a method that bridges the gap between these two groups. It specifically demands quality metrics that relate to improving the system for the stakeholders, and these metrics are expressed in terms that the business stakeholders can relate to. By agreeing the current and target levels, as well as the related stakeholder value, the requirements are unambiguously conveyed to the developers. Discussion about prioritization of the delivery of the requirements and the deliverables further aids communication to the developers of the business stakeholders' perspectives.

The contribution to knowledge of this research lies in the concept of IT prioritization on the basis of stakeholder value. The main academic field potentially benefiting from this contribution to knowledge is Software Engineering.

In the field of Software Engineering, the findings of this research challenge the notion that requirements prioritization is sufficient to arrive at a sensible (as opposed to arbitrary) prioritization for IT investment (including development). Most existing prioritization methods take stakeholder value into account only in an implicit way. This research has conceptualised stakeholder value in a more differentiated way by using metrics and thus made stakeholder value more visible as an important system concept and also, as an explicit composite (multi-dimensional) concept within IT prioritization.

7.4 Limitations of the research

One of the major difficulties of this research was accessing organizations willing to give access to the required data, which more often than not were considered rather sensitive and could not be easily communicated over departmental boundaries to the researcher. The dependence on business analysts to extract these data contributed to a number of gaps in the data set.

In response to these problems future research in this area might consider more immersive strategies of inquiry so that trust can be built up and the researcher then has the possibility of collecting the relevant data instead of depending on another person. This choice seems to be between ethnographic research or Grounded Theory.

The number of case studies involved in this research allows for some generalization, but more case studies would further strengthen the claim that this method works. It would also provide the possibility to elaborate further on the concept of stakeholder value.

Finally, if the completion rate of projects using this method could be compared to projects that do not use this method, the contribution of this research to an improved success rate of IT projects could be demonstrated. This would require future research to go beyond the scope of the current research and compare not only project planning, but also project completion.

7.5 Further work

Based on the above limitations of the research, future research would have to include one or several of these aspects:

1. Two or three additional case studies would strengthen the claim that this prioritization method works.
2. These case studies could be extended in scope and not only focus on the project planning, but also on the project completion. Furthermore, it would be an interesting extension of the scope to include the business side of the organization.
3. To improve the completeness of the data a different research strategy should be adopted, which includes the immersion into the studied organizations. Ethnography type research or action research would be appropriate.

Further recommended future research could also include:

1. Case Study 2, the IT Services Bid could be followed up. Having investigated the output of a services proposal, further work could be carried out with a view to improving the bid process itself. One suggestion is to trial making the

production of a VIE table a mandatory output of a bid process to help ensure that the business cases for both the client and the IT services organization are made more explicitly.

2. Expanding the VIE method to include the aspects of IE that have been considered out of scope for this initial proof of concept research, namely uncertainty, credibility, safety deviations and safety margins.
3. The IE method, and hence the VIE method, has weaknesses in understanding where the ‘sweet spots’ are for quality/benefit impacts regarding resource utilization. For example, research would be useful to investigate how development costs vary against improving target quality levels, and how stakeholder value varies against improving target quality levels. There is a need to recognize at which point efforts to improve target quality levels become less cost-efficient.
4. Research could be carried out extending the VIE method to include stakeholder costs alongside stakeholder value. This is discussed earlier in Section 5.2.5 (Proposition 3).
5. Further work could be done in developing a generic framework for quality metrics (that is the quantified quality requirements). One key aspect is developing a set of reusable scales of measure. Such a framework would likely assist analysts to create VIE tables as identifying the quality metrics is seen by the researcher as the biggest barrier to using IE and VIE.
6. The link between business processes and specifying quantified quality requirements could be further researched. Identifying quality metrics by considering the business process is identified within this research as helpful, but further work to investigate generalizing this as a useful front end to the VIE method is needed.
7. As discussed in Section 2.6, there is recognition within the literature that more research is needed on dependencies.
8. Additional work on defining stakeholder value is required. This research has identified the prioritization factors as dimensions of stakeholder value and shown a simple potential way of addressing stakeholder value in the area of social inclusion.

9. The VIE method executes a prioritization process (see Section 3.8). Further development of an explicit theoretical prioritization process is seen as required.
10. Research on the use of the VIE method in domains other than IT.
11. Automation of the VIE table could be carried out. The author is aware that some work has recently started (2014) to automate Planguage and IE. There have also been some small-scale implementations of IE in the past. To date, use of an Excel spreadsheet has proved reasonably effective.
12. Based on the experience of this research, a further interesting topic could be to study the causes of inadequate system specifications.

Bibliography

Primary Sources

Abrahamsson, P., Salo, O., Ronkainen, J. and Warsta, J. (2002) *Agile software development methods: review and analysis*. [Online] Espoo, Finland: VTT Publications. ISBN 951-38-6009-4. Available at: www.vtt.fi/inf/pdf/publications/2002/P478.pdf (Accessed: 26 November 2014).

Ahl, V. (2005) *An experimental comparison of five prioritization methods*. [Online] Master's Thesis. School of Engineering, Blekinge Institute of Technology, Ronneby, Sweden. Available at: http://www.netlearning2002.org/fou/cuppsats.nsf/all/86a759a57c335911c1257088005e42bc/$file/Master_thesis_Viggo_Ahl.pdf (Accessed: 30 November 2014).

Akao, Y. (ed.) (1990) *Quality Function Deployment: integrating customer requirements into product design*. Cambridge Mass., USA: Productivity Press Inc. ISBN 0915299410.

Anfara, A. and Mertz, N. T. (2006). *Theoretical frameworks in qualitative research*. SAGE Publications, Inc. ISBN 1412914167.

Barbacci, M., Klein, M., Longstaff, T. and Weinstock, C. (1995) *Quality attributes: technical report CMU/SEI-95-TR-021 ESC-TR-95-021*. Pittsburgh, Pennsylvania: Carnegie Mellon University. Available at: http://www.sei.cmu.edu/reports/95tr021.pdf (Accessed: 30 November 2014).

Bass, L., Ivers, J., Klein, M. and Merson, P. (2005) *Reasoning frameworks: technical report CMU/SEI-2005-TR-007.* Pittsburgh, Pennsylvania: Carnegie Mellon

University. Available at: http://www.sei.cmu.edu/reports/05tr007.pdf (Accessed: 26 November 2014).

Beck, K. (2000) *Extreme programming explained: embrace change*. Reading, MA: Addison Wesley. ISBN 0201616416.

Berander, P. (2007) *Evolving prioritization for software product management*, Doctoral Dissertation Series, No 2007:07. Blekinge Institute of Technology. ISSN 16532090. ISBN 9789172951082. Available at: http://www.bth.se/fou/forskinfo.nsf/0/e68ddab28202b60dc125729f003936e0/$file/Berander_diss.pdf (Accessed: 26 November 2014).

Brackett, J. W. (1990) *Software requirements*. Software Engineering Institute (SEI), Carnegie Mellon University. (Technical Report SEI-CM-19-1.2).

Buchanan, L. and O'Connell, A. (2006) 'A brief history of decision making', *Harvard Business Review*, January 2006, pp. 32–41. Available at: https://hbr.org/2006/01/a-brief-history-of-decision-making (Accessed: 30 November 2014).

Clegg, D. and Barker, R. (1994) *Case method: Fasttrack: a RAD approach*. Reading, MA: Addison Wesley. ISBN 020162432X.

Cohen, L. (1995) *Quality Function Deployment: how to make QFD work for you*. Reading, MA: Addison Wesley. ISBN 0201633302.

Creswell, J. W. (2003) *Research design: qualitative, quantitative and mixed methods approaches* (2nd edn). Sage Publications. ISBN 0761924426.

Decision Support Applications (2010) Available at: http://www.lionhrtpub.com/orms/surveys/das/das11.html (Accessed: 26 November 2014).

Deming, W. E. (1993) *The new economics for industry, government, education*. Cambridge, MA: MIT Press. ISBN 9780911379051.

Denne, M. and Cleland-Huang, J. (2004) *Software by numbers: low-risk, high-return development*. Upper Saddle River, NJ: Prentice Hall. ISBN 0131407287.

Fenton, N. E. and Pfleeger, S. L. (1998) *Software metrics: a rigorous and practical approach* (2nd edn). Boston, MA: PWS Publishing Co. ISBN 0534954251.

Flowers, S. (1996) *Software failure: management failure*. Chichester, England: John Wiley and Sons. ISBN 978-0471951131.

Gilb, T. (2005a) *Competitive engineering: a handbook for systems engineering, requirements engineering, and software engineering using Planguage*. Brodie, L. (ed.). Oxford: Elsevier Butterworth-Heinemann. ISBN 0750665076.

Gilb, T. (1988) *Principles of software engineering management*. London: Addison Wesley. ISBN 0201192462.

Gilb, T. (1976) *Software metrics*, Studentlitteratur. ISBN 914412631X. 282 pages. Out of print.

Gilb, T. and Brodie, L. J. (2007) 'What's wrong with agile methods? Some principles and values to encourage quantification', in Stamelos, I.G., and Sfetsos, P. (eds.) *Agile Software Development Quality Assurance*. Information Science Reference. ISBN 9781599042169.

Glass, R. L. (1998) *Software runaways*. Upper Saddle River, NJ: Prentice Hall. ISBN 9780136734437.

Green, P. E. and Wind, Y. (1975) 'New way to measure consumers' judgments', *Harvard Business Review*, 53, pp. 107–117. Available at: http://www.roymorgan.com/~/media/Files/Papers/Pre-2010/19750801.pdf (Accessed: 30 November 2014).

Hammond, J. S., Keeney, R. L. and Raiffa, H. (1998) ‘Even swaps: a rational method for making trade-offs’, *Harvard Business Review*, 76, pp. 137–149. Available at: https://hbr.org/1998/03/even-swaps-a-rational-method-for-making-trade-offs (Accessed: 30 November 2014).

Hubbard, D. W. (2007) How to measure anything: finding the value of intangibles in business. Hoboken, Jersey: John Wiley & Sons, Inc.

Kaplan, R. S., and Norton, D. P. (1996) *The Balanced Scorecard: translating strategy into action*. Boston, Massachusetts: Harvard Business School Press. ISBN 0875846513.

Kaposi, A. and Myers, M. (1994) *Systems, models and measures (formal approaches to computing and information technology)*. Berlin: Springer-Verlag, Heidelberg. ISBN 0387197532.

Keeney, R. L. (1992) *Value-focused thinking: path to creative decision making.* Cambridge, MA: Harvard University Press. ISBN 9780674931978.

Keeney, R. and Raiffa, H. (1976) *Decisions with multiple objectives: preferences and value tradeoffs*. John Wiley & Sons, Inc. ISBN 0-471-46510-0.

Khan, K. A. (2006) *A systematic review of requirements prioritization*. [Online] Master's thesis. Software Engineering, Blekinge Institute of Technology, Ronneby, Sweden. Available at: http://www.bth.se/fou/cuppsats.nsf/all/cd4aa06c829ac1bac125724300592671/$file/Thesis.pdf (Accessed: 30 November 2014).

Kuhn, T. (1962 [50th Anniversary Edition: 2012]) *The structure of scientific revolutions*. Chicago: University of Chicago Press. ISBN 9780226458113.

Larman, C. (2004) *Agile and iterative development: a manager’s guide*. Boston, MA: Addison Wesley. ISBN 9780131111554.

Lauesen, S. (2002) *Software requirements: styles and techniques*. Addison Wesley. ISBN 0201745704.

Leffingwell, D. (2010) *Agile software requirements: lean requirements practices for teams, programs, and the enterprise (agile software development)*. Boston, MA: Addison-Wesley. ISBN 978-0321635846.

Leffingwell, D. and Widrig, D. (1999) *Managing software requirements: A unified approach*. Addison-Wesley. ISBN 978-0201615937.

Lehtola, L. (2006) *Providing value by prioritizing requirements throughout software product development: state of practice and suitability of prioritization methods*, Licentiate thesis. Helsinki University of Technology, Department of Computer Science and Engineering.

Mead, N. (2006) *Requirements prioritization introduction*. Software Engineering Institute (SEI), Carnegie Mellon University. Available at https://buildsecurityin.us-cert.gov/articles/best-practices/requirements-engineering/requirements-prioritization-introduction (Accessed: 26 November 2014). Extracted and adapted from a report by Chung, Hung, Hough and Ojoko-Adams, Security Quality Requirements Engineering (SQUARE): Case Study Phase III (Technical report CMU/SEI-2006-SR-003).

Oates, B. J. (2006) *Researching information systems and computing*. Sage Publications. ISBN 1412902231.

Perez, J., Jimeno, J. and Mokotoff, E. (2001) *Another potential strong shortcoming of AHP*. [Online] Universidad de Alcalá, Madrid. Available at: http://EconPapers.repec.org/RePEc:alc:alcddt:8/02 (Accessed: 30 November 2014).

Pohl, K. (1996) *Process-centered requirements engineering*. New York, NY: John Wiley & Sons Inc. ISBN 978-0863801938.

RAE and BCS (2004) *The challenges of complex IT projects*. Royal Academy of Engineering and British Computer Society Working Group.

Saunders, M., Lewis, P. and Thornhill, A. (2000) *Research methods for business students* (2nd edn). Upper Saddle River, NJ: Pearson Education Limited. ISBN 0273639773.

Sivzattian, S. V. (2003) *Requirements as economic artifacts: a portfolio-based approach*. Ph.D. Thesis. Department of Computing, Imperial College of Science, Technology and Medicine, London.

Stapleton, J. (ed.) (2003) *DSDM: Business focused development* (2nd edn). Addison Wesley. ISBN 0321112245.

Wiegers, K. E. (1999) *Software requirements*. Microsoft Press.

Woodward, S. (2003) 'Quantified objectives: managing software development projects', *Methods and Tools*, 11, pp. 25–41. Available at: www.methodsandtools.com/archive/archive.php?id=6. (Accessed November 30 2014).

Yardley, D. (2002) *Successful IT project delivery: learning the lessons of project failure*. Addison-Wesley. ISBN 978-0201756067.

Yin, R. K. (2009) *Case study research: design and methods* (4th edn). Sage Publications. ISBN 978-1412960991.

Secondary Sources (Peer-reviewed Research)

Abran, A., Bourque, P., Dupuis, R., and Moore, J.W. (eds.) (2004) *SWEBOK: Guide to the Software Engineering Body of Knowledge: 2004 Version*. IEEE Computer Society Press. ISBN 978-0769523309.

Achimugu, P., Selamat, A., Ibrahim, R. and Mahrin, M. N. (2014) 'A systematic review of software requirements prioritization research', *Information and Software Technology*, 56, pp. 568–585.

Andrews, A., Runeson, P. and France, R. (2004) 'Requirements Trade-offs During UML Design', *11th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems (ECBS '04).* Brno, Czech Republic, 24–27 May 2004. Washington DC: IEEE Computer Society Press. pp. 282–291.

Akao, Y. (1997) 'QFD: past, present, and future', in Gustafsson A., Bergman B. and Ekdahl F., (eds.), *Third International QFD Symposium. Linköping: Quality Technology & Management at Linköping University (ISQFD'97).* Linköping, Sweden, 1–2 October 1997. pp.19–29.

Aurum, A. and Wohlin, C. (2007) 'A value-based approach in requirements engineering: explaining some of the fundamental concepts', in Sawyer, P., Paech, B. and Heymans, P. (eds), *13th International Working Conference on Requirements Engineering: Foundation for Software Quality (REFSQ* 2007*).* Trondheim, Norway, 11–12 June 2007. Berlin: Springer-Verlag, pp.109–115.

Avesani, P., Bazzanella, C., Perini, A and Susi, A. (2005) 'Facing Scalability Issues in Requirements Prioritization with Machine Learning Techniques', *13th IEEE International Conference on Requirements Engineering (RE '05).* Paris, France, 29 August–2 September 2005. Washington DC, USA: IEEE Computer Society Press. pp. 297–305.

Bahl, H. C. and Hunt, R. G. (1984) 'Decision-making theory and DSS design', *ACM SIGMIS Database*, 15, pp. 10–14. Available at: http://dx.doi.org/10.1145/1017726.1017728 (Accessed: 30 November 2014).

Barney, S., Aurum, A. and Wohlin, C. (2008) 'A product management challenge: creating software product value through requirements selection', *Journal of Systems Architecture*, 54, pp. 576–593.

Barney, S., Aurum, A. and Wohlin, C. (2006) 'Quest for a silver bullet: creating software product value through requirements selection', *32nd EUROMICRO Conference on Software Engineering and Advanced Applications (EUROMICRO-SEAA'06).* Dubrovnik, Croatia, 29 August to 1 September 2006. Washington DC: IEEE Computer Society Press, pp. 274–281.

Basili, V. R., Heidrich, J., Lindvall, M., Münch, J., Seaman, C. and Regardie, M. (2009) 'Determining the impact of business strategies using principles from goal-oriented measurement', *9th Internationale Tagung Wirtschaftsinformatik (WI 2009).* Wien, Austria, 25–27 February 2009, 1, pp. 545-554.

Berander, P. and Andrews, A. (2005) 'Requirements prioritization', in Aurum, A. and Wohlin, C. (eds.) *Engineering and managing software requirements*. Berlin: Springer-Verlag, Heidelberg. ISBN 3540250433.

Berander, P. and Jönsson, P. (2006) 'Hierarchical cumulative voting (HCV) - prioritization of requirements in hierarchies', *International Journal of Software Engineering and Knowledge Engineering*, 16(6), pp. 819–849. doi: 10.1142/S0218194006003026.

Berander, P. and Wohlin, C. (2007) 'Decision Aspects' (Chapter 8). *Berander (2007).*

Bhawnani, P., Far, B. and Ruhe, G. (2005) 'Explorative study to provide decision support for software release decisions', *21st IEEE International Conference on Software Maintenance (ICSM'05)*. Budapest, Hungary, 25–30 September 2005.

Bleistein, S. J., Cox, K., Verner, J. and Phalp, K. T. (2006) 'B-SCP: a requirements analysis framework for validating strategic alignment of organizational IT based on strategy, context, and process', *Information and Software Technology*, 48, pp. 846–868. Available at: http://dx.doi.org/10.1016/j.infsof.2005.12.001 (Accessed: 30 November 2014)

Boehm, B. W. (2006) 'Value-based software engineering: seven key elements and ethical considerations', in Biffl, S., Aurum, A., Boehm, B., Erdogmus, H., and Grünbacher, P. (eds.) *Value-Based Software Engineering*. Springer Berlin Heidelberg. pp. 109–132. ISBN 978-3-540-25993-0.

Boehm, B. (2003) 'Value-based software engineering', *ACM SIGSOFT Software Engineering Notes,* 28, p. 4. Available at: http://dx.doi.org/10.1145/638750.638776 (Accessed: 30 November 2014).

Boehm, B., Huang, L. G., Jain, A. and Madachy, R. (2004) 'The ROI of software dependability: the iDAVE Model', *IEEE Software*, 21, pp. 54–61. Available at: http://dx.doi.org/10.1109/MS.2004.1293073 (Accessed: 30 November 2014).

Boehm, B. and Huang, L. G. (2003) 'Value-based software engineering: a case study', *IEEE Computer*, 36, pp. 33–41. Available at: www.csun.edu/~hbcsc521/documents/Value_Based_Case_Study.pdf (Accessed: 30 November 2014).

Boehm, B. W. and Jain, A. (2006) 'An initial theory of value-based software engineering', in Biffl, S., Aurum, A., Boehm, B., Erdogmus, H., and Grünbacher, P. (eds.) *Value-Based Software Engineering*. Springer Berlin Heidelberg. pp. 15–37. ISBN 978-3-540-25993-0.

Boehm, B. and Ross, R. (1989) ‘Theory W Software Project Management: Principles and Examples’, *IEEE Transactions on Software Engineering*, (July 1989), pp. 902–916.

Boehm, B., Port, D. and Basili, V. R. (2002) ‘Realizing the benefits of the CMMI with the CeBASE method’, *Systems Engineering*, 5(1), pp.73–88.

Boehm, B. W. and Sullivan, K. J. (2000) ‘Software economics: a roadmap’, *The Future of Software Engineering, 22nd International Conference on Software Engineering*. Limerick, Ireland, 4–11 June 2000. New York: ACM Press, pp.319–344.

Botta, R. and Bahill, A. T. (2007) ‘A prioritization process’, *Engineering Management Journal*, 19(4), pp. 20–27.

Brereton, P., Kitchenham, B., Budgen, D. and Li, Z. (2008) ‘Using a protocol template for case study planning’, *12th International Conference on Evaluation and Assessment in Software Engineering*, University of Bari, Italy, 26–27 June 2008. Swindon, UK: British Computer Society, pp. 41–48.

Brodie, L. and Woodman, M. (2011) ‘Prioritization of stakeholder value using metrics’, in Maciaszek, L. and Loucopoulos, P. (eds.), *5th International Conference on the Evaluation of Novel Approaches to Software Engineering (ENASE 2010).* Athens, Greece, 22–24 July 2010. Berlin: Springer-Verlag, pp.76–78.

Brodie, L. and Woodman, M. (2008) ‘Towards a rational prioritization process for incremental and iterative systems engineering’, in Nistazakis, M. (ed), *1st International Workshop on Requirements Analysis (IWRA 2008)*. London, England, 6–7 December 2008. London: Pearson. ISBN: 978-1-84776-663-2.

Burke, E., Kloeber, Jr., J. M. and Deckro, R. F. (2002–2003) ‘Using and Abusing QFD Scores’, *Quality Engineering*, 15(1), pp. 2–21.

Carlshamre, P., Sandahl, K., Lindvall, M., Regnell, B. and Natt och Dag, J. (2001) ‘An industrial survey of requirements interdependencies in software product release planning’, *5th IEEE International Symposium on Requirements Engineering (RE 2001),* 27–31 August 2001, Toronto. Washington DC: IEEE Computer Society Press, pp. 84–91.

Chan, L.-K. and Wu, M.-L. (2002) ‘Quality Function Deployment: A literature review’, *European Journal of Operational Research*, 143, pp. 463–497.

Chou, T.-Y., Chou, S.-C. T. and Tzeng, G.-H. (2006) ‘Evaluating IT/IS investments: A fuzzy multi-criteria decision model approach’, *European Journal of Operational Research*, 173, pp. 1026–1046.

Chung, L. and Nixon, B. (1995) ‘Dealing with non-functional requirements: three experimental studies of a process-oriented approach’, *17th International Conference on Software Engineering* (ICSE ’95). Seattle, Washington USA, 23–30 April 1995. New York: Association for Computing Machinery, pp.25–37.

Cusumano, M., MacCormack, A., Kemerer, C. F., and Crandall, W. (2003) ‘Software development worldwide: the state of the practice’*, IEEE Software*, 20, pp. 28–34. Available at: http://dx.doi.org/10.1109/MS.2003.1241363 (Accessed: 30 November 2014).

Dahlstedt, A. G. and Persson, A. (2005) ‘Requirements interdependencies: state of the art and future challenges’, in Aurum, A. and Wohlin, C. (eds.) *Engineering and Managing Software Requirements*. Berlin: Springer-Verlag. pp. 95–116. ISBN 3540250433.

Daneva, M. and Herrman, A. (2008) ‘Prioritization based on benefit and cost prediction: a method classification framework’, *34th Euromicro Conference Software Engineering and Advanced Applications (SEAA2008)*. Parma, Italy, September 3-5, 2008.

Daniels, J., Werner, P. W. and Bahill, A. T. (2001) 'Quantitative methods for tradeoff analyses', *Systems Engineering*, 4, pp. 190–212. John Wiley & Sons, Inc. Available at: http://www.sie.arizona.edu/sysengr/publishedPapers/QuantitativeMethods.pdf (Accessed: 30 November 2014).

Davis, A. (2003) 'The art of requirements triage', *Computer,* 36, pp. 42–49. Available at: http://dx.doi.org/10.1109/MC.2003.1185216 (Accessed: 30 November 2014).

Denzinger, J. and Ruhe, G. (2004) 'Decision support for software release planning using e-assistants', *Journal of Decision Support Systems*, 13, pp. 399–421. Available at: http://dx.doi.org/10.3166/jds.13.399-421 (Accessed: 30 November 2014).

Durillo, J. J., Zhang, Y., Alba, E., and Nebro, A. J. (2009) 'A study of the multi-objective next release problem', *1st International Symposium on Search Based Software Engineering,* Cumberland Lodge, Windsor, England, 13–15 May 2009. New York: IEEE Press, pp. 49–58.

Even, A., Shankaranarayanan, G. and Watts, S. (2006) 'Enhancing decision making with process metadata: theoretical framework, research tool, and exploratory examination', *39th Hawaii International Conference on System Sciences,* Hyatt Regency Kauai, Hawaii, 4–7 January 2006.

Faulk, S. R., Harmon, D. R., and Raffo, D. M. (2000) 'Value-based software engineering (VBSE): a value-driven approach to product-line engineering', *1st International Conference on Software Product-Line Engineering*, Denver, Colorado, August 28–31 2000.

Favaro, J. (2003) 'Value-based management and agile methods', in Marchesi, M. and Succi, G. (eds.) *4th International Conference on XP and Agile Methods*. Genova, Italy, 25–29 May 2003. Berlin: Springer-Verlag, pp.16–25.

Favaro, J. (2002) ‘Managing requirements for business value’, *IEEE Software*, 19, pp. 15–17. Available at: http://dx.doi.org/10.1109/52.991325 (Accessed: 30 November 2014).

Firesmith, D. (2004) ‘Prioritizing requirements’, *Journal of Object Technology*, 3, pp. 35–47. Available at: http://www.jot.fm/issues/issue_2004_09/column4/ (Accessed: 26 November 2014).

Gilb, T. (2006) ‘Ten Design Principles: Some implications for multidimensional quantification of design impacts on requirements’, *16th Annual International Symposium: Systems Engineering: Shining List on the Tough Issues (INCOSE 2006)*. Orlando, Florida USA, 9-13 July 2006.

Gilb, T. (2005b) ‘Design Evaluation: Estimating Multiple Critical Performance and Cost Impacts of Designs’, *15th Annual International Symposium: Systems Engineering: Bridging Industry, Government, and Academia (INCOSE 2005)*. Rochester Riverside Convention Center, Rochester, New York, USA, 10-15 July 2005. New Jersey: Wiley, pp. 1719–1732.

Gilb, T. and Cockburn, A. (2008) ‘Point/Counterpoint’, *IEEE Software*, 25, pp. 64–67. Available at: http://dx.doi.org/10.1109/MS.2008.43 (Accessed 30 November 2014).

Gilb, T. and Maier, M. W. (2005) ‘Managing priorities: a key to systematic decision-making’, *15th Annual International Symposium: Systems Engineering: Bridging Industry, Government, and Academia (INCOSE 2005)*. Rochester Riverside Convention Center, Rochester, New York, USA, 10–15 July 2005. New Jersey: Wiley, pp. 1687–1705.

Gorschek, T. and Wohlin, C. (2006) ‘Requirements abstraction model’, *Requirements Engineering Journal*, 11, pp. 79–101. Available at: http://dx.doi.org/10.1007/s00766-005-0020-7 (Accessed: 30 November 2014).

Greer, D., Bustard, D. W. and Sunazuka, T. (1999) ‘Prioritisation of system changes using cost-benefit and risk assessments’, *4th IEEE International Symposium on Requirements (RE’99).* Limerick, Ireland, 7–11 June, 1999. Washington DC: IEEE Computer Society Press, pp. 180–187.

Greer, D. and Ruhe, G. (2004) ‘Software release planning: an evolutionary and iterative approach’, *Information and Software Technology*, 46, pp. 243–253. Available at: http://www.cs.qub.ac.uk/~Des.Greer/greer%20ruhe%20IST.pdf (Accessed: 30 November 2014).

Grünbacher, P., Köszegi, S., and Biffl, S. (2006) ‘Stakeholder value proposition elicitation and reconciliation’, in *Value-Based Software Engineering*. Springer Berlin Heidelberg. pp. 133–154. ISBN 978-3-540-25993-0.

Gunasekaran, A., Ngai, E. W. T. and McGaughey, R. E. (2006) ‘Information technology and systems justification: a review for research and applications’, *European Journal of Operational Research*, 173, pp. 957–983. Available at: http://dx.doi.org/10.1016/j.ejor.2005.06.002 (Accessed: 30 November 2014).

Hanssen, G. K. and Faegri, T. E. (2006) ‘Agile customer engagement: a longitudinal qualitative case study’, *5th ACM-IEEE International Symposium on Empirical Software Engineering (ISESE ’06)*. Rio de Janeiro, Brazil, 21–22 September 2006. New York: ACM Press, pp. 164–173.

ISO/IEC 25010:2011 (2011) *ISO/IEC 25010:2011 Systems and software engineering -- Systems and software Quality Requirements and Evaluation (SQuaRE) -- System and software quality models* [Online]. Available at: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=35733 (Accessed: November 26 2014).

Jarke, M., Loucopoulos, P., Lyytinen, K., Mylopoulos, J. and Robinson, W. (2011) ‘The brave new world of design requirements’, *Information Systems*, 36, pp. 992–1008. Available at: http://dx.doi.org/10.1016/j.is.2011.04.003 (Accessed: 30 November 2014).

Johansen, T. (2004) 'FIRM: From Waterfall to evolutionary development (Evo) or how we rapidly created faster, more user-friendly, and more productive software products for a competitive multi-national market', *11th European Software Process Improvement Conference, EuroSPI 2004*, Trondheim, Norway, November 10-12, 2004. Berlin: Springer. See also Proceedings of INCOSE 2005.

Johnson, R. B. (1997) 'Examining the validity structure of qualitative research', *Education*, 118, pp. 282–290.

Jones, C. (1995) 'Patterns of large software systems: failure and success', *Computer*, 28, pp. 86–87. Available at: http://dx.doi.org/10.1109/2.366170 (Accessed: 30 November 2014).

Joshi, K. D. (2001) 'A Framework to Study Knowledge Management Behaviors During Decision Making', *34th Hawaii International Conference on System Sciences*. Maui, Hawaii, USA, 3-6 January 2001. Washington DC: IEEE Computer Society Press, pp. 4024.

Kajko-Mattsson, M. and Meyer, P. (2005) 'Evaluating the Acceptor Side of EM^3: Release Management at SAS', *IEEE 4th International Symposium on Empirical Software Engineering*. Queensland, Australia, 17–18 November 2005, Los Alamitos: IEEE Computer Society Press, pp. 315–324.

Karlsson, J. (1996) 'Software requirements prioritizing', *2nd International conference on requirements engineering (ICRE'96)*. Colorado, USA, 15–18 April 1996. New York: IEEE Computer Society Press, pp. 110–116.

Karlsson, L., Berander, P., Regnell, B. and Wohlin, C. (2004) 'Requirements prioritisation: an experiment on exhaustive pair-wise comparisons versus Planning Game partitioning', *8th International Conference on Empirical Assessment in Software Engineering (EASE 2004)*. Edinburgh, Scotland, 24–25 May 2004. Stevenage: IEE, pp.145–154.

Karlsson, J. and Ryan, K. (1997) 'A cost-value approach for prioritizing requirements', *IEEE Software*, 14, pp. 67–74. Available at: http://dx.doi.org/10.1109/52.605933 (Accessed: 30 November 2014).

Karlsson, J. and Ryan, K. (1996) 'Supporting the Selection of Software Requirements', *8th International Workshop on Software Specification and Design (IWSSD '96)*. Schloss Velen, Germany, 22–23 March 1996. Washington DC: IEEE Computer Society Press, pp.146–149.

Karlsson, J., Wohlin, C. and Regnell, B. (1998) 'An evaluation of methods for prioritizing software requirements'. *Information and Software Technology,* 39, pp. 939–947. Available at: http://www.wohlin.eu/ist98-1.pdf (Accessed: 30 November 2014).

Kazman, R., Asundi, J. and Klein, M. (2001) 'Quantifying the costs and benefits of architectural decisions', *23rd International Conference on Software Engineering (ICSE 2001).* Toronto, Ontario, Canada, *12–19 May 2001*. New York: IEEE Computer Society, New York, pp. 297–306.

Kazman, R., In, H. and Chen, H-M. (2005) 'WinCBAM: from requirements negotiation to software architecture decisions', *Information and Software Technology*, 47, pp. 511-520.

Keeney, R. (2013) 'Identifying, prioritizing, and using multiple objectives', *EURO Journal on Decision Processes*, 1, pp. 45–67. Available at: http://dx.doi.org/10.1007/s40070-013-0002-9 (Accessed: 30 November 2014).

Larman, C. and Basili, V. R. (2003) 'Iterative and incremental development: a brief history', *Computer*, 36, pp. 47–56.

Lehtola, L. and Kauppinen, M. (2006) 'Suitability of requirements prioritization methods for market-driven software product development', *Software Process Improvement and Practice*, 11, pp. 7–19. John Wiley and Sons, Ltd. Available at: http://dx.doi.org/10.1002/spip.249 (Accessed: 30 November 2014).

Lehtola, L., Kauppinen, M. and Kujala, S. (2005) 'Linking the business view to requirements engineering: long-term product planning by roadmapping', *13th IEEE International Conference on Requirements Engineering (RE '05)*. Paris, France, 29 August to 2 September 2005. Los Alamitos: IEEE Computer Society Press, pp. 439–443.

Liew, A. and Sundaram, D. (2005) 'Complex decision making processes: their modelling and support', *38th Hawaii International Conference on System Sciences*. Big Island, Hawaii, Jan. 3, 2005 to Jan. 6, 2005. Washington DC: IEEE Computer Society Press.

Lundberg, C. G. (2006) 'Generating early favorites in decision making. Are simple heuristics involved?', *Proceedings of the 39th Hawaii International Conference on System Sciences*. Kauia, Hawaii, USA, 4–7 January 2006. Washington DC: IEEE Computer Society Press, pp. 31c.

Martins, A. and Aspinwall, E. (2001) 'Quality function deployment: an empirical study in the UK', *Total Quality Management*, 12, pp. 575–588. Available at: http://dx.doi.org/10.1080/09544120120060060 (Accessed: 30 November 2014).

McCaffrey, J. D. (2009) 'Using the Multi-Attribute Global Inference of Quality (MAGIQ) technique for software testing', *6th International Conference on Information Technology: New Generations (ITNG '09)*. Washington DC: IEEE Computer Society Press, pp. 738-742. 10.1109/ITNG.2009.81.

Mohamed, A., Ruhe, G. and Eberlein, A. (2007a) 'COTS selection: past, present, and future', *IEEE Intl. Conf. and Workshops on the Engineering of Computer-Based Systems (ECBS'07)*. Tucson, Arizona, 26-29 March 2007. Washington DC: IEEE Computer Society Press, pp. 103–114.

Mohamed, A., Ruhe, G. and Eberlein, A. (2007b) 'Decision support for handling mismatches between COTS products and system requirements', *COTS-Based*

Software Systems (ICCBSS'07). Banff, Alta, 26 February 2007 to 2 March 2007. Washington DC: IEEE Computer Society Press, pp. 63 – 72.

Moisiadis, F. (2002) 'The fundamentals of prioritising requirements', *SETE 2002 Conference: The Five Layers of Systems Engineering and Test & Evaluation*. Sydney, Australia, 29 to 30 October 2002.

Moisiadis, F. (2001) 'A requirements prioritisation tool', *6th Australian Workshop on Requirements Engineering (AWRE 2001)*. Sydney, Australia, 22–23 November 2001.

Moisiadis, F. (2000) 'Prioritising scenario evolution', *International Conference on Requirements Engineering (ICRE 2000)*. Illinois, USA, 19–23 June 2000. Washington DC: IEEE Computer Society Press, pp. 85–94.

Ngo-The, A. and Ruhe, G. (2008) 'A systematic approach for solving the wicked problem of software release planning', *Soft Computing*, 12, pp. 95–108.

Nuseibeh, B. (2001) 'Weaving together requirements and architecture', *Computer*, 34, pp. 115–119. Available at: http://dx.doi.org/doi:10.1109/2.910904 (Accessed: 30 November 2014).

Nuseibeh, B. and Easterbrook, S. (2000) 'Requirements engineering: a roadmap', *Conference on the Future of Software Engineering*. Limerick, Ireland, 4–11 June 2000. New York: ACM Press, pp. 35–46.

Omasreiter, H. (2007) 'Balanced decision making in software engineering – general thoughts and a concrete example from industry', *1st International Workshop: Economics of Software and Computation (ESC'07)*. Minneapolis, USA, 20–26 May 2007. Washington DC: IEEE Computer Society Press, pp. 4.

Ozkaya, I., Bass, L., Nord, R. L. and Sangwan, R. S. (2008) 'Making practical use of quality attribute information', *IEEE Software,* 25, pp.25 -33.

Ozkaya, I., Kazman, R. and Klein, M. (2007) 'Quality-attribute based economic valuation of architectural patterns', *29th International Conference on Software Engineering Workshops (ICSEW'07)*. Minneapolis, USA, 19 to May 27, 2007. Washington DC: IEEE Computer Society Press, pp. 84.

Park, J., Port, D., Boehm, B. and In, H. (1999) 'Supporting distributed collaborative prioritization for Win-Win requirements capture and negotiation', *International Third World Multi-conference on Systemics, Cybernetics and Informatics (SCI'99)*. Orlando, Florida, 31 July to 4 August 1999. Orlando, Florida: International Institute of Informatics and Systemics, 2, pp. 578–584.

Regnell, B., Höst, M., Natt och Dag, J., Beremark, P., and Hjelm, T. (2001) 'An industrial case study on distributed prioritisation in market-driven requirements engineering for packaged software', *Requirements Engineering Journal*, 6, pp. 51–62. Available at: http://dx.doi.org/10.1007/s007660170015 (Accessed: 30 November 2014).

Regnell, B., Svensson, R. and Olsson, T. (2008) 'Supporting roadmapping of quality requirements', *IEEE Software*, 2, pp. 42–47. Available at: http://dx.doi.org/10.1109/MS.2008.48 (Accessed: 30 November 2014).

Rouse, P. and Putterill, M. (2003) 'An integral framework for performance measurement', *Management Decision,* 41, pp. 791–805. Available at: http://dx.doi.org/10.1108/00251740310496305 (Accessed: 30 November 2014).

Royce, W. (1970) 'Managing the development of large software systems', *IEEE WESCON: Western Electronic Show and Convention (WesCon).* Los Angeles, USA, 25-28 August 1970. Washington DC: IEEE, pp. 1–9.

Ruhe, G. and Saliu, M. O. (2005) 'The art and science of software release planning', *IEEE Software*, 22, pp. 47–53. Available at: http://doi.ieeecomputersociety.org/10.1109/MS.2005.164 (Accessed: 30 November 2014).

Ruhe, G., Eberlein, A. and Pfahl, D. (2003) 'Trade-off analysis for requirements selection', *International Journal of Software Engineering and Knowledge Engineering*, 13, pp. 345–366. Available at: http://dx.doi.org/10.1142/S0218194003001378 (Accessed: 30 November 2014).

Runeson, P. and Höst, M. (2008) 'Guidelines for conducting and reporting case study research in software engineering', *Empirical Software Engineering*, 14, pp. 131–164. Available at: http://dx.doi.org/10.1007/s10664-008-9102-8 (Accessed: 30 November 2014).

Saaty, T. L. (1997) 'That is not the analytic hierarchy process: what the AHP is and what it is not', *Journal of Multi-Criteria Decision Analysis*, 6, pp. 320–339.

Saaty, T. L. (1999) 'Fundamentals of the Analytic Network Process', *International Symposium on Analytic Hierarchy Process (ISAHP 1999)*, Kobe, Japan, August 12-14, 1999.

Saaty, T. L. (1990) 'How to make a decision: the Analytic Hierarchy Process', *European Journal of Operational Research*, 48, pp. 9–26. Available at: https://www.ida.liu.se/~TDDD06/literature/saaty.pdf (Accessed 30 November 2014).

Saliu, O. and Ruhe, G. (2005) 'Supporting software release planning decisions for evolving systems', *29th Annual IEEE/NASA Software Engineering Workshop (SEW'05)*. Maryland, USA, 6-7 April 2005. New York: IEEE Press, pp. 14–26.

Siddiqi, J. and Shekaran, M. C. (1996) 'Requirements engineering: the emerging wisdom', *IEEE Software*, 13, 2, pp. 15-19.

Sullivan, K. (2007) 'Introduction to the first workshop on the economics of software and computation', *29th International Conference on Software Engineering (ESC 2007)*. Minneapolis, Minnesota, USA, 19–27 May 2007. Washington, DC: IEEE Computer Society, pp. 125–126.

Taborda, L. J. M. (2004) ‘The Release Matrix for component-based software’, *7th International Symposium on Component-Based Software Engineering (CBSE 2004).* Edinburgh, UK, 24–25 May 2004. Berlin: Springer-Verlag, LNCS 3054, pp. 100–113.

Wagner, S. (2007) ‘Using economics as basis for modelling and evaluating software quality’, *29th International Conference on Software Engineering Workshops (ICSEW’07).* Minneapolis, Minnesota, USA, 20–26 May 2007. Washington DC: IEEE Computer Society Press, pp. 2.

Wang, Y., Liu, D. and Ruhe, G. (2004) ‘Formal description of the cognitive process of decision making’, *Third International Conference on Cognitive Informatics (ICCI’04).* Victoria, Canada, 16–17 August 2004. Washington DC: IEEE Computer Society Press, pp. 74–83

Wohlin, C. and Aurum, A. (2005a) ‘What is important when deciding to include a software requirement in a project or release?’, *International Symposium on Empirical Software Engineering*. Noosa Heads, Australia, 17–18 November 2005. Washington DC: IEEE Computer Society Press. pp. 246–255. Available at: http://wohlin.eu/isese05.pdf (Accessed: November 26 2014).

Wohlin, C. and Aurum, A. (2005b) ‘Criteria for selecting software requirements to create product value: an industrial empirical study’, in Biffl, S., Aurum, A., Boehm, B., Erdogmus, H., Grunbacher, P. (eds.) *Value-Based Software Engineering*. Springer. ISBN 978-0899307282.

Zave, P. (1995) ‘Classification of research efforts in requirements engineering’, *2nd IEEE International Symposium on Requirements Engineering*. York, England, 27–29 March 1995. Washington DC: IEEE Computer Society Press. pp. 214–216.

Zhang, H., Li, J., Zhu, L., Jeffrey, R., Liu, Y., Wang, Q. and Li, M. (2014) ‘Investigating dependencies in software requirements for change propagation analysis’, *Information and Software Technology*, 56, pp. 40–53. Available at: http://dx.doi.org/10.1016/j.infsof.2013.07.001 (Accessed: 30 November 2014).

Appendix A

Planguage Specifications for the Case Studies

A.1 Planguage specification for the Bank Loan System

Note the following is a simplified version of Planguage, for example all dates have been removed and no source information (who or where the data originated) is given. The data highlighted in bold in this specification is captured in the VIE table in Figure 5.4, The VIE table for the Bank Loan System.

Stakeholders: Regulator, IT Department, Customer, Rules Administration, Business Units, Back Office.

Requirements:
Function: Submit request.
Quality requirement: **Reduce time for customer to submit request**.
Scale: Average time taken for defined [request type: Default = Loan].
Past: **30 minutes**.
Goal: **10 minutes**.

Function: Enter customer request details.
Quality requirement: **Reduce time for Back Office to enter request**.
Past: **30 minutes**.
Goal: **10 minutes**.

Function: **Process a customer request.**

Quality requirement: **Reduce time to process customer request**.

Past: **5 days**.

Goal: **20 seconds**.

Quality requirement: Reduce number of complaints.

Scale: Average number of complaints in defined [Time] from defined [Stakeholder].

Past **[Back Office]: 10 per week.**

Goal: **0 per week.**

Past **[Customer]: 25 per week**.

Goal: **5 per week**.

Function: **Update the business rules**.

Quality requirement: Reduce time to update rules.

Scale: Average time taken for defined [request type].

Past: **1 month**.

Goal: **1 day.**

Function: **Distribute business rules**.

Quality requirement: Reduce time taken. Scale: Average time taken.

Past: **2 weeks**.

Goal: **1 day**.

Designs:

APTM: **Automate the rules & test manually**.

Rationale: Speed up the distribution to Back Office staff.

BD: **Back Office loan decisioning system**.

Rationale: Automating applying the rules will save time.

Dependency: **APTM**.

WSS: **Web self-service.**

Rationale: Customers can get a rapid response.

Dependency: **BD, APTM**.

APAT: **Automate the rules & test automatically.**

Rationale: Speed up the distribution to Back Office staff.

Dependency: **APTM**.

A.2 Planguage specification for the IT Services Bid

Stakeholders: Services provider {Services delivery management, Service desk, Hardware engineering, Lifecycle services, New builds, Service providers resolver groups {engineering resolver groups and software resolver groups}}, Customer {Retail store staff, Retail IT management, Retail senior management, Data centres and cloud services, Customer resolver groups}.

Function requirement: Provide services. **Set services pricing**.
Quality requirement: **Reduce cost of ownership.**
Aim: Reduce costs by 20-30% in the first year for services.
Scale: % cost reduction achieved in first year for services.
Goal: 20-30% cost reduction.
Stakeholder value. Cost reduction [Retail IT management, Retail senior management]: The financial sum in GBP of the cost reduction achieved.
Stakeholder value. Fit [Services delivery management]: {geographic operation, staff skills, staff resources, spares resources, customer skills}.

- Does extent of services provider's geographic operation matches the retail store operations?
- Has services provider the right staff skills
- Has the services provider the right staff levels? Is training required? Is there spare capacity?
- Has the services provider the right spares handling in place?
- Can current retail stores staff cope with interaction with services provider or would some additional training help?
- Note [1]: This is an overarching objective. Any cost reductions shown against other objectives for ongoing services are included in the calculation for this requirement.
- Note [2]: Costs due to any retail store expansion will not be included in the reduction calculation.
- Note [3]: Cost savings for customer considered to come from initial pricing, hardware refresh and product integration.

Function requirement: Provide services. Achieve CSI. Monitor customer satisfaction.
Quality requirement: **Ensure customer satisfaction.**
Scale: % customer satisfaction from CSAT and CSIP.
Goal [CSAT] = 80% <- Guess.
Goal [CSIP] = 80% <- Guess.
Stakeholder value. Opinion [Retail IT management, Retail senior management]: Measured by CSAT and CSIP surveys.

Function requirement: Manage services delivery. Ensure business agility.

Quality requirement: **Provide scalability.**

Aim: To enable the customer business to change over time – to expand with new stores, to move stores and to close stores.

Scale: % ability to deliver required services provision including any ad hoc services to support changes in the retail estate within the agreed timescales.

Goal: 95% <- Guess.

Stakeholder value. Effort saving [Retail IT management]: Time saving of IT management effort. (From the customer's viewpoint, this means only the need for a CCP meeting rather than a new contract each time. Assume 20 new stores a year and 8 hours per contract versus 30 mins = 150 hours of senior manager's time = 4K).

Rationale: For the customer to focus on changing their business without having to worry about whether services provision is in place.

Stakeholder value. Cost [Retail IT management]: Financial cost of the technology upgrade.

Stakeholder value. Financial income [Services delivery management]: Financial income from technology upgrade and ongoing extra service calls.

Stakeholder value. Cost [Services delivery management]: Additional costs due to ongoing extra service calls (service desk staff, hardware engineers, spares, etc.).

Note [1]: Pricing mechanism to support this is the CCP (Change Control Procedure). Could have impacts on hardware support if say expanded to a new retail store location that then took the services provision over capacity or if the spares provision had to be adjusted.

Note [2]: Services provider gains more revenue as customer business grows, will have to keep track of the changes by asset management, will have to ensure services delivery capacity when needed and where needed (engineers and spares).

Function requirement: Provide services. Manage services delivery. Ensure governance.

Quality requirement: **Ensure governance.**

Scale: % of specified governance measures in place when audited.

Goal = 95% <- Guess

Stakeholder value. Legal [Retail IT management, Retail senior management]: Peace of mind over due diligence.

Function requirement: Provide services. Manage services delivery. Ensure security.

Quality requirement: **Ensure security.**

Scale: % of specified security measures in place and up-to-date when audited.

Goal = 95% <- Guess.

Stakeholder value. Legal [Retail IT management, Retail senior management]: Peace of mind over due diligence, health and safety, virus attacks.

Function requirement: Provide services. Manage services delivery. Ensure business continuity planning.
Quality requirement: **Achieve business continuity planning.**
Note: On any problem occurring with access to or loss of services data centre.
Scale: % conformance with business continuity plans when audited.
Goal = 95% <- Guess.
Stakeholder value. Opinion [Retail IT management]: Peace of mind over support being always available during contracted hours.

Function requirement: Provide services. Deliver services. Provide first line support.
Quality requirement: **Achieve answer speed SLA.**
Scale: % of contacts answered within 30 seconds of the end of the IVR message.
Goal = 90%
Minimum = 70%
Stakeholder value. Effort saving [Retail store staff]: Saves two minutes per call logged <- Guess. 57600 calls a year = 115,200 minutes saved = 1,920 hours per annum. At say GBP13 an hour = approx. 25K.

Function requirement: Provide services. Deliver services. Provide first line support.
Quality requirement: **Reduce abandon rate SLA.**
Scale: The percentage of calls where the user does not hang up before the call is answered. (Contacts where the user hangs up during or within 30 seconds of the end of the IVR message shall not be considered abandoned).
Goal = 95%
Minimum = 75%
Stakeholder value. Effort saving [Retail store staff]: Saves ten minutes per call logged <- Guess. 57600 calls a year = 576000 minutes saved = 9,600 hours per annum. At say, GBP13 an hour = approx. 125K.

Function requirement: Provide services. Deliver services. Provide first line support.
Quality requirement: **Attempt first time fix PI.**
Scale: For telephone calls to the service desk, the percentage of incidents resolved during the first contact.
Goal = 50%
Stakeholder value. Effort saving [Retails store staff]: Time saving if call is resolved on first call. Say saves an extra hour for each call. Then if 30% of calls are software related (Guess), 50% resolved = 8640 hours and GBP13 an hour for staff time = GBP 112,320.

Function requirement: Provide services. Deliver services. Provide first line support.
Quality requirement: **Transfer from web portal SLA.**

Scale: % of contacts made by web portal, which are entered into an incident management system within 30 minutes (during the relevant hours of support).
Goal = 90%
Note: Why is the transfer across mentioned like this? Is this an opportunity for improvement?
Stakeholder value. Risk of financial loss [Service provider]: Service credits awarded if SLA breached.

Function requirement: Provide services. Deliver services. Provide first line support.
Quality requirement: **Achieve average incident creation time PI.**
Scale: The average time between a call being logged at a service desk and it being assigned to a resolver group.
Goal = 20 minutes.
Note: Could impact on SLA conformance if delays.

Function requirement: Provide services. Deliver services. Provide second line support.
Quality requirement: **Fix hardware severity 1 SLA.**
Scale: Resolve within **four hours** of being passed to the engineering resolver group (providing less than 20 minutes spent at the service desk)
Goal = 90%
Minimum = 70%
Stakeholder value. Risk of financial loss [Retail senior management]: Business impact of critical failure.
Stakeholder value. Extra effort [Retail store staff]: Will have to take measures to work around the faults.
Stakeholder value. Risk of financial loss [Service provider]: Service credits awarded if SLA breached.

Function requirement: Provide services. Deliver services. Provide second line support.
Quality requirement: **Fix hardware severity 2 SLA.**
Scale: Resolve Severity 2 **by end of next trading day** of being passed to the engineering resolver group (providing less than 20 minutes spent at the service desk).
Goal [Severity 2] = 90%
Minimum [Severity 2] = 70%
Stakeholder value. Risk of financial loss [Retail senior management]: Business impact of failure.
Stakeholder value. Extra effort [Retail store staff]: Will have to take measures to work around the faults.
Stakeholder value. Financial loss [Service provider]: Service credits awarded if SLA breached.

Function requirement: Provide services. Deliver services. Provide second line support.
Quality requirement: **Fix hardware severity 3 SLA.**

Scale: Resolve Severity 3 **by end of the third trading day** of being passed to the engineering resolver group (providing less than 20 minutes spent at the service desk).
Goal [Severity 3] = 90%
Minimum [Severity 3] = 70%
Stakeholder value. Extra effort [Retail store staff]: Will have to take measures to work around the faults.
Stakeholder value. Risk of financial loss [Service provider]: Service credits awarded if SLA breached.
Note: The hardware support service levels shall be uplifted by 1% when the services provider has met the service levels in a minimum of 10 months during the last contract year.

Function requirement: Provide services. Deliver services. Provide lifecycle services.
Quality requirement: **Update patches SLA.**
Scale: % of managed assets that are patched within an agreed monthly release window of not less than three days. (If no release then level shall be considered met.)
Goal = 95%
Minimum = 75%
Stakeholder value. Effort saving [Retail store staff]: Time saved due to reduced number of incidents to log and progress.
Stakeholder value. Effort saving [Service desk]: Fewer calls to log, resolve, transfer and progress.
Stakeholder value. Effort saving [Software resolver groups]: Fewer calls to resolve.
Stakeholder value. External dependency [Retail IT management]: Reduce dependency on specific resolver groups.

Function requirement: Provide services. Deliver services. Provide new build. Refresh hardware.
Aim: To swap out old legacy hardware products to improve resilience.
Quality requirement: **Achieve hardware refresh**.
Scale: % reduction in calls due to the replaced hardware products.
Goal: 10% <- Guess.
Stakeholder value. Effort saving [Retail store staff]: Saves time to log and progress the call.
Stakeholder value. Cost reduction [Retail IT management]: Fewer calls needing resolution due to better hardware.
Stakeholder value. Effort saving [Service desk]: Saves time to log and progress the call.
Stakeholder value. Effort saving [Hardware engineers]: Saves time to resource spares and resolve the call.
Stakeholder value. Financial income reduction [Services delivery management]: Ongoing fewer calls due to better customer products in place.

Function requirement: Provide services. Deliver services. Manage print services.

Quality requirement: **Deliver consumables PI.**

Scale: % delivered the next business day.

Goal = 75%

Stakeholder value. Effort saving [Retail store staff]: Saves staff having to monitor and place orders.

Stakeholder value. Environmental concerns [Retail senior management]: Responsible recycling of printer cartridges.

Function requirement: Provide services. Achieve Continual Service Improvement (CSI). Integrate products.

Quality requirement: **Achieve product integration.**

Aim: Reduction in the amount of software and hardware products being supported by transition to smaller core of key technology (aka Integration).

Scale: Cost reduction due to Integration changes.

Goal: ? Unknown. This needs to be determined by the services delivery management.

Stakeholder value. Effort saving [Retail store staff]: Less staff effort as fewer calls to log and progress (more resilient kit in place, newer kit) – will save staff time.

Stakeholder value. Financial saving [Retail IT management]: Reduction in ongoing service delivery charges due to fewer products and fewer calls.

Stakeholder value. Financial income [Services delivery management]: Gain revenue from carrying out the ad hoc integration tasks and commissioning the new kit.

Stakeholder value. Financial income reduction [Services delivery management]: Ongoing fewer calls due to better customer products in place.

Stakeholder value. Cost reduction [Services delivery management]: Ongoing fewer software applications and antiquated hardware to support so will not need to fund those skills and spares.

A.3 **Planguage specification for the Research Project**

Stakeholders: Down's Syndrome organizations, researchers, systems support, tertiary users – teachers, secondary users – carers and primary users [14 years and over].

Functional requirement: Make mobile calls.

Quality requirement: Improve ability to make mobile / smartphone calls unaided (Mobile calls).

Aim: More primary users should be able to make a mobile phone / smartphone call without any assistance. Enhanced interface options and optional control over calls made and received should enable this.

Scale: % of primary users able to make mobile calls unaided.
Past [**Age 10-18,** Across Norway, Germany and UK (N=144), January 2014**]: 28%** <- p60 D2.1.

Past [**Age 19 or over**, Across Norway, Germany and UK (N=131), January 2014**]: 63%** <- p60 D2.1.

Past [**Age 14 or over]: 50 <- Guess** based on other past levels.

Goal [Age 14 or over, Poseidon Pilot 1 software, Assessed as able to use unaided**]: > 75%** <- Rough Guess (based on age ≥19 'Uses apps on a smartphone' percentage = 40% (N= 131) <- p51 D2.1, and able to make calls with no, little or some help = 82% (N=131) = <- p60 D2.1).

Frequency of use: Assume makes 10 calls a day.

Measurement: Initially during pilots, by interview with secondary user. Also by observation. When live, by questionnaire. The issue being how unaided the use of the phone actually is. Another means of measuring could be by determining the actual usage in the field against location type and maybe personalized user profile (that is, live system usage statistics).

Stakeholder value:

The primary user achieves better social inclusion by being able to make mobile calls (use of 'cool' technology) and by being able to make mobile calls unaided is enabled to be more independent.

- Primary user, secondary user, tertiary user: functionality.able to contact
- Primary user: use of technology.social inclusion
- Primary user: able to contact.social inclusion
- Primary user: use of technology.wellbeing – self-confidence
- Primary user: able to contact.wellbeing – self-confidence
- Primary user: able to contact.greater independence
- Secondary user: able to contact.wellbeing - peace of mind
- Primary user: able to contact.wellbeing - peace of mind
- Primary user: able to contact.wellbeing - safety

- Secondary user: support organizing.efficiency/time-saving.

Assumption: That mobile signal is present across Norway, Germany and UK. There are issues in certain parts of the UK.
Assumption: Will probably have to be a smartphone?
Assumption: That smartphones can be purchased. Already age 10-18 ownership of smartphones is 23% (N=33), and age 19 or over ownership is 39% (N=51) <- p18 D2.1.

Functional requirement: Plan local journeys.
Quality requirement: Improve ability to plan in advance a local journey unaided (Local journeys).
Aim: Poseidon should enable more IT tasks to be accomplished without any assistance. 'Plan Journey' seems an appropriate task as 'assisting travelling' is identified as being key functionality and it would also assist a primary user if the need to reroute a journey occurred while travelling. Restricted to local journeys.
Scale: % of primary users able to plan in advance local journeys unaided.
Past [Age 10-18, Across Norway, Germany and UK (N=143), Plan Journey**]:** approximately **0%** <- p59 D2.1.
Past [Age 19 or over, Across Norway, Germany and UK (N=130), Plan Journey**]:** approximately **7%** <- p59 D2.1.
Past [Age 14 or over]: 5 <- Guess based on other past levels.
Goal [Age 14 or over, Poseidon Pilot 1 software, Plan Journey**]: > 40% <- Guess** (set based on age ≥19 ability to travel in local area (N=131) = 37% <- p58 D2.1).
Note: Maybe targets are rather ambitious given that with some help the primary users only currently achieve 5% and 13% respectively <- p58 D2.1. Rather depends on what is causing the problems with planning. Restricting to local journeys and planning in advance of travelling should help.
Frequency of use: Assume uses once a week.
Measurement: Initially during pilots, by interview with secondary user. Also by observation. When live, by questionnaire. Will also eventually have live system usage statistics.
Stakeholder value:

- Primary user, secondary user, tertiary user: support travel.functionality
- Primary user: support activities.social inclusion
- Primary user: use of technology.social inclusion
- Primary user: use of technology.wellbeing – self-confidence
- Primary user: support travel.greater independence
- Primary user: support travel.well-being –safety
- Primary user: time management.greater independence
- Primary user: access to data.greater independence
- Secondary user: support organizing.efficiency/time-saving.

Functional requirement: Send message.

Quality requirement: Increase ability to send messages unaided (Messages).

Aim: The primary user should be able to send messages to other primary users and to secondary (carers and friends) and tertiary users unaided. Likely this will be achieved via an app with supporting functionality.

Scale: % of primary users sending messages unaided.

Past [Age 10-18, unaided SMS/email, January 2014, Across Norway, Germany and UK (N=144)**]: 15%/8%** <- p60 D2.1.

Past [Age 19 or over, unaided SMS/email, January 2014, Across Norway, Germany and UK (N=144)**]: 37%/25%** <- p60 D2.1.

Past [Age 14 or over]: 25 <- Guess based ober other past levels.

Goal [Age 14 or over, App message, Poseidon Pilot 1**]: >48% <- Guess** (Based on age ≥19 figures for current use with little and some help <- p60 D2.1).

Frequency of use: Assume sends 10 messages a day.

Measurement: Initially during pilots, by interview with secondary user. Also by observation. When live, by questionnaire. Will also eventually have live system usage statistics.

Assumption: That all messages will be sent from within an app, which will allow logging of conversations to aid continuity of conversations. It will also allow greater support to be provided to the primary user in creating and sending messages.

Stakeholder value:

- Primary user.secondary user, tertiary user: able to contact.functionality
- Primary user: able to contact.social inclusion
- Primary user: able to contact.wellbeing – self-confidence
- Primary user: use of technology.social inclusion
- Primary user: use of technology.wellbeing – self-confidence
- Primary user: support activities.social inclusion
- Primary user: able to contact.greater independence
- Secondary user: support organizing.efficiency/time-saving.

Functional requirement: Determine location of primary user.

Quality requirement: Reduce time spent determining individual's location (Location).

Aim: The mutually agreed, on-going monitoring of location against travel plans would represent the full functionality. However, an initial step would be to provide the secondary user with the ability to locate the last known position of the primary user at all times (given signal can be lost indoors). D2.1 p35 states that approximately 78% of secondary users (N=366 to 380) would find checking of location to be "very helpful".

Scale: % of time away from home that location of primary user can be reasonably established.

Past [Age 14 or over]: 0% <- Guess – maybe Find a Friend is used?

Goal [Age 14 or over]: >80% <- Guess.

Assumption: Currently the secondary user relies on being contacted if any problems arise over arrival or departure from a location. The secondary user will possibly monitor setting off to school/work, arriving home from school/work, if a different person from normal is collecting, or if going out with a friend, etc. So Poseidon will add additional possibilities for checking. As such, it might actually take up more time initially, though probably is more time effective than the alternative of phoning the primary user to find out their location. Poseidon will definitely be more time-efficient once it monitors against travel plans.

Frequency of use: Assume used 4 times a day.

Measurement: Before and during the pilots, the secondary users would be asked to keep a diary and log whenever they needed to check the location of a primary user. Assumption: For independent travel, that currently this information has to be obtained by making a mobile call or relying on school/work to alert secondary user if primary user fails to arrive. So there could be savings in effort for the tertiary users if the system took over this task.

Stakeholder value:

- Secondary user: able to locate.functionality
- Primary user: support travel.greater independence
- Primary user: support travel.well-being – safety
- Secondary user: able to locate.wellbeing - peace of mind
- Secondary user: able to locate.efficiency/time-saving
- Tertiary user: reporting effort.efficiency/time-saving.

Functional requirement: Check on emotional wellbeing of primary user.
Quality requirement: Reduce time spent checking on emotional wellbeing (Check all is well).
Aim: Poseidon should enable routine and ad hoc checking with the primary user that 'all is well' to be set up by the secondary user. This can involve simple response to a question over emotional wellbeing, monitoring the primary user's patterns of behaviour to detect potential issues, or 'Face-time' conversations.
Scale: Time in minutes spent weekly by the secondary user monitoring the emotional state of the primary user when away from home.
Past [Age 10-18]: 120 minutes <- Guess based on making phone calls to check up.
Past [Age 19 or over]: 240 minutes <- Guess based on making phone calls to check up.
Goal [Age 14 or over]: <60 minutes <- Guess based on ad hoc use of simple app exchanges asking about wellbeing.
Frequency of use: If set on auto, then continuously, else assume 4 times a day.
Measurement: Before and during the pilots, the secondary users would be asked to keep a diary and log whenever they needed to check the emotional welfare of a primary user.

Stakeholder value:

- Secondary user: able to check.functionality
- Primary user: faster reassurance.wellbeing - health
- Primary user: able to contact.greater independence
- Secondary user: able to check.wellbeing - peace of mind
- Secondary user: able to check.efficiency/time-saving.

Assumption: The secondary user should be able to set up checking to occur automatically at appropriate times. For example, monitoring to occur at a school break time or after an hour at work. The secondary user then has some of the task of remembering to check made easier. In addition, if the technology can detect that the primary user is getting upset, then the secondary user can be alerted, again removing the need for the secondary user to keep monitoring the primary user in case there is a problem. It would also mean that with timely interventions that situations are less likely to get out of control. One outcome should be greater independence for the primary user as there is the ability to communicate that 'all is well'.

Functional requirement: Setup a system user.
Quality requirement: Improve time taken to setup a system user (Setup time).
Aim: Through use of default profiles and select of options for personalization, the task of setting up a tablet or mobile phone for the primary user should be time efficient.
Scale: Average time in minutes to set up the technology for a primary user.
Past [Initial set up of Tablet, 5 contacts and 5 apps, New Owner of Tablet, June 2015**]: < 60 minutes? <- Guess** (based on setting up a Samsung tablet).
Goal [Initial set up of Tablet, Primary User, Systems Support, Pilot 1**]: < 30 minutes <- Guess.**
Frequency of use: Once per user – though might have numerous devices which may or may not sync.
Measurement: Time the amount of time it takes to set up a primary user's smartphone from 'new, out of the box'. The more automated the set up, the less chance for errors to be made, the more rapidly changes can be assimilated, and the more efficient the process.

Stakeholder value:

- Systems support: time to setup.efficiency/time-saving
- Secondary user: time to setup.efficiency/time-saving.

Functional requirement: Monitor system performance.
Quality requirement: Ensure acceptable system reliability (System reliability).
Aim: The system must be available for use by end-users and one key aspect of this is that the system should not frequently breakdown. At Pilot 1 stage, it is quite early to be monitoring this, however the measure is so important that any issues need picking up early. The UniversAAL platform could also be measured.
Scale: MTBF in hours for system network (especially during Pilot 1 and 2).
Past [SmartTracker Platform, **June 2014]: ?** - current data needed.
Goal [Poseidon/SmartTracker Platform, **June 2015]: 300 <- Guess** -setting system failure at about once every two weeks.
Frequency of use: Measure monthly. Impact depends on the number of breaks.
Measurement: System statistics.

Stakeholder value:

- Systems support: support effort.efficiency/time-saving.maintenance
- Systems support: maintenance support effort.cost
- Secondary user: system works.wellbeing - peace of mind
- Primary user: system works.wellbeing - peace of mind
- Primary user: system works.well-being – safety.

Functional requirement: Monitor system quality of software and hardware.
Quality requirement: Ensure system quality of software and hardware (Incidents).
Aim: The Poseidon platform and app software must not have too high a level of defects, especially incidents that impact on the end-users. Probably useful to measure the number of problems actually hit by the system as well. The UniversAAL platform could also be measured.

Scale: Number of incidents of defined type by defined severity reported weekly.
Past [SmartTracker Platform, software, **last week, B severity]: ?** – current data needed.
Goal [Poseidon/SmartTracker Platform, software, **last week, B severity]: < 5 per month <- Guess** – B severity incidents are serious and some might bring down the system, but do not stop the system completely. Given a reasonable fix rate, this would seem a reasonable target.
Frequency of use: Measure weekly. Impact depends on the type, severity and number of incidents.
Measurement: via reported incidents log.

Stakeholder value:

- Systems support: maintenance support effort.efficiency/time-saving
- Systems support: maintenance support effort.cost
- Secondary user: system works.wellbeing - peace of mind
- Primary user: system works.wellbeing - peace of mind
- Primary user: (depending on type of defect) system works.well-being –safety.

Functional requirement: Monitor public awareness of system.
Quality requirement: Increase the awareness among third party developers (Third party developers).
Aim: Poseidon aims to provide a basis for further third party development and so encouraging and monitoring third party interest from such developers is essential.
Scale: Number of third party developers participating in or following POSEIDON on social media.
Past [Poseidon, **June 2014]: 0 <- Guess.** Anyone in project group know the current data?
Goal [Poseidon, **June 2015]: 10 <- Guess.** This is a D2.2 suggested measure of project impact.
Frequency of use: Measure monthly. (It might at some stage be useful to measure how often system developers access the website, though that depends on the type of information available.)

Measurement: Use social media statistics. Maybe could also track visitors to the website?

Stakeholder value:

- Poseidon project researchers: visibility. promotion of ideas and research
- Downs Syndrome organizations: visibility.promotion of new ideas
- Primary user: potentially more apps (more functionality).social inclusion.

Functional requirement: Monitor system impact.

Quality requirement: Satisfaction of the primary users with the system (Satisfaction with the system).

Aim: The secondary users should observe that overall the delivered Poseidon system provides benefit for their primary users.

Scale: % of secondary users observing a positive impact on a primary user from the use of Poseidon technology to support tasks.

Past [All tasks, Use of Laptops&PV/Tablets/Smartphones]: 82%/85%/56% <- p37 D2.1 says approximately half can use without help. Doesn't capture user opinion though.

Goal [All tasks]: >50% <- D2.2 ">50% should like using Poseidon".

Frequency of use: Measured one for each prototype during evaluation.

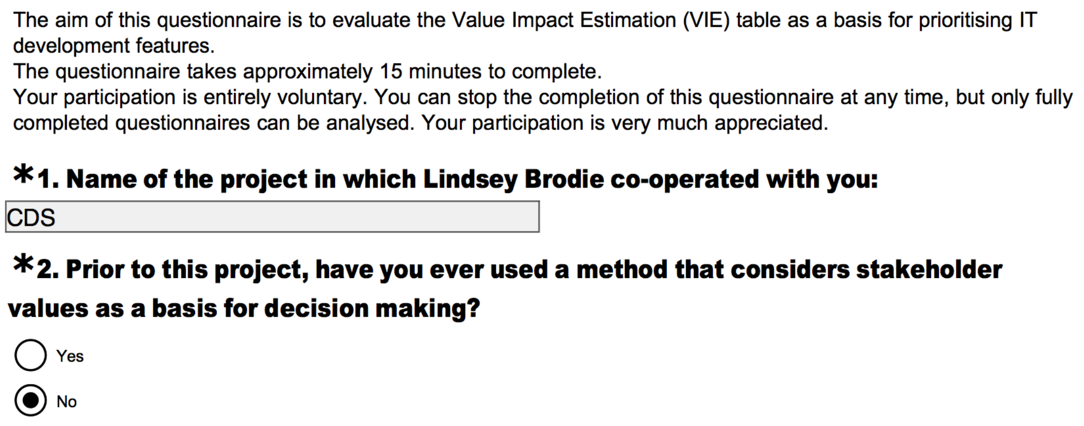
Measurement: Initially during pilots, by interview with secondary user.

Stakeholder value: Overall system measure.

Appendix B

Evaluation of the Case Studies

B.1 Evaluation of Case Study 1 – The Bank Loan System by one of the consultants



The aim of this questionnaire is to evaluate the Value Impact Estimation (VIE) table as a basis for prioritising IT development features.
The questionnaire takes approximately 15 minutes to complete.
Your participation is entirely voluntary. You can stop the completion of this questionnaire at any time, but only fully completed questionnaires can be analysed. Your participation is very much appreciated.

***1. Name of the project in which Lindsey Brodie co-operated with you:**

CDS

***2. Prior to this project, have you ever used a method that considers stakeholder values as a basis for decision making?**

Yes [✓ No]

3. Which method did you use that considered stakeholder values?

4. On a scale of 1 to 10, with 1 being the worst possible method and 10 being the best possible method, how do you rate this previously used method of IT prioritisation?

	worst possible method 1	2	3	4	5	6	7	8	9	best possible method 10
Usefulness	○	○	○	○	○	○	○	○	○	○
Power of clarification	○	○	○	○	○	○	○	○	○	○
Ease of use	○	○	○	○	○	○	○	○	○	○
Effectiveness	○	○	○	○	○	○	○	○	○	○
Time efficiency	○	○	○	○	○	○	○	○	○	○

***5. On a scale of 1 to 10, with 1 being the worst possible method and 10 being the best possible method, how do you rate the Value Impact Estimation (VIE) table as a method of IT prioritisation?**

	worst possible method 1	2	3	4	5	6	7	8	9	best possible method 10
Usefulness	○	○	○	○	○	○	○	[✓]	○	○
Power of clarification	○	○	○	○	○	○	○	○	[✓]	○
Ease of use	○	○	○	○	○	○	[✓]	○	○	○
Effectiveness	○	○	○	○	○	○	○	[✓]	○	○
Time efficiency	○	○	○	○	○	○	○	○	[✓]	○

***6. Below you find a number of statements related to the use of a Value Impact Estimation table (VIE table) in your project. Please choose for each statement, how much you agree or disagree with it.**

	strongly agree	agree	undecided	disagree	strongly disagree
The VIE table captures explicit stakeholder values	[✓]				
The VIE table supports the mapping of requirements and design solutions to increments.		[✓]			
The VIE table gives a clear understanding of the different view points and values of different stakeholders involved in the project	[✓]				
The VIE table gives a clear understanding of the system development costs of each component			[✓]		
THE VIE table handles performance attributes (i.e.non-functional requirements) well	[✓]				
The VIE table can cope with large numbers of requirements (scales up well)		[✓]			
The VIE table allows reprioritization in future increments without further input from stakeholders (unless major changes have taken place)		[✓]			
The VIE table links the IT development of the project at hand to other IT investments		[✓]			
The VIE table enables the integration of functional and performance requirements without loss of stakholder value		[✓]			
The VIE table captures requirements at different levels of refinement	[✓]				
The VIE table is useless for the prioritisation of the stakeholder values.					[✓]
The VIE table captures interdepencies among requirements and among design solutions		[✓]			

The VIE table shows how major system requirements impact on stakeholder values	○	●	○	○	○
The VIE table supports future prioritisation by allowing the capture of data after the system has become operational	○	●	○	○	○
The VIE table provides data related to return of investment	○	○	○	●	○
The VIE table is overly complex	○	○	●	○	○

7. Please list below the three best aspects of the VIE table (in no particular order)

Good Aspect 1	Facilitates communication between parties
Good Aspect 2	Clarifies priorities
Good Aspect 3	Engenders more agreement

8. Please list below the three worst aspects of the VIE table (in no particular order)

Poor Aspect 1	Extra workload
Poor Aspect 2	No automation
Poor Aspect 3	Sometimes far from actual situation

***9. I intend to use the VIE table in other projects**

[✓ yes]

no

10. Do you have any other comments, questions, or concerns regarding the VIE table?

A useful way of looking at developments and ongoing improvements

This is the end of the questionnaire. I would like to thank you for your kind co-operation. If you are interested in the findings of this study, please email me at l.brodie@mdx.ac.uk to request the summary of the findings. These will be available in October 2014.

B.2 Evaluation of Case Study 2, the IT Services Bid Project by a bid manager

The aim of this questionnaire is to evaluate the Value Impact Estimation (VIE) table as a basis for prioritising IT development features.
The questionnaire takes approximately 15 minutes to complete.
Your participation is entirely voluntary. You can stop the completion of this questionnaire at any time, but only fully completed questionnaires can be analysed. Your participation is very much appreciated.

***1. Name of the project in which Lindsey Brodie co-operated with you:**

Retail IT Services Bid

***2. Prior to this project, have you ever used a method that considers stakeholder values as a basis for decision making?**

○ Yes

[✓ No]

3. Which method did you use that considered stakeholder values?

N/A

4. On a scale of 1 to 10, with 1 being the worst possible method and 10 being the best possible method, how do you rate this previously used method of IT prioritisation?

	worst possible method 1	2	3	4	5	6	7	8	9	best possible method 10
Usefulness	○	○	○	○	○	○	○	○	○	○
Power of clarification	○	○	○	○	○	○	○	○	○	○
Ease of use	○	○	○	○	○	○	○	○	○	○
Effectiveness	○	○	○	○	○	○	○	○	○	○
Time efficiency	○	○	○	○	○	○	○	○	○	○

***5. On a scale of 1 to 10, with 1 being the worst possible method and 10 being the best possible method, how do you rate the Value Impact Estimation (VIE) table as a method of IT prioritisation?**

	worst possible method 1	2	3	4	5	6	7	8	9	best possible method 10
Usefulness	○	○	○	○	○	○	[✓]	○	○	○
Power of clarification	○	○	○	○	○	○	○	○	[✓]	○
Ease of use	○	○	○	○	○	○	[✓]	○	○	○
Effectiveness	○	○	○	○	○	○	[✓]	○	○	○
Time efficiency	○	○	○	○	○	○	[✓]	○	○	○

***6. Below you find a number of statements related to the use of a Value Impact Estimation table (VIE table) in your project. Please choose for each statement, how much you agree or disagree with it.**

	strongly agree	agree	undecided	disagree	strongly disagree
The VIE table captures explicit stakeholder values	○	●	○	○	○
The VIE table supports the mapping of requirements and design solutions to increments.	●	○	○	○	○
The VIE table gives a clear understanding of the different view points and values of different stakeholders involved in the project	○	●	○	○	○
The VIE table gives a clear understanding of the system development costs of each component	○	●	○	○	○
THE VIE table handles performance attributes (i.e.non-functional requirements) well	●	○	○	○	○
The VIE table can cope with large numbers of requirements (scales up well)	○	○	●	○	○
The VIE table allows reprioritization in future increments without further input from stakeholders (unless major changes have taken place)	○	○	●	○	○
The VIE table links the IT development of the project at hand to other IT investments	○	●	○	○	○
The VIE table enables the integration of functional and performance requirements without loss of stakholder value	○	●	○	○	○
The VIE table captures requirements at different levels of refinement	○	●	○	○	○
The VIE table is useless for the prioritisation of the stakeholder values.	○	○	○	○	●
The VIE table captures interdepencies among requirements and among design solutions	●	○	○	○	○

The VIE table shows how major system requirements impact on stakeholder values	●	○	○	○	○
The VIE table supports future prioritisation by allowing the capture of data after the system has become operational	○	●	○	○	○
The VIE table provides data related to return of investment	○	●	○	○	○
The VIE table is overly complex	○	○	○	○	●

7. Please list below the three best aspects of the VIE table (in no particular order)

Good Aspect 1	VIE supports decisions based on quantitative as well as qualitative considerations, while other ad hoc methods are more qualitative.
Good Aspect 2	Creation of VIE is simple (uses standard office productivity tool, Excel)
Good Aspect 3	Content of VIE table maps well to types of information and data used in defining IT offering

8. Please list below the three worst aspects of the VIE table (in no particular order)

Poor Aspect 1	For a first time user, aspects of using VIE tables, are not intuitive, e.g. whether completing all cells is mandatory.
Poor Aspect 2	Manipulation of table is fiddly, e.g. rearranging rows and columns, viewing/printing a complete table
Poor Aspect 3	For a first time user it is difficult to understand, the conceptual and methodological framework within which the VIE is used.

*9. I intend to use the VIE table in other projects

[✓ yes]

no

10. Do you have any other comments, questions, or concerns regarding the VIE table?

I believe the case study demonstrates that the VIE table provides an effective means of analysing and understanding complex environments, such as the definition of an IT solution within the bidding process. By providing a top level, end to end description in client language of an offering the VIE simplifies complexity and supports decision making

This is the end of the questionnaire. I would like to thank you for your kind co-operation. If you are interested in the findings of this study, please email me at l.brodie@mdx.ac.uk to request the summary of the findings. These will be available in October 2014.

Appendix C

List of Publications

C.1 Papers

Evans, C., Brodie, L., and Augusto, J. C. (2014) 'Requirements Engineering for Intelligent Environments', *International Conference on Intelligent Environments (IE).* Shanghai, 30 June 2014 to 4 July 2014. Washington DC*:* IEEE Computer Society Press, pp. 154-161.

Gilb, T. and Brodie, L. (2012) 'What's fundamentally wrong? Improving our approach towards capturing value in requirements specification', *Proceedings of the 22nd Annual* INCOSE *International Symposium* (IS 2012). Rome, Italy, 9–12 July 2012. California: International Council on Systems Engineering (INCOSE), 1, p.1010.

Brodie, L. and Woodman, M. (2012) 'Towards understanding multi-dimensional stakeholder value using 'real world' metrics', 4th International Conference, (SWQD 2012), Vienna, Austria, 17-19 January 2012.

Brodie, L. and Woodman, M. (2011) 'Prioritization of stakeholder value using metrics', in Maciaszek, L. and Loucopoulos, P. (eds.), *5th International Conference on the Evaluation of Novel Approaches to Software Engineering*

(ENASE 2010). Athens, Greece, 22–24 July 2010. Berlin: Springer-Verlag, pp.76–78.

Brodie, L. and Woodman, M. (2009) 'Using metrics to express quality, *17th International Conference on Software Quality Management (SQM 2009).* Southampton, 6-8 April 2009.

Brodie, L. and Woodman, M. (2008) 'Towards a rational prioritization process for incremental and iterative systems engineering', in Nistazakis, M. (ed), *1st International Workshop on Requirements Analysis (IWRA 2008)*. London, England, 6–7 December 2008. London: Pearson. ISBN: 978-1-84776-663-2.

C.2 **Books**

Dalcher, D. and Brodie, L. (2007) *Successful IT projects*. London: Cengage Learning EMEA.

C.3 **Articles**

Gilb, T. and Brodie, L. (2011) 'Agile Aspects of Planguage for Cost-Effective Engineering', *Agile Record*, 5, (January 2011), pp. 5–9. Available at: http://www.agilerecord.com/agilerecord_05.pdf (Accessed: 30 November 2014).

Gilb, T. and Brodie, L. (2010) 'Values for Value', *Agile Record*, 4, (October 2010), pp. 14–22. Available at: http://agilerecord.com/agilerecord_04.pdf (Accessed: 30 November 2014).

Gilb, T. and Brodie, L. (2009) Preventative software quality control: using human checking to change defective human practice. In Dalcher, D. and Fernandez-Sanz (eds), *Experiences and Advances in Software Quality*, Upgrade, X(5, October 2009), pp. 6–13. Available at: http://www.cepis.org/upgrade/files/V-2009-dalcher.pdf (Accessed: 30 November 2014).

Gilb, T. and Brodie, L. (2007) What’s wrong with agile methods: some principles and values to encourage quantification. *Methods and Tools*, Summer 2007. Available at: http://www.methodsandtools.com/archive/archive.php?id=58 (Accessed: 30 November 2014). Also published as Chapter 3 in: Stamelos, I. G. and Sfetsos, P. (eds), (2007) *Agile Software Development Quality Assurance*. London: Information Science Reference, pp.56–69. ISBN: 9781599042169.

Appendix D :

Selection of relevant publications

Gilb, T. and Brodie, L. (2012) 'What's fundamentally wrong? Improving our approach towards capturing value in requirements specification', *Proceedings of the 22nd Annual* INCOSE *International Symposium* (IS 2012). Rome, Italy, 9–12 July 2012. California: International Council on Systems Engineering (INCOSE), 1, pp.1010.

Brodie, L. and Woodman, M. (2012) 'Towards understanding multi-dimensional stakeholder value using 'real world' metrics', 4th International Conference, (SWQD 2012), Vienna, Austria, 17-19 January 2012.

Brodie, L. and Woodman, M. (2011) 'Prioritization of stakeholder value using metrics', in Maciaszek, L. and Loucopoulos, P. (eds.), *5th International Conference on the Evaluation of Novel Approaches to Software Engineering (ENASE 2010).* Athens, Greece, 22–24 July 2010. Berlin: Springer-Verlag, pp.76–78.

Brodie, L. and Woodman, M. (2009) 'Using Metrics to Express Quality, *17th International Conference on Software Quality Management (SQM 2009).* Southampton, 6-8 April 2009.

Brodie, L. and Woodman, M. (2008) 'Towards a rational prioritization process for incremental and iterative systems engineering', in Nistazakis, M. (ed), *1st International Workshop on Requirements Analysis (IWRA 2008)*. London, England, 6–7 December 2008. London: Pearson. ISBN: 978-1-84776-663-2.

What's fundamentally wrong? Improving our approach towards capturing value in requirements specification

TOM GILB
Independent Consultant
Email: Tom@Gilb.com

LINDSEY BRODIE
Middlesex University
Email: L.Brodie@mdx.ac.uk

Abstract. We are all aware that many of our IT projects fail and disappoint: the poor state of requirements practice is frequently stated as a contributing factor. This article proposes a fundamental cause is that we think like programmers, not engineers and managers. We fail to concentrate on value delivery, and instead focus on functions, on use-cases and on code delivery. Our requirements specification practices fail to adequately address capturing value-related information. Compounding this problem, senior management is not taking its responsibility to make things better: managers are not effectively communicating about value and demanding value delivery. This article outlines some practical suggestions aimed at tackling these problems and improving the quality of requirements specification.

Keywords: Requirements; Value Delivery; Requirements Definition; Requirements Specification

Introduction

We know many of our IT projects fail and disappoint, and that the overall picture is not dramatically improving (Carper 2008; Standish Group 2009). We also know that the poor state of requirements practice is frequently stated as one of the contributing failure factors (McManus and Wood-Harper (2008; Yardley 2002). However, maybe a more fundamental cause can be proposed? A cause, which to date has received little recognition, and that certainly fails to be addressed by many well known and widely taught methods. What is this fundamental cause? In a nutshell: that we think like programmers, and not as engineers and managers. In other words, we do not concentrate on value delivery, but instead focus on functions, on use cases and on code delivery. As a result, we pay too little attention to capturing value and value-related information in our requirements specifications. We fail to capture the information that allows us to adequately consider priorities, and engineer and manage stakeholder-valued solutions.

This article outlines some practical suggestions aimed at tackling these problems and improving the quality of requirements specification. It focuses on 'raising the bar' for communicating about value within our requirements. Of course, there is much still to be learnt about specifying value, but we can make a start – and achieve substantial improvement in IT project delivery – by applying what is already known to be good practice.

Note there is little that is new in what follows, and much of what is said can be simply regarded as commonsense. However, since IT projects continue not to grasp the significance of the approach advocated, and as there are people who have yet to encounter this way of thinking, it is worth repeating!

Definition of Value

The whole point of a project is achieving ‘realized value’ (also known as ‘benefits’), for the stakeholders: it is not the defined functionality, and not the user stories that actually count. Value can be defined as ‘the benefit we think we get from something’ (Gilb 2005, 435). See Figure 1.

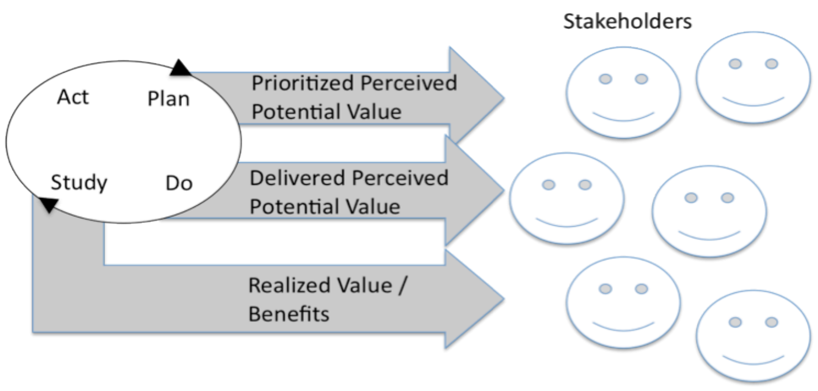

Stakeholders
Act
Plan
Study
Do
Prioritized Perceived Potential Value
Delivered Perceived Potential Value
Realized Value / Benefits

Figure 1. The INCOSE Value can be delivered gradually to stakeholders. Different stakeholders will perceive different value

Notice the subtle distinction between initially perceived value (‘I think that would be useful’), and realized value: effective and factual value (‘this was in practice more valuable than we thought it would be, because …’). Realized value has dependencies on the stakeholders actually utilizing a project’s deliverables.

The issue with much of the conventional requirements thinking is that it is not closely enough coupled with ‘value’. IT business analysts frequently fail to gather the information supporting a more precise understanding and/or the calculation of value. Moreover, the business people when stating their requirements frequently fail to justify them using value. The danger if requirements are not closely tied to value is that we lack the basic information allowing us to engineer and prioritize implementation to achieve value delivery, and we risk failure to deliver the required expected value, even if the ‘requirements’ are satisfied.

It is worth pointing out that ‘value’ is multi-dimensional. A given requirement can have financial value, environmental value, competitive advantage value, architectural value, as well as many other dimensions of value. Certainly value requires much more explicit definition than the priority groups used by MoSCoW (‘Must Have’, ‘Should Have’, ‘Could Have’, and ‘Would like to Have/Won’t Have This Time’) (Stapleton 2003) or by the Planning Game (‘Essential’, ‘Less Essential’ and ‘Nice To Have’) (Cohn 2004) for prioritizing requirements. Further, for an IT project, engineering ‘value’ also involves consideration of not just the requirements, but also the optional designs and the resources available: tradeoffs are needed. However, these are topics for future articles, this article focuses on the initial improvements needed in requirements specification to start to move towards value thinking.

Definition of Requirement

Do we all have a shared notion of what a ‘requirement’ is? This is another of our problems. Everybody has an opinion, and many of the opinions about the meaning of the concept ‘requirement’ are at variance: few of the popular definitions are correct or useful - especially when you consider the concept of ‘value’ alongside them. We have decided to define a requirement as a “stakeholder-valued end state”. You possibly will not accept, or use this definition yet, but we have chosen it to emphasize the ‘point’ of IT systems engineering.

In previous work, we have identified, and defined a large number of requirement concepts (Gilb 2005, 321-438). A sample of these concepts is given in Figure 2. You can use these concepts and the notion of a “stakeholder-valued end state” to re-examine your current requirements specifications. In the rest of this article, we provide more detailed discussion about some of the key points (the “key principles”) you should consider.

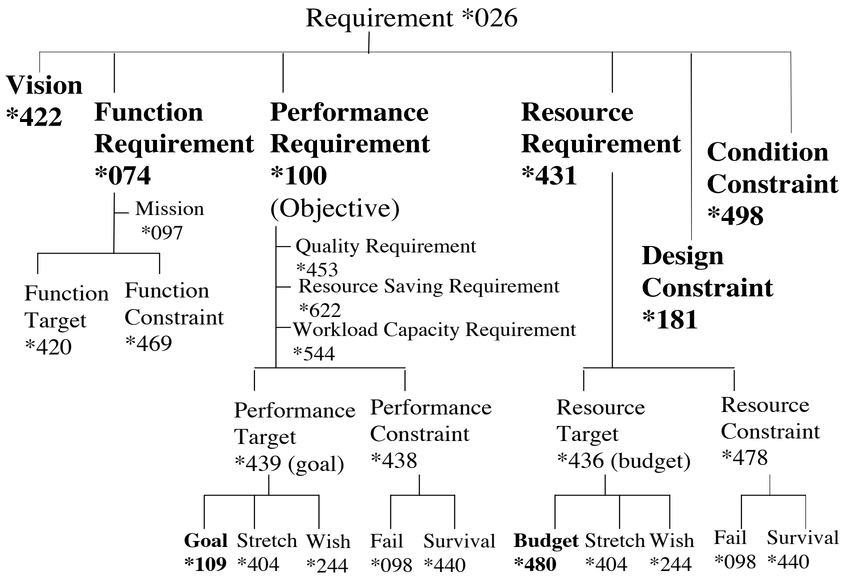

Requirement *026
Vision *422
Function Requirement *074
Performance Requirement *100 (Objective)
Resource Requirement *431
Condition Constraint *498
Design Constraint *181
Mission *097
Function Target *420
Function Constraint *469
Quality Requirement *453
Resource Saving Requirement *622
Workload Capacity Requirement *544
Performance Target *439 (goal)
Performance Constraint *438
Resource Target *436 (budget)
Resource Constraint *478
Goal *109
Stretch *404
Wish *244
Fail *098
Survival *440
Budget *480
Stretch *404
Wish *244
Fail *098
Survival *440

Figure 2. Example of Planguage requirements concepts

The Key Principles

The key principles are summarized in Figure 3. Let’s now examine these principles in more detail. *Note, unless otherwise specified, further details on all aspects of Planguage (a planning language developed by one of the authors, Tom Gilb) can be found in (Gilb 2005).*

Ten Key Principles for Successful Requirements

1. Understand the top level critical objectives
2. Think stakeholders: not just users and customers!
3. Focus on the required system quality, not just its functionality
4. Quantify quality requirements as a basis for software engineering
5. Don’t mix ends and means
6. Capture explicit information about value
7. Ensure there is ‘rich specification’: requirement specifications need far more information than the requirement itself!
8. Carry out specification quality control (SQC)
9. Consider the total lifecycle and apply systems-thinking - not just a focus on software
10. Recognize that requirements change: use feedback and update requirements as necessary

Figure 3. Ten Key Principles for Successful Requirements

Principle 1. Understand the top-level critical objectives. The ‘worst requirement sin of all’ is found in almost all the IT projects we look at, and this applies internationally. Time and again, the high-level requirements – also known as the top-level critical objectives (the ones that fund the project), are vaguely stated, and ignored by the project team. Such requirements frequently look like the example given in Figure 4 (which has been slightly edited to retain anonymity). These requirements are for a real project that ran for eight years and cost over 100 million US dollars. The project failed to deliver any of them. However, the main problem is that these are not top-level critical objectives: they fail to explain in sufficient detail what the business is trying to achieve: there are no real pointers to indicate the business aims and priorities. There are additional problems as well that will be discussed further later (such as lack of quantification, mixing optional designs into the requirements, and insufficient background description).

Example of Initial Weak Top-Level Critical Objectives

1. Central to the corporation’s business strategy is to be the world’s premier integrated <domain> service provider
2. Will provide a much more efficient user experience
3. Dramatically scale back the time frequently needed after the last data is acquired to time align, depth correct, splice, merge, recomputed and/or do whatever else is needed to generate the desired products
4. Make the system much easier to understand and use than has been the case with the previous system
5. A primary goal is to provide a much more productive system development environment then was previously the case
6. Will provide a richer set of functionality for supporting next generation logging tools and applications
7. Robustness is an essential system requirement
8. Major improvements in data quality over current practices

Figure 4. Example of Initial Weak Top Level Critical Objectives

Management at the CEO, CTO and CIO level did not take the trouble to clarify these critical objectives. In fact, the CIO told me that the CEO actively rejected the idea of clarification! So management lost control of the project at the very beginning. Further, none of the technical ‘experts’ reacted to the situation. They happily spent $100 million on all the many suggested architecture solutions that were mixed in with the objectives.

It actually took less than an hour to rewrite one of these objectives, “Robustness”, so that it was clear, measurable, and quantified (see later). So in one day’s work the project could have clarified the objectives, and perhaps avoided some of the eight years of wasted time and effort.

Principle 2. Think stakeholders: not just users and customers! Too many requirements specifications limit their scope to being too narrowly focused on user or customer needs. The broader area of stakeholder needs and values should be considered, where a ‘stakeholder’ is anyone or anything that has an interest in the system (Gilb 2005, 420). It is not just the users and customers that must be considered: IT development, IT maintenance, senior management, operational management, regulators, government, as well as other stakeholders can matter. The different stakeholders will have different viewpoints on the requirements and their associated value. Further, the stakeholders will be “experts” in different areas of the requirements. These different viewpoints will potentially lead to differences in opinion over the implementation priorities.

Principle 3. Focus on the required system quality, not just its functionality. Far too much attention is paid to what the system must do (function) and far too little attention is given to how well it should do it (qualities). Many requirements specifications consist of detailed explanation of the functionality with only brief description of the required system quality. This is in spite of the fact that quality improvements tend to be the major drivers for new projects.

In contrast, here's an example, the Confirmit case study (Johansen and Gilb 2006), where the focus of the project was not on functionality, but on driving up the system quality. By focusing on the "Usability" and "Performance" quality requirements the project achieved a great deal! See Table 1.

Table 1. Extract from Confirmit Case Study [8]

Description of requirement/work task	**Past**	**Current Status**
Usability.Productivity: Time for the system to generate a survey	7200 sec	15 sec
Usability.Productivity: Time to set up a typical market research report	65 min	20 min
Usability.Productivity: Time to grant a set of end-users access to a report set and distribute report login information	80 min	5 min
Usability.Intuitiveness: The time in minutes it takes a medium-experienced programmer to define a complete and correct data transfer definition with Confirmit Web Services without any user documentation or any other aid	15 min	5 min
Performance.Runtime.Concurrency: Maximum number of simultaneous respondents executing a survey with a click rate of 20 sec and a response time < 500ms given a defined [Survey Complexity] and a defined [Server Configuration, Typical]	250 users	6000

By system quality we mean all the "-ilities" and other qualities that a system can express. Some system developers limit system quality to referring to bug levels in code. However, a broader definition should be used. System qualities include availability, usability, portability, and any other quality that a stakeholder is interested in, like intuitiveness or robustness. See Figure 5, which shows a set of quality requirements. It also shows the notion that resources are "input" or used by a function, which in turn "outputs" or expresses system qualities. Sometimes the system qualities are mis-termed "non-functional requirements (NFRs)", but as can be seen in this figure, the system qualities are completely linked to the system functionality. In fact, different parts of the system functionality are likely to require different system qualities.

Principle 4. Quantify quality requirements as a basis for software engineering. Frequently we fail to practice "software engineering" in the sense of real engineering as described by engineering professors, like Koen (2003). All too often quality requirements specifications consist merely of words. No numbers, just nice sounding words; good enough to fool managers into spending millions for nothing (for example, "a much more efficient user experience").

We seem to almost totally avoid the practice of quantifying qualities. Yet we need quantification in order to make the quality requirements clearly understood, and also to lay the basis for measuring and tracking our progress in improvement towards meeting them. Further, it is the quantification that is the key to a better understanding of cost and value – different levels of quality have different associated cost and value.

The key idea for quantification is to define, or reuse a definition, of a scale of measure.

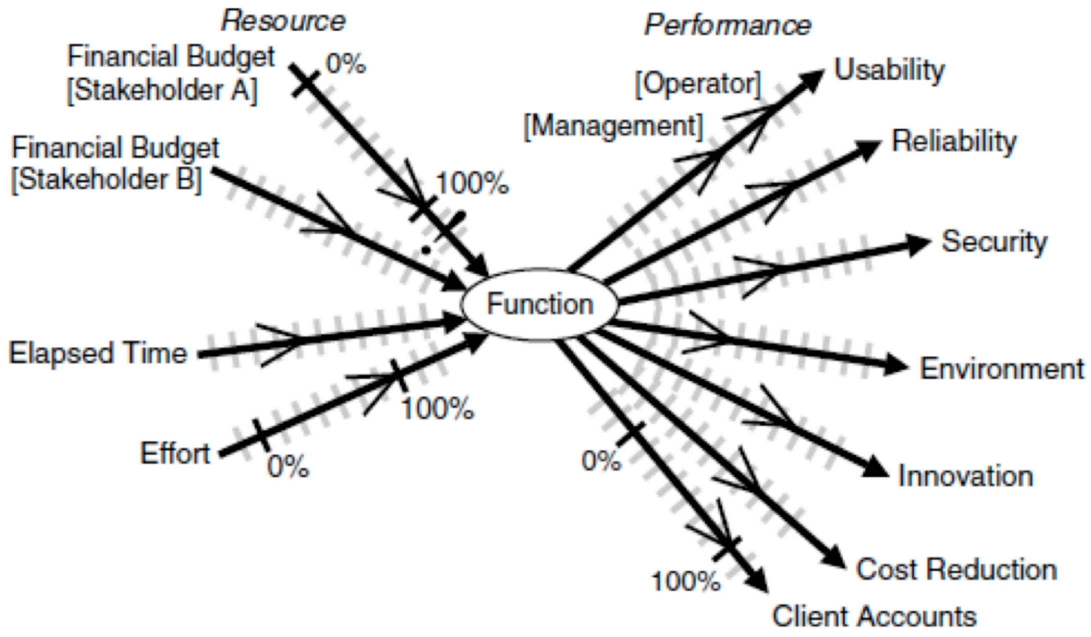


Figure 5. A way of visualizing qualities in relation to function and cost. Qualities and costs are scalar variables, so we can define scales of measure in order to discuss them numerically. The arrows on the scale arrows represent interesting points, such as the requirement levels. The requirement is not 'security' as such, but a defined, and testable degree of security (Gilb 2005, 163)

For example, for a quality "Intuitiveness", a sub-component of "Usability", see Figure 6.

Usability.Intuitiveness:
Type: Marketing Product Quality Requirement.
Ambition: Any potential user, any age, can immediately discover and correctly use all functions of the product, without training, help from friends, or external documentation.
Scale: % chance that defined [User] can successfully complete defined [Tasks] <immediately> with no external help.
Meter: Consumer reports tests all tasks for all defined user types, and gives public report.
Goal [Market = USA, User = Seniors, Product = New Version, Task = Photo Tasks Set, When = 2012]: 80% ±10% <- Draft Marketing Plan.

Figure 6. A simple example of quantifying a quality requirement, 'Intuitiveness'

To give some explanation of the key quantification features in Figure 6:
1. Ambition is a high-level summary of the requirement. One that is easy to agree to, and understand roughly.
2. Scale is the formal definition of our chosen scale of measure. The parameters [User] and [Task] allow us to generalize here, while becoming more specific in detail below (see later). They also encourage and permit the reuse of the Scale, as a sort of 'pattern'.
3. Meter provides a defined measuring process. There can be more than one for different occasions.
4. Goal is one of many possible requirement levels (see earlier detail in Figure 2 for some others: Stretch, Wish, Fail and Survival). We are defining a stakeholder-valued future state

(for example: 80% ± 10%).

One *stakeholder* is 'USA Seniors'. The *future* is 2012. The requirement level type, Goal, is defined as a very high priority, budgeted promise of delivery. It is of higher priority than a Stretch or Wish level. Note other priorities may conflict and prevent this particular requirement from being delivered in practice.

If you know the *conventional* state of requirements methods, then you will now, from this example alone, begin to appreciate the difference proposed by such quantification - especially for *quality* requirements. IT projects already quantify time, cost,, response time, burn rate, and bug density – but there is much *more to achieve system engineering*!

Here is another example of quantification (see Figure 7). It is the initial stage of the rewrite of Robustness from the Figure 4 example. First we determined that Robustness is complex and composed of many different attributes, such as Testability.

Robustness:
Type: *Complex* Product Quality Requirement.
Includes: {Software Downtime, Restore Speed, Testability, Fault Prevention Capability, Fault Isolation Capability, Fault Analysis Capability, Hardware Debugging Capability}.

Figure 7. Definition of a complex quality requirement, Robustness

Then we defined Testability in more detail (see Figure 8).

Testability:
Type: Software Quality Requirement.
Version: Oct 20, 2006.
Status: Draft.
Stakeholder: {Operator, Tester}.
Ambition: Rapid duration automatic testing of <critical complex tests> with extreme operator setup and initiation.
Scale: The duration of a defined [Volume] of testing or a defined [Type of Testing] by a defined [Skill Level] of system operator under defined [Operating Conditions].
Goal [All Customer Use, Volume = 1,000,000 data items, Type of Testing = WireXXXX vs. DXX, Skill Level = First Time Novice, Operating Conditions = Field]: < 10 minutes.
Design: Tool simulators, reverse cracking tool, generation of simulated telemetry frames entirely in software, application specific sophistication for drilling – recorded mode simulation by playing back the dump file, application test harness console <- 6.2.1 HFS.

Figure 8. Quantitative definition of Testability, an attribute of Robustness

Note this example shows the notion of there being different levels of requirements. Principle 1 also has relevance here as it is concerned with top-level objectives (requirements). The different levels that can be identified include: corporate requirements, the top-level critical few project or product requirements, system requirements and software requirements. We need to clearly document the level and the interactions amongst these requirements.

An additional notion is that of 'sets of requirements'. Any given stakeholder is likely to have a set of requirements rather than just an isolated single requirement. In fact, achieving value could depend on meeting an entire set of requirements.

Principle 5. Don't mix ends and means

"Perfection of means and confusion of ends seem to characterize our age." Albert Einstein. 1879-1955

The problem of confusing ends and means is clearly an old one, and deeply rooted. We specify a solution, design and/or architecture, instead of what we really value – our real requirement. There are explanatory reasons for this – for example solutions are more concrete, and what we want (qualities) are more abstract for us (because we have not yet learned to make them measurable).

The problems occur when we do confuse them: if we do specify the means, and not our true ends. As the saying goes: "Be careful what you ask for, you might just get it" (unknown source). The problems include:

- You might not get what you *really* want
- The solution you have specified might *cost too much* or have bad *side effects*, even if you do get what you want
- There may be much *better solutions* you don't know about yet.

So how to we find the 'right requirement', the 'real requirement' (Gilb YYYY) that is being 'masked' by the solution? *Assume* that there probably is a better formulation, which is a more accurate expression of our real values and needs. Search for it by asking 'Why?' Why do I want X, it is because I really want Y, and assume I will get it through X. But, then why do I want Y? Because I really want Z and assume that is the best way to get X. Continue the process until it seems reasonable to stop. This is a slight variation on the '5 Whys' technique (Ohno 1998), which is normally used to identify root causes of problems (rather than high-level requirements).

Assume that our stakeholders will *usually* state their values in terms of some perceived means to get what they really value. Help them to identify (The 5 Whys?) and to acknowledge what they really want, and make that the 'official' requirement. Don't insult them by telling them that they don't know what they want. But explain that you will help them more-certainly get what they more deeply want, with better and cheaper solutions, perhaps new technology, if they will go through the '5 Whys?' process with you. See Figure 9.

Why do you require a 'password'? For Security!
What kind of security do you want? Against stolen information.
What level of strength of security against stolen information are you willing to pay for? At least a 99% chance that hackers cannot break in within 1 hour of trying! Whatever that level costs up to €1 million.
So that is your real requirement? Yep.
Can we make that the official requirement, and leave the security design to both our security experts, and leave it to proof by measurement to decide what is really the right design? Of course!
The aim being that whatever technology we choose, it gets you the 99%?
Sure, thanks for helping me articulate that!

Figure 9. Example of the requirement, not the design feature, being the real requirement

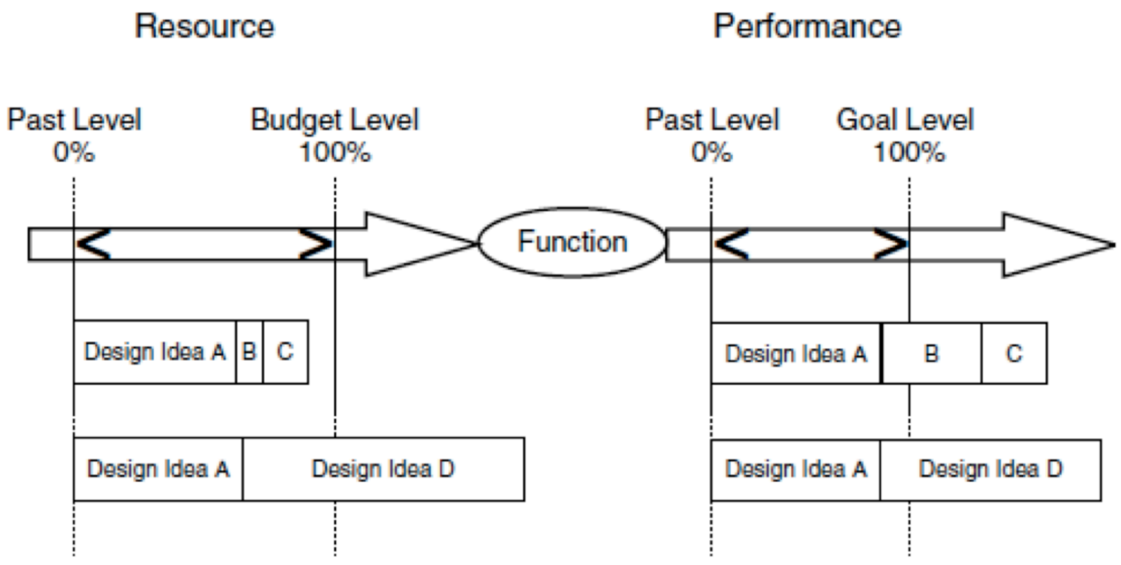


Figure 10. A graphical way of understanding performance attributes (which include all qualities) in relation to function, design and resources. Design ideas cost some resources, and design ideas deliver performance (including system qualities) for given functions. Source (Gilb 2005, 192)

Note that this separation of designs from the requirements does not mean that you ignore the solutions/designs/architecture when software engineering. It is just that you must separate your requirements - including any mandatory means - from any optional means. The key thing is to understand what is optional so that you consider alternative solutions. See Figure 10, which shows two alternative solutions: Design A with Designs B and C, or Design A with Design D. Assuming that say, Design B was mandatory, could distort your project planning.

Principle 6. Capture explicit information about value. How can we articulate and document notions of value in a requirement specification? See the example for Intuitiveness, a component quality of Usability, given in Figure 11, which expands on Figure 6.

Usability.Intuitiveness:
Type: Marketing Product Requirement.
Stakeholders: {Marketing Director, Support Manager, Training Center}.
Impacts: {Product Sales, Support Costs, Training Effort, Documentation Design}.
Supports: Corporate Quality Policy 2.3.
Ambition: Any potential user, any age, can immediately discover and correctly use all functions of the product, without training, help from friends, or external documentation.
Scale: % chance that a defined [User] can successfully complete the defined [Tasks] <immediately>, with no external help.
Meter: Consumer Reports tests all tasks for all defined user types, and gives public report.
-------------------------------- Analysis --------------------------------------
Trend [Market = Asia, User = {Teenager, Early Adopters}, Product = Main Competitor, Projection = 2013]: 95%±3% <- Market Analysis.
Past [Market = USA, User = Seniors, Product = Old Version, Task = Photo Tasks Set, When = 2010]: 70% ±10% <- Our Labs Measures.
Record [Market = Finland, User = {Android Mobile Phone, Teenagers}, Task = Phone+SMS Task Set, Record Set = January 2010]: 98% ±1% <- Secret Report.
----------------------------- Our Product Plans -------------------------------
Goal [Market = USA, User = Seniors, Product = New Version, Task = Photo Tasks Set, When = 2012]: 80% ±10% <- Draft Marketing Plan.

Value [Market =USA, User = Seniors, Product = New Version, Task = Photo Tasks Set, Time Period = 2012]: 2M USD.
Tolerable [Market = Asia, User = {Teenager, Early Adopters}, Product = Our New Version, Deadline = 2013]: 97%±3% <- Marketing Director Speech.
Fail [Market = Finland, User = {Android Mobile Phone, Teenagers}, Task = Phone+SMS Task Set, Product Release 9.0]: Less Than 95%.
Value [Market = Finland, User = {Android Mobile Phone, Teenagers}, Task = Phone+SMS Task Set, Time Period = 2013]: 30K USD.

Figure 11. A fictitious Planguage example, designed to display ways of making the value of a requirement clear

For brevity, a detailed explanation is not given here. Hopefully, the Planguage specification is reasonably understandable without detailed explanation. For example, the Goal statement (80%) specifies which market ("USA") and users ("Seniors") it is intended for, which set of tasks are valued (the "Photo Tasks Set"), and when it would be valuable to get it delivered ("2012"). This 'qualifier' information in all the statements, helps document where, who, what, and when the quality level applies. The additional Value parameter specifies the perceived value of achieving 100% of the requirement. Of course, more could be said about value and its specification, this is merely a 'wake-up call' that explicit value needs to be captured within requirements. It is better than the more common specifications of the Usability requirement, that we often see, such as: "The product will be more user-friendly, using Windows".

So who is going to make these value statements in requirements specifications? I don't expect developers to care much about value statements. Their job is to deliver the requirement levels that someone else has determined are valued. Deciding what sets of requirements are valuable is a Product Owner (Scrum) or Marketing Management function. Certainly, the IT staff should only determine the value related to IT stakeholder requirements!

Principle 7. Ensure there is 'rich specification': requirement specifications need far more information than the requirement itself! Far too much emphasis is often placed on the requirement itself; and far too little concurrent information is gathered about its background, for example: who wants this requirement and when? The requirement itself might be less than 10% of a complete requirement specification that includes the background information. It should be a corporate standard to specify this related background information, and to ensure it is intimately and immediately tied into the requirement itself.

Such background information is *useful* related information, but is not *central* (*core)* to the implementation, and nor is it commentary. The central information includes: Scale, Meter, Goal, Definition and Constraint.

Background specification includes: benchmarks {Past, Record, Trend}, Owner, Version, Stakeholders, Gist (brief description), Ambition, Impacts, and Supports. The rationale for background information is as follows:

- To help judge the value of the requirement
- To help prioritize the requirement
- To help understand the risks associated with the requirement
- To help present the requirement in more or less detail for various audiences and different purposes
- To give us help when updating a requirement
- To synchronize the relationships between different but related levels of the requirements
- To assist in quality control of the requirements
- To improve the clarity of the requirement.

Commentary is any detail that probably will not have any economic, quality or effort consequences if it is incorrect, for example, notes and comments.

Reliability:
Type: Performance Quality.
Owner: Quality Director. Author: John Engineer.
Stakeholders: {Users, Shops, Repair Centers}.
Scale: Mean Time Between Failure.
Goal [Users]: 20,000 hours <- Customer Survey, 2004.
Rationale: Anything less would be uncompetitive.
Assumption: Our main competitor does not improve more than 10%.
Issues: New competitors might appear.
Risks: The technology costs to reach this level might be excessive.
Design Suggestion: Triple redundant software and database system.
Goal [Shops]: 30,000 hours <- Quality Director.
Rationale: Customer contract specification.
Assumption: This is technically possible today.
Issues: The necessary technology might cause undesired schedule delays.
Risks: The customer might merge with a competitor chain and leave us to foot the costs for the component parts that they might no longer require.
Design Suggestion: Simplification and reuse of known components.

Figure 12. A requirement specification can be embellished with many background specifications that will help us to understand risks associated with one or more elements of the requirement specification [12].

See Figure 12 for an example, which illustrates the help given by background information regarding risks.

Background information must not be scattered around in different documents and meeting notes. It needs to be directly integrated into a sole master reusable requirement specification object. Otherwise it will not be available when it is needed: it will not be updated, or shown to be inconsistent with emerging improvements in the requirement specification.

See Figure 13 for a requirement template for function specification (Gilb 2005, 106), which hints at the richness possible for background information.

TEMPLATE FOR FUNCTION SPECIFICATION <with hints>
Tag: <Tag name for the function>.
Type: <{Function Specification, Function (Target) Requirement, Function Constraint}>.
============= Basic Information ================================
Version: <Date or other version number>.
Status: <{Draft, SQC Exited, Approved, Rejected}>.
Quality Level: <Maximum remaining major defects/page, sample size, date>.
Owner: <Name the role/email/person responsible for changes and updates to this specification>.
Stakeholders: <Name any stakeholders with an interest in this specification>.
Gist: <Give a 5 to 20 word summary of the nature of this function>.
Description: <Give a detailed, unambiguous description of the function, or a tag reference to someplace where it is detailed. Remember to include definitions of any local terms>.
============= Relationships ==
Supra-functions: <List tag of function/mission, which this function is a part of. A hierarchy of tags, such as A.B.C, is even more illuminating. Note: an alternative way of expressing supra-function is to use Is Part Of>.
Sub-functions: <List the tags of any immediate sub-functions (that is, the next level down), of this function. Note: alternative ways of expressing sub-functions are Includes and Consists Of>.
Is Impacted By: <List the tags of any design ideas or Evo steps delivering, or capable of delivering, this function. The actual function is NOT modified by the design idea, but its presence in the system is, or can be, altered in some way. This is an Impact Estimation table relationship>.
Linked To: <List names or tags of any other system specifications, which this one is related to intimately, in addition to the above specified hierarchical function relations and IE-related links. Note: an alternative way is to

express such a relationship is to use Supports or Is Supported By, as appropriate>.
============= Measurement ==
Test: <Refer to tags of any test plan or/and test cases, which deal with this function>.
============= Priority and Risk Management ==========================
Rationale: < Justify the existence of this function. Why is this function necessary? >.
Value: <Name [Stakeholder, time, place, event>]: <Quantify, or express in words, the value claimed as a result of delivering the requirement>.
Assumptions: <Specify, or refer to tags of any assumptions in connection with this function, which could cause problems if they were not true, or later became invalid>.
Dependencies: <Using text or tags, name anything, which is dependent on this function in any significant way, or which this function itself, is dependent on in any significant way>.
Risks: <List or refer to tags of anything, which could cause malfunction, delay, or negative impacts on plans, requirements and expected results>.
Priority: <Name, using tags, any system elements, which this function can clearly be done after or must clearly be done before. Give any relevant reasons>.
Issues: <State any known issues>.
============= Specific Budgets ======================================
Financial Budget: <Refer to the allocated money for planning and implementation (which includes test) of this function>.

Figure 13. A template for function specification (Gilb 2005, 106)

Principle 8. Carry out specification quality control (SQC). There is far too little quality control of requirements against relevant standards. All requirements specifications ought to pass their quality control checks before they are released for use by the next processes. Initial quality control of requirements specification, where there has been no previous use of specification quality control (SQC) (also known as Inspection), using three simple quality-checking rules ('unambiguous to readers', 'testable' and 'no optional designs present'), typically identifies 80 to 200+ words per 300 words of requirement text as ambiguous or unclear to intended readers! (Gilb 2009)

Principle 9. Consider the total lifecycle and apply systems-thinking - not just a focus on software. If we don't consider the total lifecycle of the system, we risk failing to think about all the things that are necessary prerequisites to actually delivering full value to real stakeholders on time. For example, if we want better maintainability then it has to be designed into the system. If we are really engineering costs, then we need to think about the total operational costs over time. This is much more than just considering the programming aspects.

You must take into account the nature of the system: an exploratory web application doesn't need to same level of software engineering as a real-time banking system!

Principle 10. Recognise that requirements change: use feedback and update requirements as necessary. Ideally requirements must be developed based on on-going feedback from stakeholders, as to their real value. System development methods, such as the agile methods, enable this to occur. Stakeholders can give feedback about their perception of value, based on the realities of actually using the system. The requirements must be evolved based on this realistic experience. The whole process is a 'Plan Do Study Act' Shewhart cyclical learning process involving many complex factors, including factors from outside the system, such as politics, law, international differences, economics, and technology change.

Attempts to fix the requirements in advance of feedback, are typically wasted energy (unless the requirements are completely known upfront, which might be the case in a straightforward system rewrite with no system changes). Committing to fixed requirements specifications in contracts is not realistic.

Who or What Will Change Things?

Everybody talks about requirements, but few people seem to be making progress to enhance the quality of their requirements specifications and improve support for software engineering. Yes, there are internationally competitive businesses, like HP and Intel that have long since improved their practices because of their competitive nature and necessity (Johansen and Gilb 2005; Gilb YYYY). But they are very different from the majority of organizations building software. The vast majority of IT systems development teams we encounter are not highly motivated to learn or practice first class requirements (or anything else!). Neither the managers nor the systems developers seem strongly motivated to improve. The reason is that they get by with, and even get well paid for, failed projects.

The universities certainly do not train IT/computer science students well in requirements, and the business schools also certainly do not train managers about such matters (Hopper and Hopper 2007). The fashion now seems to be to learn oversimplified methods, and/or methods prescribed by some certification or standardization body. Perhaps insurance companies and lawmakers might demand better industry practices, but I fear that even *that* would be corrupted in practice if history is any guide (for example, think of CMMI and the various organizations certified as being at Level 5).

Summary

Current requirements specification practice is often woefully inadequate for today's critical and complex systems. Yet we do know a considerable amount (Not all!) about good practice. The main question is whether your 'requirements' actually capture the true breadth of information that is needed to make a start on engineering value for your stakeholders.

Here are some specific questions for you to ask about your current IT project's requirements specification:

- Do you have a list of top-level critical objectives?
- Do you consider multiple stakeholder viewpoints?
- Do you know the expected stakeholder value to be delivered?
- Have you quantified your top five quality attributes? Are they are testable? What are the current levels for these quality attributes?
- Are there any optional designs in your requirements?
- Can you state the source of each of your requirements?
- What is the quality level of your requirements documentation? That is, the number of major defects remaining per page?
- When are you planning to deliver stakeholder value? To which stakeholders?

If you can't answer these questions with the 'right' answers, then you have work to do! And you might also better understand why your IT project is drifting from delivering its requirements. The good news is that the approach outlined in this article should allow you to focus rapidly on what really matters to your stakeholders: value delivery.

References

Carper, T. 2008. Off-Line and Off-Budget: The Dismal State of Federal Information Technology Planning. *Report Card to the Senate Hearing*. See http://uscpt.net/CPT_InTheNews.aspx [Last Accessed: August 2010].

Cohn, M. 2004. *User Stories Applied: For Agile Software Development*. Addison Wesley. ISBN 0321205685.

Gilb, T. YYYY. Real Requirements. See http://www.gilb.com/tiki-download_file.php?fileId=28

———. YYYY. Rich Requirement Specs: The use of Planguage to clarify requirements. See http://www.gilb.com/tiki-download_file.php?fileId=44

———. 2005. *Competitive Engineering: A Handbook for Systems Engineering, Requirements Engineering, and Software Engineering using Planguage*. Elsevier Butterworth-Heinemann.

———. 2009. "Agile Specification Quality Control". *Testing Experience*, March 2009. Download from www.testingexperience.com/testing experience01_08.pdf [Last Accessed: August 2010].

———. YYYY. Top Level Objectives: A slide collection of case studies. See http://www.gilb.com/tiki-download_file.php?fileId=180

Hopper, K. and W. Hopper. 2007. *The Puritan Gift*. I. B. Taurus and Co. Ltd..

Johansen, T. and T. Gilb. 2005. "From Waterfall to Evolutionary Development (Evo): How we created faster, more user-friendly, more productive software products for a multi-national market". *Proceedings of INCOSE, 2005*. See http://www.gilb.com/tiki-download_file.php?fileId=32

Koen, Dr. W. V. 2003. *Discussion of the Method: Conducting the Engineer's Approach to Problem Solving*. Oxford University Press.

McManus, J. and T. Wood-Harper. 2008. A Study in Project Failure. See http://www.bcs.org/server.php?show=ConWebDoc.19584 [Last Accessed: August 2010].

Ohno, T. 1988. *Toyota production system: beyond large-scale production*. Productivity Press.

Standish Group. 2009. *Chaos Summary*. See http://www.standishgroup.com/newsroom/chaos_2009.php [Last Accessed: August 2010].

Stapleton, J. (Editor). 2003. *DSDM: Business Focused Development* (2nd Edition). Addison Wesley. ISBN 0321112245. First edition published in 1997.

Yardley, D. 2002. *Successful IT Project Delivery*, Addison-Wesley. ISBN 0201756064.

Towards Understanding Multi-dimensional Stakeholder Value Using 'Real World' Metrics

Lindsey Brodie and Mark Woodman

School of Engineering and Information Sciences
Middlesex University
London, UK
L.Brodie@mdx.ac.uk

Abstract. Measurements using 'real world' scales of measure can capture stakeholder value far more explicitly than use of rankings or weightings, or indeed qualitative data. Requirements and stakeholder value captured using 'real world' metrics are less ambiguous, more intelligible and testable. However, very few systems development methods capture and utilize such metrics in their decision-making process. The Planguage method is an exception and its approach to metrics is described using examples from its use in industry. Further, simple extensions to improve Planguage's capture of stakeholder viewpoints and stakeholder value are proposed. The approach taken is that there are multiple stakeholder viewpoints for the stakeholder value associated with a system. In turn, stakeholder value is considered as multi-dimensional and associated with the fundamental system concepts: objectives, requirements, designs, increment deliverables and system contexts. Only by consideration of combinations of all these system concepts can stakeholder value be appropriately assessed.

Keywords: metrics; stakeholder value; Planguage; impact estimation; goal-orientated measurement.

1 Introduction

The lack of adequate theory and methods for handling value within IT has been recognized for several years [1] [2]. Indeed, current IT industry guidance offers little advice on identifying and prioritizing value. For example, SWEBOK (the software engineering book of knowledge) [3] only makes passing reference to prioritization being needed. Meanwhile, this is against a widely accepted background that IT projects are too often failing [4]. While other factors are also likely to have a role in influencing IT project delivery, the issues of identifying where value resides, what designs would best deliver that value, and measurement of value delivery must be seen as being of fairly prime importance: improved consideration of value would provide a stronger foundation for IT project planning and control.

Moreover, additional needs for identifying and measuring value are introduced by the adoption of evolutionary and agile systems development methods [5]. Such methods demand dynamic identification within each increment of where the high value resides in a system and require feedback about the actual value delivered.

Prioritization methods are considered the means by which value is currently assessed in systems development. However, in practice many organizations are not actually using any identifiable prioritization method and are assessing value informally [6]. In this paper, we discuss the shortfalls in identifying value that we observe in the approaches taken by several existing prioritization methods and explain why we consider the use of 'real world' metrics is an essential basis for quantifying value.

We identify Planguage [7], and specifically its impact estimation (IE) table as a practical platform for our research: Planguage's use of 'real world' metrics and its integrated approach across the systems development lifecycle from objectives to deployed deliverables are seen as key. We briefly explain how Planguage captures performance requirements as metrics, and outline the basics of IE. Finally we discuss a simple extension to IE, value impact estimation (VIE), which makes the expression of stakeholder viewpoints and stakeholder value more explicit. We use data from a small case study on a bank loan system to illustrate our points and findings. Our on-going empirical research work continues with the aim of developing a practical framework to help improve the assessment of value in IT projects.

Note the term 'metric' is defined within this paper as numeric data on a defined scale of measure (which is consistent with Gilb's definition [7]). Here we are using the term 'real world metric' (often termed 'metric' within this paper for brevity) to emphasis the use of scales of measure (units of measurement) found within the system that are understood by the stakeholders, and in some cases already used by the stakeholders. This is opposed to use of more indirect rankings or weightings.

The term 'stakeholder' is defined here as any group of people with an interest in the system. They can be identified by role and/or by location. Decisions over how stakeholders are modeled will always be determined by the specific system of interest. A common assumption is often made that conflict over value within any identified stakeholder group is reasonably likely to be rare. The main concerns for this paper with regard stakeholders are that: 1) numerous different stakeholder groups exist for any typical system in industry and 2) that stakeholders have different viewpoints: they have knowledge of different aspects of the system and are impacted by it in different ways. For example, a senior strategy manager, customers and IT maintenance staff will all be interested in system reliability, but they will have different considerations that they take into account (see also later discussion).

Lastly, the terms 'stakeholder value' and 'value' are used interchangeably within this paper. 'Stakeholder value' has the merit of being distinguishable from other unqualified uses of the term 'value'. The term 'values' is avoided as it is ambiguous in English: it is both the plural of value and an expression of what a stakeholder considers as meriting value.

Existing Prioritization Methods

To date, we have identified over 60 different prioritization methods in the literature. A selection of these prioritization methods categorized by subject area is as follows: (Note selection was by amount of coverage in the literature or to give coverage of the prioritization methods by subject area)

- Requirements Prioritization: MoSCoW [9] and Requirements Prioritization Tool (RPT) [10].
- Requirements (and Effort) Prioritization: Cost-Value Approach [11] and Wiegers' Method [6].
- Requirements (and Design) Prioritization: Analytic Hierarchy Process (AHP) [12], House of Quality (HoQ) within Quality Function Deployment (QFD) [13] [14] and Impact Estimation (IE) [7].
- Architecture (Design) Prioritization: Cost Benefit Analysis Method (CBAM) [15].
- Release Planning: Planning Game [16], EVOLVE/EVOLVE* [17] [18] and Requirements Triage [19]
- Financial Prioritization: Incremental Funding Method (IFM) [20].
- Negotiation Prioritization: Quantitative WinWin [21].

The prioritization methods given most coverage in the literature include AHP, QFD, the Cost-Value Approach, and more recently, the Planning Game. However, it is not clear to what extent all the various prioritization methods are used by software development in industry, or indeed how successful they have been [6].

All the existing IT prioritization methods demonstrate problems with the expression of value. Specifically value is:

- Often implicit
- Rarely considered as multi-dimensional
- Frequently limited or aggregated to a single stakeholder viewpoint
- Restricted in its linkage to system concepts, such as requirements, designs, and increment deliverables, as well as lacking due consideration of system context (time, place and event)
- Other problems.

The following sections explain these points in greater detail.

Value is often Implicit

Within prioritization methods, value can be captured using groupings, rankings, weightings or metrics. To give some examples, MoSCoW [9] captures value under the groupings: 'Must Have', 'Should Have', 'Could Have' and 'Would Like to Have, but Won't Have This Time'. The Planning Game [16] also uses groupings ('essential', 'less essential' and 'nice to have'), but also ranks its user stories in order of preference ('1', '2', '3', etc. where there is a stated descending or ascending order of preference). AHP [12] calculates normalized weightings (Expressed very simply, this results in say, values such as '0.15' for a requirement x and '0.30' for requirement y. In which case, requirement y can then be said to be twice as preferred as requirement x). Only IE [7] is found to make use of ‘real world’ metrics (QFD can demand the capture real world metrics, but it fails to use them in its calculations). The benefit being that real world metrics provide a more explicit basis for understanding value, and also support arithmetical calculations, such as return on investment (ROI). To give an example, for a performance requirement of ‘usability’, metrics using ‘real world’ scales of measure such as ‘average number of errors made per 100 transactions’ or ‘average time taken in minutes for a new customer to place an order’ would be used rather than saying that usability was ‘very important’ (grouping) or ranked with a priority of ‘1’ (ranking).

The benefit of ‘real world’ metrics is that numeric data instances of a ‘metric’ can be expressed for defined levels, such as the target levels for goals and budgets, and the constraint levels for system survival and failure. Such ‘real world’ data has the advantages of not only being unambiguous and testable, it also aids communication amongst stakeholders. Further, as we shall outline, it enables more explicit determination of value: if you know the current and targeted levels, you have a basis for discussion (if we reduced the number of errors occurring in a week by 100, how much staff time would that save on average? What if you reduced errors occurring in a week by 200?) If you can estimate the cost of developing the proposed solution(s), then you can calculate potential ROI(s). This contrasts with trying to assess value against ‘very important’ or priority ‘1’.

A further point is that the rationales behind the stakeholders’ choice of value tend not to be captured. So not only is there *implicit* expression of value, which is difficult to correlate to real world value, there is also no explicit information about what the stakeholder was taking into account in expressing value. This especially hinders reprioritizing or auditing value.

Value is Rarely Considered as Multi-Dimensional

Value for a requirement or a design is typically expressed as a single selection or estimate. However, there are different types of value, for example strategic value, financial value, legal value and environmental value. The literature (the main sources include [22] [23] [10] [17] [6] [24] and [25]) identifies over 50 prioritization factors, which we consider all contribute to the various different types (dimensions) of value. We grouped the identified prioritization factors by concept area: opinion, strategy, time, legal, financial, cost, fit, external dependency and risk (see later Table 1 for further detail). We note that several of these concept areas are also identified by Berander and termed by him as “aspects” [26]. We assume these groups/concept areas can be taken to express – as a first-cut – the different types/dimensions of value.

Identifying such types/dimensions of value is a current focus of our empirical research work. While some of the prioritization factors can be reduced to financial terms, it is unlikely all can be. Further, aggregating the financial value of the different types of value would lose information necessary for a prioritization process. That is, while you can translate environmental impact to a financial value, there is merit in considering the environmental value in its own right as well.

Value is Frequently Limited or Aggregated to a Single Stakeholder Viewpoint

Within any system there are numerous stakeholder groups, who have different perspectives on what types of value matter, where value resides, and also the amount of value. As discussed earlier, the stakeholders' areas of expertise and interests in a system differ. Handling of multiple stakeholder viewpoints seems universally poor within existing prioritization methods. It is not clear if any method presents an overview of different stakeholder viewpoints. RPT [10] is perhaps the closest.

In addition, some suspect practices such as weighting stakeholders according to job title/role or averaging across stakeholder's preferences can be found. Such practices distort stakeholder value and/or hide where the value resides.

Value is Often Restricted in its Linkage to System Concepts

Many prioritization methods focus on value associated with one or a small set of system concepts (for example, only using requirements data or only design data). In fact, in the literature the research on prioritization methods tends also to be segregated into subject areas (as identified earlier, see the list of selected prioritization methods), for example considering requirements prioritization or architecture prioritization or release planning. While such research specialization is not necessarily a problem, the key issue is the lack of an integrated approach. This is for two reasons: first, IT projects in industry need some form of prioritization process that works across the system development lifecycle (they are unlikely to use a diverse set of prioritization methods), and secondly, perhaps more importantly, the prioritization process itself can be seen as requiring a wide scope across all system concepts if value is to be adequately addressed. To give a simple, admittedly non-IT example: buying a car. The single stakeholder, requirements-only approach would be to simply go out and buy the car of your dreams (and admittedly if the car was looked after, this could be a long-term investment!) However, a more common reality is that a wider system, multi-stakeholder approach is taken. You consider the stakeholders (for example, other family members, the servicing garage, the government, the environment, and other passengers). You then consider the quality requirements (for example, less petrol consumption, improved reliability, improved safety, improved security, better in-car sound, minimum amount of storage space, cheaper running costs and the more modern appearance of the car). Next you consider the potential cars (designs) that you could buy. Purchase cost is often an immediate consideration and can rule out some designs. Then you investigate each design's impacts on your set of requirements. Further, in some cases, the availability (timescales) for the delivery of the car becomes a consideration: maybe there is a waiting list or your financial planning requires a certain delay. The point being made using this example is that there is a need to balance stakeholder requirements, potential designs and delivery options. When considering these concepts you are thinking about the resulting value across the different stakeholders. In turn, the question becomes why it is acceptable in many IT prioritization methods to just consider a very limited set of system concepts, when in reality we are actually taking into account many more factors?

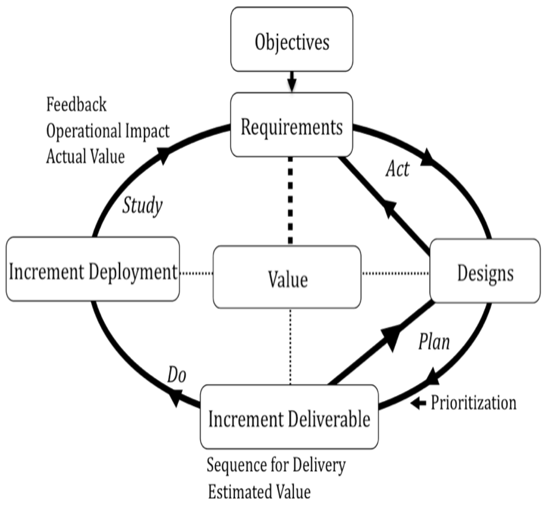


Fig. 1. An increment cycle, which shows the main system concepts. Each increment consists of one complete cycle. The requirements, designs and deliverable have associated value, and after deployment (delivery) the actual benefits (realised value) can be assessed.

Fig. 1 shows the main system concepts within an increment cycle (based on Shewhart's Plan-Do-Study-Act cycle [7]). It shows the ideal positioning for prioritization to decide what design(s) to implement in the increment and shows value linked to the various system concepts.

To conclude this section, we include two tables. Table 1 provides an overview of the existing prioritization methods summarizing their division by subject area, their use of the system concepts and the type of prioritization data that they utilize. From the findings to date, IE emerges as a prioritization method worth further consideration. It not only is the only prioritization method found to actively use metrics, but it also offers better coverage of the system concepts (including handling increment deployment). However, it only offers limited coverage of multiple stakeholder viewpoints and mainly expresses stakeholder value implicitly.

Table 2 brings together the main points discussed so far concerning the potential multiple dimensions for value. For Table 2, we chose to sub-divide the 50+ prioritization factors into three categories by stakeholder role and note the similarity to the choices of Lehtola [27], and Barney, Aurum and Wohlin [22]. For brevity we have chosen what we consider the three main stakeholder roles in a prioritization process: strategy management, systems development and operations management. Clearly there are many more stakeholder roles than these three in a system. However it is impossible to generalize about all the organizational stakeholder roles for a system. The main point being investigated here is whether different stakeholders validly have different viewpoints on a system (see later).

We added further sub-division under four system concepts (used earlier in Fig. 1). We decided that: strategic management has responsibility for the objectives, systems

Table 1. Use of system concepts and types of prioritization data within a selection of prioritization methods.

Key: N = Not supported Y = Yes, supported H = By use of hierarchy V1 = One Viewpoint I = Implicit W = Weightings G = Groupings M = 'Real World' Metrics F = Financial ( ) = Qualified		Subject Area											
		Requirements Prioritization		Requirements & Effort Prioritization		Requirements & Design Prioritization			Architecture (Design) Prioritization	Release Planning			Financial Prioritization
		Prioritization Method											
		MoSCoW	Requirements Prioritization Tool (RPT)	Cost-Value Approach	Wiegers' Method	Analytic Hierarchy Process (AHP)	House of Quality (HoQ) within Quality Function Deployment (QFD)	Impact Estimation (IE)	Cost Benefit Analysis Method (CBAM)	Planning Game	EVOLVE/EVOLVE*	Requirements Triage	Incremental Funding Method (IFM)
System Concept	Objective	N	Y	N	N	H	N	H	N	N	N	N	N
	Functional Requirement	Y	Y	Y	Y	Y	Y	(N)	Y	Y	Y	Y	Y
	Quality Requirement	(N)	N	(N)	N	Y	Y	Y	N	(N)	(N)	(N)	N
	Design	N	N	N	N	Y	Y	Y	Y	N	N	N	(Y)
	Impact of Design	N	N	N	N	Y	Y	Y	Y	N	N	N	N
	Increment Deliverable	N	N	N	N	N	N	Y	N	Y	Y	Y	Y
	Result of Increment Delivery	N	N	N	N	N	N	Y	N	N	N	N	N
Prioritization Data	Stakeholder	V1	Y	V1	V1	V1	V1	V1	V1	V1	V1	V1	V1
	Stakeholder Value	I	I	I	I	I	I	(M)	I	I	I	I	F&I
	Prioritization Technique	G	W	W	W	W	W&G	M	W	GR	G	G&W	W

development owns the requirements and designs, and operations management has responsibility for accepting the planned, and subsequently delivered, increment deliverables. Finally, we divided the prioritization factors into the concept areas (discussed earlier). Note also that these concept areas and prioritization factors are likely not to be complete; this table only reflects the main prioritization factors found in the literature.

It emerges from Table 2 that the prioritization factors can be allocated across system concepts and across stakeholders. This provides limited theoretical evidence of the need to support multiple stakeholder viewpoints and to consider value in relation to a wider set of system concepts.

Table 2. Prioritization factors by stakeholder role & system concept.

STAKEHOLDERS	**Strategic management**	**Systems development**		**Operations management/ customers**
CONCEPTS	**Organizational objectives (objective)**	**Systems requirements (requirement)**	**Design solutions (design)**	**Delivery plans (increment/delivery)**
PRIORITIZATION FACTORS				
OPINION	Vision/intuition/gut feeling/preference/ bias	Preferences/ bias/ importance	Intuition/ preferences/ bias	Preferences/ bias
STRATEGY	Strategic alignment/ business objectives/ product strategy		Long-term Strategy for systems architecture	
	Competition	Quality		
	Customer demand	Originator of requirement		End user value
	New business potential			
TIME	Urgency/time to market/lead time		Time schedule/ time constraints	
	Long term versus short term		Long term versus short term	
LEGAL	Legal mandate/ regulations	Legal mandate/ regulations		
	Contracts in place			
FINANCIAL BENEFIT	Market value/price			
	Financial benefits			
	Financial penalties			
	Benefit/cost ratio			
	Cost of not implementing			
COST	Development costs/ implementation costs/ support costs		Development costs/ support costs	Implementation costs/ support costs
	Operational costs			Operational costs
FIT	Fit with operational context: . business processes . skills/training . delivery timing		Staff competence Balanced workload	Fit with operational context: . business processes . skills/training . delivery timing
			Resource availability/ effort constraints	Resource availability/ effort constraints
	Fit with other products		Change impact/ base code dependencies	Change impact
			Logical implementation order	
			Reuse potential	
EXTERNAL DEPENDENCY	Intermediary channels	External dependencies	External dependencies	
RISK	Business risk Sales barriers	Volatility of requirements	Technical risk in: . current system . proposed system . implementation process	
			Difficulty of implementation/ complexity	Difficulty of implementation/ complexity

Brief Description of Planguage

Having established that IE is a practical platform to use for further research on value, it is appropriate to discuss how Planguage (IE is a method within Planguage) handles metrics, and explain something of the IE method.

Use of Planguage in Industry

Planguage has been developed in industry by Tom Gilb over approximately thirty years. Planguage has been used mainly, but not exclusively, in systems engineering organizations within telecommunications, computer hardware manufacture, and semiconductor chip manufacture. Intel, for example, has over several years trained in excess of six thousand engineers to specify Planguage metrics (Personal Communication). Planguage has also been used in the banking and charity sectors. However, there is little documented in the academic literature about the method. The Confirmit project is one exception [28] [29]. Note Planguage has been used at both the strategic planning level and at project level. A documented example of its use at strategic level, planning product improvement within a computer manufacturer appears in [30]. See also Fig. 2, which shows some examples of Planguage metrics.

Planguage Metrics

Use of metrics is fundamental to Planguage, and adoption of metrics is essential for IE to be used. The basic Planguage template for a performance requirement consists of a set of parameters and is as follows: (Note there are many more parameters than shown here.)

Tag: A short name for the requirement.
Description: A short description for the requirement.
Source: The person or document stating the requirement and its parameter data.
Scale: The scale of measure for measurements taken for the requirement.
Meter: The means by which the levels for this requirement are to be practically measured.
Past: Some current or past benchmark level of the requirement at a stated date.
Goal: Some target level for the requirement at a stated date.
Survival: A constraint level for the requirement that if not met threatens the continued existence of the system.
Value: Some potential value associated with achieving the requirement.

In addition, **[Time, Place, Event]** qualifiers are specified to more accurately identify the scope of applicability of each of the parameters. By default, the current set of parameters and qualifiers apply, until they are overridden.

Headcount: **Scale**: Net employee growth rate as yearly percentage. **Past** [This Year]: 40%. **Goal** [Next Year]: 30%.
User Productivity: **Scale**: Average Percentage of user time lost due to IT-related failures. **Past** [Department A, This Year]: 20%. **Goal** [Department A, Next Year]: 10%.
Languages: **Scale**: Time taken for employees to learn new software languages. **Past**: 5 to 30 days. **Goal**: 5 days.
Customer Satisfaction: **Scale**: Average percentage of defined [customer] defined [query type] considered answered with defined [accuracy] by defined [means]. **Meter** [Cabin staff, customer query]: Ask cabin staff to rate the accuracy of their answers to customer queries.
Maintainability: **Type**: Complex Quality Requirement. **Includes**: {Problem Recognition, Administrative Delay, Tool Collection, Problem Analysis, Change Specification, Quality Control, Modification Implementation, Modification Testing {Unit Testing, Integration Testing, Beta Testing, System Testing}, Recovery}.
Problem Analysis: **Scale**: Average clock time for the assigned defined [Maintenance Instance] to analyze the fault symptoms and be able to begin to formulate a correction hypothesis.
Recovery: **Scale**: Average clock hours for defined [User Type] to return system to the state it was in prior to the fault and, to a state ready to continue with work.
Search Time: **Scale**: Average time in seconds a user with defined [User-experience, default=Normal] takes to find what they (and we) want them to find.
Robustness.Testability: **Scale**: The duration of a defined [Volume] of testing, or a defined [Type], by a defined [Skill Level] of system operator under defined [Operating Conditions].
Speed To Deliver: **Scale**: Average calendar days needed from New Idea approved until Idea Operational, for defined [Tasks], on defined [Markets].

Fig. 2. Some miscellaneous examples of Planguage metrics. Most metrics, but not all, from [7].

Planguage performance requirements can be expressed in a hierarchy. There is a notion of a generic framework, which is modified to a system. See Fig. 3, which shows a performance hierarchy developed for a small case study involving improving a bank loan business process. The Planguage specification for one of the requirements, R1 is as follows: (Note ‘Value Scale’ is an addition to Planguage.)

Performance Requirement: R1: Reduce time for customer to submit request.
Scale: Average time taken for defined [request type: Default = Loan**].**
Past: 30 minutes.
Goal: 10 minutes.
Value Scale: Figure of merit due to data sensitivity. For this case study should have been in Pounds Sterling.
Value [Customer]: 4.

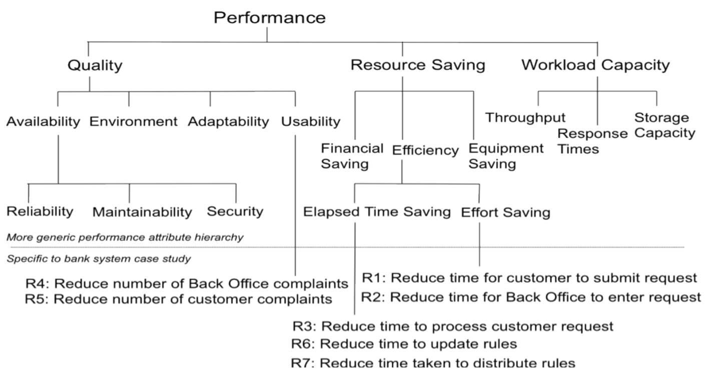
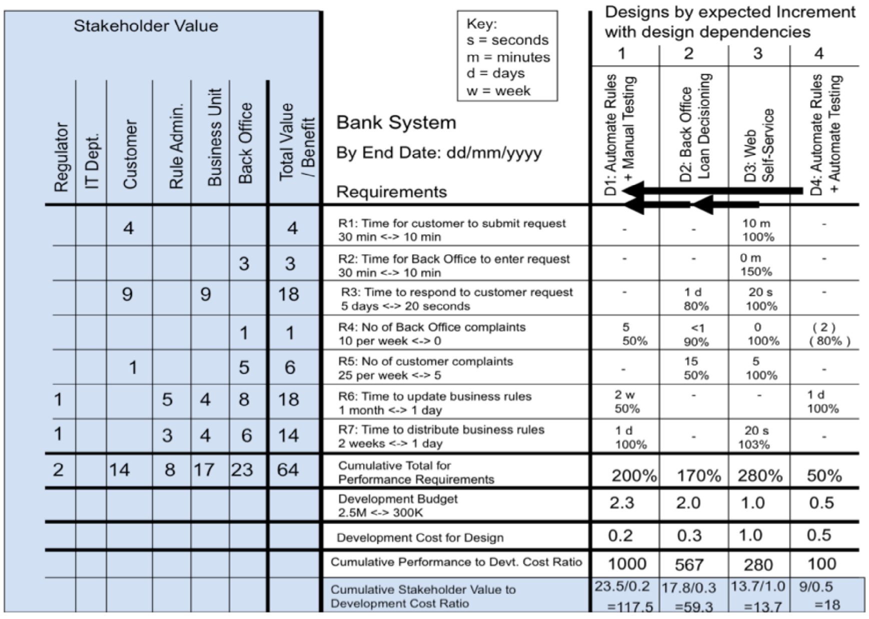

Performance
Quality
Resource Saving
Workload Capacity
Availability
Environment
Adaptability
Usability
Financial Saving
Efficiency
Equipment Saving
Throughput
Response Times
Storage Capacity
Reliability
Maintainability
Security
Elapsed Time Saving
Effort Saving
More generic performance attribute hierarchy
Specific to bank system case study
R4: Reduce number of Back Office complaints
R5: Reduce number of customer complaints
R1: Reduce time for customer to submit request
R2: Reduce time for Back Office to enter request
R3: Reduce time to process customer request
R6: Reduce time to update rules
R7: Reduce time taken to distribute rules

Fig. 3. Performance requirements hierarchy for the bank loan case study.

Impact Estimation (IE)

Requirement, R1 requires that the time taken for a customer to submit a request for a loan should reduce from taking on average 30 minutes to taking only 10 minutes. It is this requirement that is captured in the IE table, along with all the other performance requirements. See Fig. 4.



Stakeholder Value

Key:
s = seconds
m = minutes
d = days
w = week

Bank System
By End Date: dd/mm/yyyy

Designs by expected Increment with design dependencies

Regulator	IT Dept.	Customer	Rule Admin.	Business Unit	Back Office	Total Value / Benefit	Requirements	1 D1: Automate Rules + Manual Testing	2 D2: Back Office Loan Decisioning	3 D3: Web Self-Service	4 D4: Automate Rules + Automate Testing
		4				4	R1: Time for customer to submit request 30 min <-> 10 min	-	-	10 m 100%	-
					3	3	R2: Time for Back Office to enter request 30 min <-> 10 min	-	-	0 m 150%	-
		9		9		18	R3: Time to respond to customer request 5 days <-> 20 seconds	-	1 d 80%	20 s 100%	-
					1	1	R4: No of Back Office complaints 10 per week <-> 0	5 50%	<1 90%	0 100%	(2) (80%)
		1			5	6	R5: No of customer complaints 25 per week <-> 5	-	15 50%	5 100%	-
1			5	4	8	18	R6: Time to update business rules 1 month <-> 1 day	2 w 50%	-	-	1 d 100%
1			3	4	6	14	R7: Time to distribute business rules 2 weeks <-> 1 day	1 d 100%	-	20 s 103%	-
2		14	8	17	23	64	Cumulative Total for Performance Requirements	200%	170%	280%	50%
							Development Budget 2.5M <-> 300K	2.3	2.0	1.0	0.5
							Development Cost for Design	0.2	0.3	1.0	0.5
							Cumulative Performance to Devt. Cost Ratio	1000	567	280	100
							Cumulative Stakeholder Value to Development Cost Ratio	23.5/0.2 =117.5	17.8/0.3 =59.3	13.7/1.0 =13.7	9/0.5 =18

Fig. 4. VIE table for the bank loan case study.

IE involves estimating the impacts of the potential designs on the set of requirements. Each IE table is a snapshot of a system at a stated time assessed against the target required system at a specified future date. For each requirement, the current level is taken as the baseline, and assigned as 0%. In a similar manner, the required target level at the specified date in the future is assigned as 100%. The impact of a design is estimated as how far it will move the required requirement level from 0% towards 100%. By specifying percentages, further simple arithmetic can be done to establish the overall effectiveness of a potential design. By summing all the estimated percentage improvements for a design (and maybe some will be negative), and then dividing by the design's development cost, a performance to cost ratio is obtained. This provides a rough measure of which design is likely to be most effective. Note the percentage improvements delivered by each design above those delivered by the previously implemented designs are used (So for R4, D2 delivers 40% improvement following D1). Also the improvements are all capped at 100% (additional improvement above the target level might be useful, but it is not seen as required). Note once an increment has been deployed, the actual results can be input into the IE table and assessed against the estimated figures.

The VIE Table

As discussed earlier, IE is considered lacking in its support for multiple stakeholder viewpoints and explicit stakeholder value. So to address these problems, we added a simple extension to IE and named the extended method, value impact estimation (VIE). We added stakeholders into the basic IE table on its left-hand side with their associated stakeholder value, see the shaded areas of Fig. 4. The figures in the stakeholder columns represent the estimated stakeholder value on achieving 100% of each requirement.

We input the data for the bank loan case study into a VIE table (see Fig. 4) and found that capturing the stakeholders was useful and that the table assisted discussion of where stakeholder value resided. We identified that transaction volume information should (in future) be added to the VIE table.

Owing to data sensitivity in the case study, the stakeholder value in Fig. 4 is only captured using figures of merit and not the real estimated financial figures. The figures of merit represent the estimated financial gain from increased uptake of loans and the financial savings equivalent to the staff time saved by the business process improvements. Note in Fig. 4 there is no separation of these two different types of value.

In turn, the extension of adding explicit stakeholder value enables cumulative stakeholder value to development cost ratios to be calculated. See the bottom row of Fig. 4. The calculation is worked out here on the basis that an estimated percentage impact of a design on a requirement will result in the same percentage of the stakeholder value of the requirement being achieved. In other words, an assumption is made that the utility curve [31] is linear.

From the results for the value to cost ratios, it appears that maybe the implementation order of the designs, should be altered allowing the design currently positioned for implementation in increment 4 to be carried out in increment 3.

Conclusions

The rationale for adopting IE as a platform for further research into value has been explained. A simple extension to IE which provides more explicit handling of stakeholder viewpoints and stakeholder value has been shown, value impact estimation (VIE). Findings from an initial small case study have been presented confirming initial validity of this approach.

Further empirical research is needed to establish how value should usefully be captured within systems development to ensure organizational value is delivered by IT projects. Our current focus is on establishing the dimensions for value that are useful to assist a prioritization process for value.

References

1. Sullivan, K.: Introduction to the first workshop on the economics of software and computation. In: Companion to the Proceedings of the 29th International Conference on Software Engineering, 125--126. IEEE Computer Society, Washington, DC (2007)
2. Boehm, B.: Value-based software engineering. Software Engineering Notes, 28, 2. ACM (2003)
3. Bourque, P., Dupuis, R. (eds.): SWEBOK: Guide to the Software Engineering Body of Knowledge 2004 Version. IEEE Computer Society (2004)
4. Brown, N.V.: Action to Avoid "IT" Train-wrecks: An Agenda for Change. 2009 Forecast: Report Card on US Federal Projects. Presented to US Senate Subcommittee July 31, 2008. Available from: http://uscpt.net/CPT_InTheNews.aspx [Last Accessed: March 15 2011].
5. Larman, C., Basili, V.R.: Iterative and incremental development: a brief history. Computer, June 2003, 36, 6, 47--56. IEEE Computer Society (2003)
6. Lehtola, L., Kauppinen, M.: Suitability of requirements prioritization methods for market-driven software product development. Software Process Improvement and Practice, February-March 2006, 11, 7--19. John Wiley and Sons, Ltd. (2006)
7. Gilb, T.: Competitive Engineering: A Handbook For Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage. Brodie, L. (ed.). Butterworth-Heinemann, Oxford (2005)
8. Kaposi, A., Myers, M.: Systems, Models and Measures (Formal Approaches to Computing and Information Technology (FACIT)). Springer, Heidelberg (1994)
9. Stapleton, J. (ed.): DSDM: Business Focused Development (2nd Edition), Addison Wesley, London (2003)
10. Moisiadis, F.: The fundamentals of prioritising requirements. In: Systems Engineering, Test and Evaluation Conference (2002)
11. Karlsson, J., Ryan, K.: A Cost-Value Approach for prioritizing requirements. IEEE Software, September/October 1997, 14, 5, 67--74 (1997)
12. Saaty, T.L.: How to make a decision: the analytic hierarchy process. European Journal of Operational Research, 48, 9--26 (1990)

13. Akao, Y.: QFD: past, present, and future. In: International Symposium on QFD '97, Linkoping (1997)
14. Cohen, L.: Quality Function Deployment: How to Make QFD Work for You. Addison Wesley Longman, Reading (1995)
15. Kazman, R., Asundi, J., Klein, M.: Quantifying the costs and benefits of architectural decisions. In: 23rd International Conference on Software Engineering (ICSE 2001), pp. 297–306. IEEE Computer Society, New York (2001)
16. Beck, K.: Extreme Programming Explained: Embrace Change. Addison Wesley, Reading (2000)
17. Greer, D., Ruhe, G.: Software release planning: an evolutionary and iterative approach. Information and Software Technology, 46, 4, pp. 243--253 (2004)
18. Saliu, O., Ruhe, G.: Supporting software release planning decisions for evolving systems. In: 29th Annual IEEE/NASA Software Engineering Workshop (SEW'05). IEEE Press, New York (2005)
19. Davis, A.: The art of requirements triage. IEEE Computer, March 2003, 36, 3, 42--49. (2003)
20. Denne, M., Cleland-Huang, J.: Software by Numbers: Low-Risk, High-Return Development. Prentice-Hall, Upper Saddle River (2004)
21. Ruhe, G., Eberlein, A., Pfahl, D.: Trade-off analysis for requirements selection. International Journal of Software Engineering and Knowledge Engineering, 13, 4, 345--366. (2003)
22. Barney, S., Aurum, A., Wohlin, C.: A product management challenge: creating software product value through requirements selection. J. Systems Architecture, 54, 576--593. Elsevier B. V. (2008)
23. Berander, P., Andrews, A.: Requirements prioritization (Chapter 4), In: Aurum, A., Wohlin, C. (eds.) Engineering and Managing Software Requirements. Springer-Verlag, Heidelberg (2005)
24. Carlshamre, P., Sandahl, K., Lindvall, M., Regnell, B., Natt och Dag, J.: An industrial survey of requirements interdependencies in software product release planning. In: Fifth IEEE International Symposium on Requirements Engineering, Toronto, pp. 84–91. IEEE Press, New York (2001)
25. Firesmith, D.: Prioritizing requirements. J. Object Technology, September-October 2004, 3, 8, 35--47 (2004). www.jot.fm/issues/issue_2004_09/column4/ [Last Accessed: March 15 2011].
26. Berander, P.: Evolving Prioritization for Software Product Management. Blekinge Institute of Technology (2007) Doctoral Dissertation Series, No 2007:07. ISSN 16532090. ISBN 9789172951082. Available via http://www.bth.se/tek/pba.nsf/pages/8c17d1e959f1576dc12571e20031bdbd!OpenDocument / [Last Accessed: March 15 2011].
27. Lehtola, L.: Providing value by prioritizing requirements throughout software product development: state of practice and suitability of prioritization methods, Licentiate thesis, Helsinki University of Technology, Department of Computer Science and Engineering (2006)
28. Johansen, T., Gilb, T.: From Waterfall to Evolutionary Development (Evo): how we rapidly created faster, more user-friendly, and more productive software products for a competitive multi-national market. In: Proceedings of INCOSE (2005)
29. Hanssen, G.K., Faegri, T.E.: Agile customer engagement: a longitudinal qualitative case study. In: 5th ACM-IEEE International Symposium on Empirical Software Engineering (ISESE '06). ACM (2006)
30. Gilb, T.: Principles of Software Engineering Management. Addison Wesley, London (1988)
31. Daniels, J., Werner, P.W., Bahill, A.T.: Quantitative methods for tradeoff analyses. Systems Engineering, 4, 3, 190--212. John Wiley & Sons, Inc. (2001)

Prioritization of Stakeholder Value using Metrics

Lindsey Brodie and Mark Woodman

School of Engineering and Information Sciences, Middlesex University,
The Burroughs, Hendon, London, NW4 4BT, United Kingdom
{L.Brodie, M.Woodman}@mdx.ac.uk

Abstract. Given the reality of resource constraints, software development always involves prioritization to establish what to implement. Iterative and incremental development methods increase the need to support dynamic prioritization to identify high stakeholder value. In this paper we argue that the current prioritization methods fail to appropriately structure the data for stakeholder value. This problem is often compounded by a failure to handle multiple stakeholder viewpoints. We propose an extension to an existing prioritization method, impact estimation, to move towards better capture of explicit stakeholder value and to cater for multiple stakeholders. A key feature is the use of absolute scale data for stakeholder value. We use a small industry case study to evaluate this new approach. Our findings argue that it provides a better basis for supporting priority decision-making over the implementation choices for requirements and designs.

Keywords: Stakeholder value, impact estimation, requirements prioritization, design prioritization, metrics, value-based software engineering.

1 Introduction

Research into prioritization has increased in recent years with many new prioritization methods and variants being put forward. Much has been achieved in identifying the prioritization factors and the issues of concern when structuring prioritization data. However, existing prioritization methods and the prioritization data they utilize (in content and structure) continues to be insufficient to support the type of prioritization process that ideally needs to be adopted. Specifically, progress in improving the prioritization process seems hampered by inadequate conceptualizations of stakeholder value, in particular by the use of implicit notions of value. This is often compounded by an additional failure to support multiple stakeholder viewpoints. Note the term “stakeholder” is used here to mean any group of people with an interest in the system, and they can be identified by role and/or location.

In this paper, to move towards addressing the problems identified above, we propose capturing stakeholder value by stakeholder role, and using absolute scale data (as opposed to using, for example, ordinal scale data) for stakeholder value. We consider the explicit “real world” data captured by using absolute scales normally provides a better basis for supporting priority decision-making. For example, as we shall discuss later, it supports arithmetic calculations such as return on investment (ROI).

To present the argument for our proposals for stakeholder value, this paper is structured in the following way. Section 2 outlines the need for prioritization explaining why the prioritization process is important. Section 3 provides an overview of the existing research on prioritization and analyses how it relates to the problems we perceive impacting the prioritization of stakeholder value. Section 4 then investigates in detail how stakeholder value is currently expressed within the prioritization data and explains some of the resulting weaknesses. Finally, Section 5 briefly describes initial validation of using explicit absolute scale data for stakeholder value: a case study using value impact estimation (VIE). We have developed VIE as a simple extension to an existing method, impact estimation (IE). IE [1] uses absolute scale data and captures the impact of each of the potential designs on each of the requirements. VIE extends this to additionally capture explicit stakeholder value by stakeholder role. Our initial findings are that use of absolute scales is indeed beneficial for capturing stakeholder value, and that capturing stakeholder value by stakeholder role is helpful for decision-making. However, there remains considerable future work to develop adequate theory on stakeholder value and stakeholder viewpoints, and improve understanding of the prioritization process.

2 The Need for Prioritization

2.1 Lack of Guidance

Prioritization can be considered something of a “gap” in current software engineering. Certainly within the most commonly used system development methods, it has had far too low a profile in the past. Also industry standards such as the Integrated Capability Maturity Model (CMMI) [2] and SWEBOK (Software Engineering Book of Knowledge) [3] fail to offer specific guidance on the prioritization process. This lack of attention matters because of the “bigger picture”: the main purpose of prioritization is to help ensure projects are implementing the “right thing” at the “right time”, while making good use of the always limited human, monetary and time resources. Opportunities to assist project planning, and so improve project delivery, are being lost if prioritization is not intelligently executed.

In addition, the demand to move towards value-based software engineering (VBSE) [4] raises the need for greater attention to be paid to the delivery of stakeholder value. Indeed, Sullivan [5] reports on a lack of “formal, testable and tested theories, methods, and tools to support economic-based analysis and decision-making (and value-based analysis more broadly)”.

2.2 Changing Needs for Prioritization

Moreover, recent developments in software development mean that prioritization can be seen today as having a more central, on-going role to play throughout systems development. In Waterfall methods, prioritization only has to be carried out once, early on in the systems development process, and involves deciding what requirements are to be in the system and what are not. However, prioritization processes now have to support iterative and incremental development [6]. Such development requires on-going communication to capture data from the external environment, accept changing requirements, and receive feedback from each incremental delivery, in order to then establish what the stakeholders agree is of high value and should be in the next increment. This means the prioritization process has to cater for reuse of data while also accommodating changing data. Moreover, dynamic prioritization has to occur with each increment to determine what to implement next. Also that on-going identification of high stakeholder value is essential.

An additional demand comes from the recognition of the need for improved stakeholder understanding, especially the handling of multiple stakeholder viewpoints [7]. There is a need to not only capture and present the different viewpoints, but also to enable stakeholder negotiation and tradeoffs, and to help achieve stakeholder consensus and buy-in [8]. The prioritization process has a major part to play in providing better support to the system/product owners, who decide what shall be implemented.

3 Existing Research on Prioritization

Research in the area of prioritization has increased in recent years. In 1997, Karlsson and Ryan [9] wrote an influential paper describing their Cost-Value Approach based on the Analytic Hierarchy Process (AHP) [10], which acted as a springboard for much subsequent research. In this section, we briefly review the existing literature on prioritization: listing the existing prioritization methods, the identified prioritization factors and some of the identified issues with structuring prioritization data. Concurrently, we analyse how this existing research relates to the problems we perceive in prioritizing stakeholder value.

3.1 Positioning of Prioritization

Aspects of prioritization are discussed in the IT literature under several subject areas including requirements prioritization [11], [12], [13], release planning [14], architecture selection [15], COTS (Commercial Off-The-Self) selection [16], financial management [17], [18], and decision-making and negotiation methods [7]. There appears to be compartmentalization in the literature, which we argue needs questioning. While specialist areas for prioritization exist, it is essential that an overall view be considered because any given system encompasses many of these subject areas: there has to be interaction and integration at the system level. Accordingly, the stance taken by this research is that a holistic view should be taken: any overall prioritization process must include consideration of a wide range of prioritization data, which includes the fundamental software engineering concepts that we have termed here as “objective”, “requirement”, “design” and “increment”. All these four concepts impact on the concept of stakeholder value. For example, carrying out a prioritization process using just the requirements without consideration of, say, the potential designs and the operational impacts, both of which affect the costs, needs to be questioned. See Figure 1, which shows an increment delivery cycle with iteration around these concepts as software development progresses.

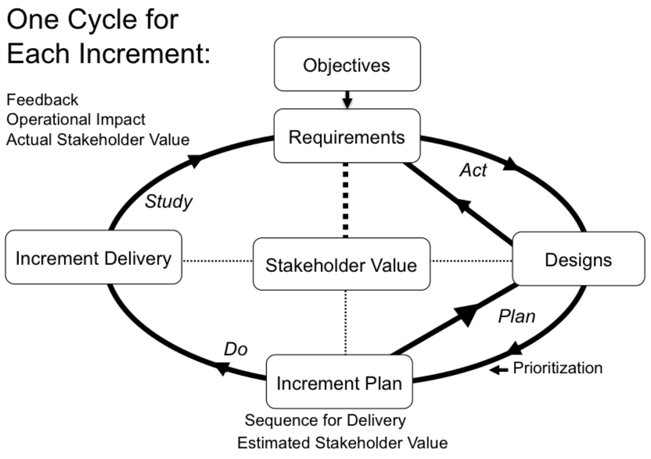


Fig. 1. Increment delivery cycle based on Deming's Plan-Do-Study-Act (PDSA) Cycle.

Despite the previous argument, responsibility for the prioritization process and data model probably should reside within requirements engineering because it interfaces with the majority, if not all, of the stakeholders, and because the system requirements form the primary (but not sole) data for prioritization. However, care needs to be taken that there is adequate consideration of the wider aspects of the prioritization process that fall within other viewpoints, such as strategy management and operations management.

3.2 Existing Prioritization Methods

To date, we have identified over 60 different prioritization methods in the literature. For brevity, full discussion of these is not given. A selection of those found categorized by subject area is as follows:
Requirements Prioritization: MoSCoW [19], the Hundred-Dollar Test [20] and Requirements Prioritization Tool (RPT) [12].
Requirements (and Effort) Prioritization: Cost-Value Approach [9] and Wiegers' Method [21].
Requirements (and Design) Prioritization: Analytic Hierarchy Process (AHP) [10], Quality Function Deployment (QFD) [22], [23] and Impact Estimation (IE) [1].
Architecture (Design) Prioritization: Cost Benefit Analysis Method (CBAM) [15] and Reasoning Frameworks [24].
COTS (Design) Prioritization: Procurement-Orientated Requirements Engineering (PORE) [16] and Mismatch Handling for COTS Selection (MiHOS) [25].

Release Planning: Planning Game [26], EVOLVE/EVOLVE* [14], [27] and Requirements Triage [8].
Financial Prioritization: Business Case Analysis/ROI [28], Incremental Funding Method (IFM) [29] and Real Options Analysis [17].
Negotiation Prioritization: Quantitative WinWin [30] and Distributed Collaborative Prioritization Tool (DCPT) [7].
Others: Conjoint Analysis [31].

The prioritization methods given most coverage in the literature include AHP, QFD the Cost–Value Approach, and more recently, the Planning Game.

However, it is not clear to what extent all these methods are used by software development in industry, or indeed how successful they have been [32]. Indeed, there appear to be some problems with the take-up and continued use of the well-known prioritization methods, such as QFD [33] and AHP [21].

3.3 Prioritization Factors

There are many prioritization factors (also sometimes called "criteria" [34], [35] or "aspects" [20], [32]) that can be considered in the prioritization process. We have identified a list of over 50 prioritization factors from the literature; the main sources include [12], [13], [14], [21], [35], [36], [37]. See Table 1, in which we chose to sub-divide the factors into three categories by stakeholder viewpoint and note the similarity to the choices of Lehtola [32] and Barney, *et al.* [35].

For brevity here, we have limited discussion of our work to just three stakeholder viewpoints that are representative of the mandatory viewpoints in any systems development prioritization process: strategy management, systems development and operations management. (Clearly there are many more stakeholder roles than these in a system.) We added a further sub-division under the four software engineering concepts used earlier in Figure 1. We decided that strategy management has responsibility for the objectives, systems development is primarily responsible for the requirements and designs, and operations management has responsibility for accepting the planned and delivered increments. In other words, the data associated with the selected system concepts would be of prime interest to the stakeholder viewpoint when establishing priorities. Furthermore, we introduced grouping of the prioritization factors by concept area, for example, strategy, cost and risk. Several of these groups are also identified by Berander [20] as "aspects". Note that, due to space limitations, any explanations of individual prioritization factors and relevant references have been omitted. Note also that these prioritization factors are not complete; this table only reflects the main prioritization factors found in the literature. The following observations can be made:

A general set of prioritization factors that could be proposed as "a starter" for a prioritization process emerges from the table. The prioritization factors span all the four software engineering concepts. This argues for a prioritization process that offers support for all these concepts. If more narrowly focused, specialized, prioritization methods are to exist then they need to integrate into an overarching prioritization process/method. The table provides support for the existence of different stakeholder viewpoints in the mappings between the stakeholder viewpoints and the prioritization

Table 1. Prioritization factors by stakeholder viewpoint and software engineering concept.

STAKEHOLDERS	**Strategic management**	**Systems development**		**Operations management/ customers**
CONCEPTS	**Organizational objectives (objective)**	**Systems requirements (requirement)**	**Design solutions (design)**	**Delivery plans (increment/delivery)**
PRIORITIZATION FACTORS				
OPINION	Vision/intuition/gut feeling/preference/ bias	Preferences/ bias/ importance	Intuition/ preferences/ bias	Preferences/ bias
STRATEGY	Strategic alignment/ business objectives/ product strategy		Long-term Strategy for systems architecture	
	Competition	Quality		
	Customer demand	Originator of requirement		End user value
	New business potential			
TIME	Urgency/time to market/lead time		Time schedule/ time constraints	
	Long term versus short term		Long term versus short term	
LEGAL	Legal mandate/ regulations	Legal mandate/ regulations		
	Contracts in place			
FINANCIAL BENEFIT	Market value/price			
	Financial benefits			
	Financial penalties			
	Benefit/cost ratio			
	Cost of not implementing			
COST	Development costs/ implementation costs/ support costs		Development costs/ support costs	Implementation costs/ support costs
	Operational costs			Operational costs
FIT	Fit with operational context: . business processes . skills/training . delivery timing		Staff competence Balanced workload	Fit with operational context: . business processes . skills/training . delivery timing
			Resource availability/ effort constraints	Resource availability/ effort constraints
	Fit with other products		Change impact/ base code dependencies	Change impact
			Logical implementation order	
			Reuse potential	
EXTERNAL DEPENDENCY	Intermediary channels	External dependencies	External dependencies	
RISK	Business risk Sales barriers	Volatility of requirements	Technical risk in: . current system . proposed system . implementation process	
			Difficulty of implementation/ complexity	Difficulty of implementation/ complexity

factors: different stakeholder viewpoints are interested in and knowledgeable about different prioritization factors. This means any prioritization process or prioritization method must cater for different stakeholder viewpoints.

A tentative observation can be made that the prioritization factor groupings (for example, strategy, legal, cost and risk), map across to the dimensions for stakeholder value.

3.4 Known Issues in Structuring Prioritization Data

A list of issues encountered when structuring the prioritization data to support the prioritization process was identified by extrapolating from discussions in the literature, for example from [8], [11], [12], [21], [36], [37], [38], [39]. The issues considered relevant to expressing prioritization data include:

Explicit stakeholder value: This is the often the expression of stakeholder priority to reflect the stakeholder value as well as the capture of explicit value.

Multiple stakeholder viewpoints: There is a need to handle different areas of interest/expertise and capture the different viewpoints together with their associated stakeholder value.

Requirements abstraction: This is the ability to handle requirements captured at different levels of refinement.

Interdependencies: The ability to express interdependencies among the requirements and also the designs. This becomes increasingly important with iterative and incremental development.

Dynamic prioritization: The priority data must be captured in order that it can be reused in subsequent prioritizations (future increments) without needing further inputs from stakeholders (unless something significant has changed in the system and/or its environment that they need to provide additional data on).

Scaling-up: This is the ability to scale up to cope with large numbers of items. Some existing prioritization methods become impractical when the number of requirements begins to grow to sizes typical of modern systems. In fact, for most large-scale projects, prioritization can tend to be carried out at a fairly high level of abstraction.

4 Analysis of Existing Prioritization Data

4.1 Expressing Prioritization Data

How prioritization data is expressed is a key factor in a prioritization process. We argue in this section that the prioritization data that the prioritization methods currently utilize (in content and structure) is insufficient to support the type of enhanced prioritization process that ideally needs to be adopted. Specifically, the lack of use of quantified data captured on absolute scale types is hindering progress.

The type of scale being used to capture the data is specifically important as it identifies the extent to which arithmetic calculations can validly be carried out. Only

Table 2. Mapping of prioritization technique(s) and scale type(s) to prioritization methods.

Prioritization Method	Prioritization Technique(s)	Scale Type(s)
QFD	Weighting and Grouping	Ratio Ordinal
AHP	Weighting (Pair-wise comparison)	Ratio
IE	Metrics	Absolute
Cost-Value Approach	Weighting (Pair-wise comparison)	Ratio
MoSCoW	Grouping	Ordinal
Planning Game	Grouping	Ordinal
Requirements Triage	Grouping and Weighting	Ordinal Ratio

the ordinal and ratio scale types are commonly used in existing prioritization methods. The absolute scale type is only occasionally used at present, but we propose it should be much more widely used and in fact, that it should replace much of the use of the ordinal and ratio scale types.

4.2 Prioritization Techniques

Several different ways (sometimes termed "prioritization techniques" [20]) of expressing prioritization data can be identified [13], [37]. We have reduced the number of different categories to four main ones as follows:

Grouping: The individual items are each categorized into one of a set of priority groups, for example the MoSCoW prioritization method demands each requirement is categorized as either "must have", "should have", "could have" or "would like, but wouldn't have this time" [19]. The results are on an ordinal scale.

Ranking: Requirements are ranked in order of preference. Ranking is carried out by bubble sort or by binary search tree [11]. This is an ordinal scale of measure as there is no information about the differentials amongst the ranked items.

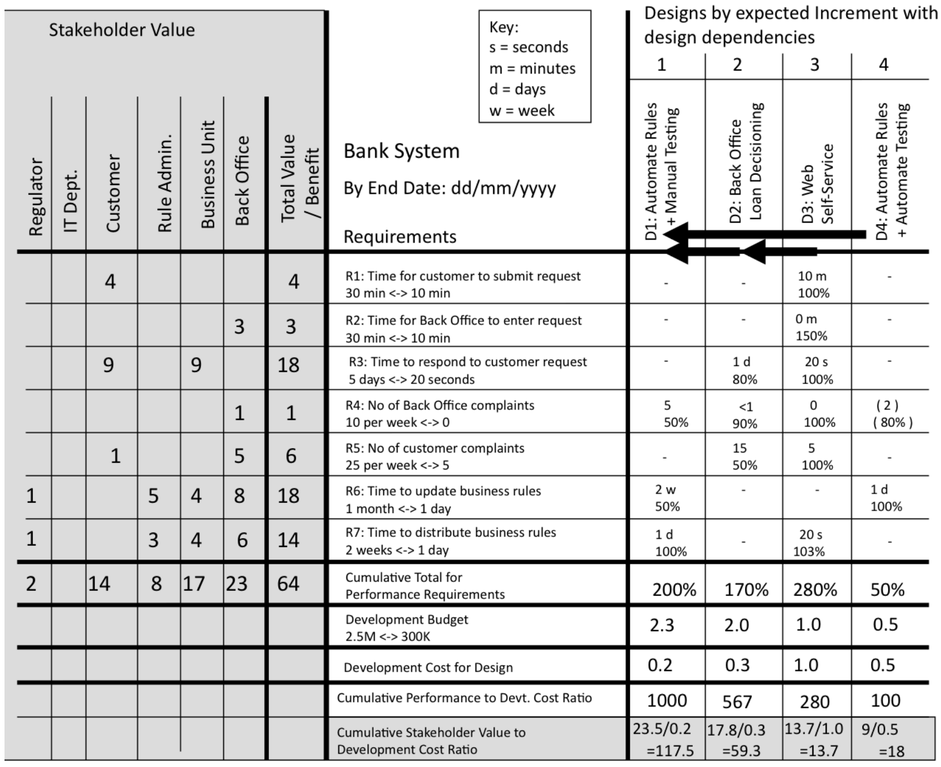
Weighting: Stakeholders assign their preferences and relative weightings are calculated. The results are on a ratio scale. One means of obtaining the weightings is by using voting [13]: stakeholders are requested to distribute some fixed number of votes (say 100 or 1000 dollars) amongst the different items being prioritized. Another means is by using pair-wise comparison: priorities are calculated by creating a hierarchy with branches of up to seven comparable items and then the items within each branch are pair-wise compared using a scale of 1 to 9 where 1 equates to "equally important" and 9 equates to "extremely more important" [12]. The scales are then converted to normalized weightings, which are then carried up the hierarchy. In AHP, pair-wise comparison is used to first weight the requirements, and then the designs.

Metrics: Absolute scales of measure are used to express certain attributes and these metrics form the basis for selection, for example by enabling calculation of ROI figures [37]. ROI calculation needs data on the amount of benefit (stakeholder value) that would be achieved by implementing a given design and the implementation cost associated with it. Only absolute scale data enables such ROI estimates to be calculated, as explicit stakeholder value data such as "a cost saving over the next year of 220,000 monetary units" would be captured. This contrasts to the ordinal scale data of say, the MoSCoW method, which simply captures requirements identified as of high stakeholder value into a "must have" priority group. In this paper, we are using Planguage [40] to express metrics, which captures the performance and resource requirements, as required levels on scales of measure.

See Table 2, which gives some examples of how the scale types and prioritization techniques map to a selection of prioritization methods.
See also Table 3, which shows how the prioritization techniques cope with a selection of prioritization data issues. Some example data has been inserted in the top row. From this row, it can be seen that use of metrics with absolute scale types results in real data that is much easier to understand and say, discuss with another stakeholder. It is less ambiguous than trying to work out what "Medium" should be interpreted to

Table 3. How prioritization techniques cope with a selection of data structuring issues.

Prioritization Technique >	**Grouping**	**Ranking**	**Weighting**	**Metrics**
Example of prioritization data	"High", "Medium" or "Low"	1, 2, 3, ... N	30/100	Time to carry out task to be reduced from 1 day to 5 minutes
Data Structuring Issue				
Stakeholder value	Implicit; value is say, "Medium"	Implicit; value is say, ranked as "2"	Implicit; value is 30% of whatever 100% equates to	Depends on metric. For this metric, an estimate of value is able to be derived if say, monetary rate of pay is known.
Multiple stakeholder viewpoints (Note assuming 2 stakeholders)	N Would be represented as say, "High" and "Medium"	N Would be represented as say, "2" and "20"	N Would be represented as say, 30/100 and 2/100	Y Time to carry out task to be reduced from 1 day down to say, 5 minutes and to 2 hours
Requirements Abstraction	N	N	Y Create hierarchy	Y Create hierarchy
Interdependencies	(Y) Would have to work by selecting an item and then seeing if there were any prior dependencies that would override	(Y) Ditto	(Y) Ditto	(Y) Ditto
Dynamic prioritization	Y Add any new data to an existing data grouping. No extra effort (unless something has changed)	Y Would need to re-examine existing ranks	(Y) Considerable effort needed by stakeholders	Y Add to existing data and reprocess
Scaling up	N Too many in a group	N Difficult to keep track of numerous rankings	N Considerable effort to carry out all the additional pair-wise comparisons	(Y) Would use high-level hierarchical data to reduce numbers



Key:
s = seconds
m = minutes
d = days
w = week

Stakeholder Value: Regulator	IT Dept.	Customer	Rule Admin.	Business Unit	Back Office	Total Value / Benefit	Bank System By End Date: dd/mm/yyyy Requirements	Designs by expected Increment with design dependencies 1 D1: Automate Rules + Manual Testing	2 D2: Back Office Loan Decisioning	3 D3: Web Self-Service	4 D4: Automate Rules + Automate Testing
		4				4	R1: Time for customer to submit request 30 min <-> 10 min	-	-	10 m 100%	-
					3	3	R2: Time for Back Office to enter request 30 min <-> 10 min	-	-	0 m 150%	-
		9		9		18	R3: Time to respond to customer request 5 days <-> 20 seconds	-	1 d 80%	20 s 100%	-
					1	1	R4: No of Back Office complaints 10 per week <-> 0	5 50%	<1 90%	0 100%	(2) (80%)
		1			5	6	R5: No of customer complaints 25 per week <-> 5	-	15 50%	5 100%	-
1			5	4	8	18	R6: Time to update business rules 1 month <-> 1 day	2 w 50%	-	-	1 d 100%
1			3	4	6	14	R7: Time to distribute business rules 2 weeks <-> 1 day	1 d 100%	-	20 s 103%	-
2		14	8	17	23	64	Cumulative Total for Performance Requirements	200%	170%	280%	50%
							Development Budget 2.5M <-> 300K	2.3	2.0	1.0	0.5
							Development Cost for Design	0.2	0.3	1.0	0.5
							Cumulative Performance to Devt. Cost Ratio	1000	567	280	100
							Cumulative Stakeholder Value to Development Cost Ratio	23.5/0.2 =117.5	17.8/0.3 =59.3	13.7/1.0 =13.7	9/0.5 =18

Fig. 2. VIE table for bank case study. The shaded area represents the extensions to IE.

mean. An observation can be made that all the techniques, apart from metrics, are generating additional data that captures some indirect notion of stakeholder value (such as "must have"), but not any explicit value (such as 220,000 monetary units).

5 Some Examples from a Case Study

5.1 Choice of Prioritization Method

By comparing how prioritization methods handled the prioritization factors and the data structure issues [41], and by considering the usage of software engineering concepts and scale types, we determined that IE offered an initial sound basis for this research: it spans the concepts of requirement, design and increment, and uses absolute scale data [1]. However, the IE method lacks consideration of explicit stakeholder value and stakeholder viewpoint, so we extended it to cater for stakeholder value by stakeholder role. We chose to link stakeholder value to requirement. See Figure 2 for an example of an extended IE table, which we term a value impact estimation (VIE) table. The non-shaded area is a basic IE table and the shaded area represents the extensions to IE.

5.2 Case Study Description

The case study examples are from a customer business rules "decisioning" system for a bank. The bank's objectives are customer satisfaction and, more efficient and effective internal processes. The main problems perceived by the bank are the time, effort and accuracy of updating and using the business rules, and the elapse time taken and the accuracy of dealing with customer requests. Of course, having up-to-date business rules in place impacts the accuracy of the handling of the customer requests. As the intention is to demonstrate that absolute scale data helps prioritization reasoning, a detailed discussion of all the requirements is not given here. We also limit our comments here about the use of performance requirements (also known as non-functional requirements) apart from recognizing that this is an additional reason why IE merits attention (given very few prioritization methods handle performance requirements [37]).

For brevity, a very restricted, cut down sample of the system specification is presented below. Note the data highlighted in bold in this specification is captured in the VIE table in Figure 2.

Stakeholders: Regulator, IT Department, Customer, Rules Administration, Business Units, Back Office.

Requirements:
Function: Submit request.
Performance requirement: **Reduce time for customer to submit request.**
Scale: Average time taken for defined [request type: Default = Loan].
Past: **30 minutes.**
Goal: **10 minutes.**

Function: Enter customer request details.
Performance requirement: **Reduce time for Back Office to enter request.**
Past: **30 minutes.**
Goal: **10 minutes.**

Function: **Process a customer request.**
Performance requirement: **Reduce time to process customer request.**
Past: **5 days.**
Goal: **20 seconds.**
Performance requirement: Reduce number of complaints.
Scale: Average number of complaints in defined [Time] from defined [Stakeholder].
Past **[Back Office]: 10 per week.**
Goal: **0 per week.**
Past **[Customer]: 25 per week.**

```
Goal: **5 per week.**

Function: **Update the business rules.**
Performance requirement: Reduce time to update rules.
Scale: Average time taken for defined [request type].
Past: **1 month.**
Goal: **1 day.**

Function: **Distribute business rules.**
Performance requirement: Reduce time taken. Scale: Average
time taken.
Past: **2 weeks.**
Goal: **1 day.**

**Designs:**
APTM: **Automate the rules & test manually.**
Rationale: Speed up the distribution to Back Office staff.

BD: **Back Office loan decisioning system.**
Rationale: Automating applying the rules will save time.
Dependency: **APTM.**

WSS: **Web self-service.**
Rationale: Customers can get a rapid response.
Dependency: **BD, APTM.**

APAT: **Automate the rules & test automatically.**
Rationale: Speed up the distribution to Back Office staff.
Dependency: **APTM.**
```

5.3 Description of a Basic IE Table

To create a basic IE table, the performance requirements are placed down the left-hand column. Each performance requirement shows its current baseline (Past) level and the required target (Goal) level. Beneath the performance requirements, the resource requirements are listed. Next, the designs are placed on the top row, and the estimated impact of each of the designs on each of the performance and resource impacts can be filled in as a level on the scale of measure. As discussed earlier, this involves estimating the level on the scale of measure that will result for a requirement if the design is implemented. If the baseline level is taken as 0% and the target level is taken as 100%, then the estimated percentage impact of the design on achieving the requirement can be calculated and the percentage change can be determined.

By looking down a column for a given design, you can see which requirements it is contributing towards meeting. By summing the estimated percentage changes in the performance requirements down a column for a given design, and then dividing by the estimated development cost of the design, an estimated cumulative performance to cost ratio for each design can be calculated. The performance to cost ratios for the potential designs within an IE table can be compared to determine which design offers the most impact given its cost. In this case study, the designs are complementary so the aim is to sequence the implementation order to deliver the highest value as early as possible. Looking at Figure 2, it can be seen that the designs are in the right order regarding the figures for the estimated cumulative performance to (development) cost ratios. Note the totals for the performance impacts are calculated based on the estimated *additional* percentage impact over the estimated percentage impact achieved after the last design was implemented. So design BD contributes an additional 40% (90% - 50%) over the 50% estimated for APTM, so its total estimated percentage impact is 80% + 40% + 50% = 170%. A further refinement was to cap any percentage impact at 100% for the calculations. As implementation proceeds and increments are completed, the actual results can be measured and captured in the IE table alongside the estimates. This allows deviations from the planned levels to be identified and future plans adjusted as appropriate.

5.4 Extending IE to Cater for Multiple Stakeholders and Stakeholder Value

To address the problems with multiple stakeholders and stakeholder value, as already outlined earlier, we extended the basic IE table to capture on its left-hand side the different stakeholders of interest and their associated stakeholder value, see the shaded areas of Figure 2. The figures for stakeholder value given in the stakeholder columns represent the stakeholder value of achieving 100% of each requirement. In this VIE table stakeholder value was estimated on the financial value of the estimated time saving and the estimated additional sales. Note the actual financial values are not given here due to the commercial sensitivity of this information. Instead the financial figures for stakeholder value were all divided by the lowest figure and then rounded to the nearest integer.

In turn, the extension of adding explicit stakeholder value enables cumulative stakeholder value to development cost ratios to be calculated for the different designs as shown in the shaded bottom row of Figure 2. The calculation is worked out on the basis that an estimated percentage impact of a design on a requirement will result in the same percentage of the stakeholder value of the requirement being achieved. In other words, an assumption that the utility curve [42] is linear. From the results for the value to cost ratios, it appears that maybe the implementation order of the designs, WSS and APAT should be reversed.

6 Conclusions

Despite much research in the last decade on prioritization in software engineering projects, progress is being hampered by inadequate representation of stakeholder value. The issue is becoming more urgent because dynamic prioritization of stakeholder value is increasingly needed as iterative and incremental development methods become more widely used. We have argued in this paper that the use of absolute scale data is essential to address the problems with the current prioritization processes: specifically, to provide unambiguous prioritization data that stakeholders can understand and relate to, and to support arithmetic calculations.

This paper has briefly reviewed existing research on prioritization. Our findings include:

- By categorizing and analysing the existing prioritization methods, that many (but not all of) the existing prioritization methods are restricted in their scope (for example, some methods are just considering the requirements).
- By investigation of the prioritization factors discussed in the literature, we have shown that the scope of the prioritization process spans system-wide data from organizational objectives to increment delivery. An additional finding from this data is that different stakeholders have different viewpoints on the prioritization factors, and that therefore, multiple stakeholder viewpoints need to be supported.
- By identifying the known issues with structuring prioritization data and analyzing how the prioritization techniques and scale types used in prioritization methods tackle these issues, we determine that the techniques of grouping, ranking and weighting are weaker than metrics in addressing the issues. Specifically, expression of stakeholder value is implicit in the prioritization data, and that arithmetic calculations are often impossible or problematic, apart from when metrics are used.

Further, we demonstrate the validity of the use of absolute scale data in prioritization by using VIE, an extended version of the IE prioritization method with some examples from a case study. We specifically extended IE to cater for stakeholder value for multiple stakeholders. We show the ability to carry out calculations to investigate requirement and design priorities.

Work is underway to investigate further extending the IE method to represent additional aspects of stakeholder value. Future work plans to make the detailed decision-making of a rational prioritization process explicit.

References

1. Gilb, T.: Competitive Engineering: A Handbook For Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage. L. Brodie (Ed.). Butterworth-Heinemann. ISBN 0750665076 (2005)
2. Boehm, B., Port, D., Basili, V.R.: Realizing the Benefits of the CMMI with the CeBASE Method. Systems Engineering. J. Wiley and Sons, Inc. (2002)
3. Bourque, P., Dupuis, R. (eds.): SWEBOK: Guide to the Software Engineering Body of Knowledge 2004 Version. IEEE Computer Society (2004)
4. Boehm, B.: Value-Based Software Engineering. Software Engineering Notes, Vol. 28, No. 2, ACM (2003)
5. Sullivan, K.: Introduction to the First Workshop on the Economics of Software and Computation. In: Companion to the Procs. of the 29th International Conf. on Software Engineering. IEEE (2007)
6. Larman, C., Basili, V.R.: Iterative and Incremental Development: a Brief History. IEEE Computer, 36, 6 (2003)
7. Park, J., Port, D., Boehm, B.: Supporting Distributed Collaborative Prioritization for Win-Win Requirements Capture and Negotiation, Procs. of the International Third World Multi-conference on Systemics, Cybernetics and Informatics (SCI'99), International Institute of Informatics and Systemics (1999)

8. Davis, A.: The Art of Requirements Triage. IEEE Computer, March 2003, 36, 3, pp. 42-49 (2003)
9. Karlsson, J., Ryan, K.: A Cost-Value Approach for Prioritizing Requirements. IEEE Software (1997)
10. Saaty, T.L.: How to Make a Decision: The Analytic Hierarchy Process. European Journal of Operational Research, 48, 1990, pp. 9-26 (1990)
11. Karlsson, J., Wohlin, C., Regnell, B.: An Evaluation of Methods for Prioritizing Software Requirements. Information and Software Technology, Elsevier Science B.V. (1998)
12. Moisiadis, F.: The Fundamentals of Prioritising Requirements. Procs. of the Systems Engineering, Test and Evaluation Conference (2002)
13. Berander, P., Andrews, A.: Requirements Prioritization (Chapter 4). In: Aurum, A. and Wohlin, C. (Eds.) Engineering and Managing Software Requirements. Springer-Verlag. ISBN 3540250433 (2005)
14. Greer, D., Ruhe, G.: Software release planning: an evolutionary and iterative approach. Information and Software Technology, 46, 4, pp. 243-253 (2004)
15. Kazman, R., Asundi, J., Klein, M.: Quantifying the Costs and Benefits of Architectural Decisions. Procs. of the 23rd International Conference on Software Engineering (ICSE 2001), IEEE (2001)
16. Mohamed, A., Ruhe, G., Eberlein, A.: COTS Selection: Past, Present, and Future. Procs. of the IEEE Intl. Conf. and Workshops on the Engineering of Computer-Based Systems (ECBS'07), IEEE (2007)
17. Favaro, J.: Managing Requirements for Business Value. IEEE Software, March/April 2002 (2002)
18. Sivzattian, S.V.: Requirements as Economic Artifacts: A Portfolio-Based Approach, Ph.D. Thesis. Department of Computing, Imperial College of Science, Technology and Medicine, London (2003)
19. Stapleton, J. (Editor): DSDM: Business Focused Development (2nd Edition). Addison Wesley (2003)
20. Berander, P.: Evolving Prioritization for Software Product Management. Blekinge Institute of Technology. Doctoral Dissertation Series (2007)
21. Lehtola, L., Kauppinen, M.: Suitability of Requirements Prioritization Methods for Market-driven Software Product Development. Software Process Improvement and Practice. Wiley (2006)
22. Cohen, L.: Quality Function Deployment: How to Make QFD Work for You. Addison Wesley (1995)
23. Akao, Y.: QFD: Past, Present, and Future. Procs. of the International Symposium on QFD '97, Linkoping (1997)
24. Bass, L., Ivers, J., Klein, M., Merson, P.: Reasoning Frameworks (CMU/SEI-2005-TR-007). Software Engineering Institute, CMU (2005)
25. Mohamed, A., Ruhe, G., Eberlein, A.: Decision Support for Handling Mismatches between COTS Products and System Requirements. Procs. of the COTS-Based Software Systems (ICCBSS'07) Conf. (2007)
26. Beck, K.: Extreme Programming Explained: Embrace Change. Addison Wesley (2000)
27. Saliu, O., Ruhe, G.: Supporting Software Release Planning Decisions for Evolving Systems. Procs. of the 2005 29th Annual IEEE/NASA Software Engineering Workshop (SEW'05), IEEE (2005)
28. Favaro, J.: Value-Based Management and Agile Methods. Marchesi, M. and Succi, G. (eds.) Procs. of the 4th International Conference on XP and Agile Methods, May 2003. Springer-Verlag (2003)
29. Denne, M., Cleland-Huang, J.: Software by Numbers: Low-Risk, High-Return Development. Prentice-Hall. ISBN 0131407287. 190 pages (2004)

30. Ruhe, G., Eberlein, A., Pfahl, D.: Trade-off Analysis for Requirements Selection. International Journal of Software Engineering and Knowledge Engineering, 13, 4, pp.345-366 (2003)
31. Green, P.E., Wind, Y.: New Way to Measure Consumers' Judgments. Harvard Business Review (1975)
32. Lehtola, L.: Providing value by prioritizing requirements throughout software product development: State of practice and suitability of prioritization methods. Licentiate thesis, Helsinki University of Technology (2006)
33. Martins, A., Aspinwall, E.: Quality Function Deployment: an empirical study in the UK. Total Quality Management, August 2001 (2001)
34. Wohlin, C., Aurum, A.: Criteria for Selecting Software Requirements to Create Product Value: An Industrial Empirical Study. In: Biffl, S., Aurum, A., Boehm, B., Erdogmus, H., Grunbacher, P. (eds.) Value-Based Software Engineering. Springer (2005)
35. Barney, S., Aurum, A., Wohlin, C.: A product management challenge: Creating software product value through requirements selection. Journal of Systems Architecture, Elsevier (2008)
36. Carlshamre, P., Sandahl, K., Lindvall, M., Regnell, B., Natt och Dag, J.: An Industrial Survey of Requirements Interdependencies in Software Product Release Planning. Procs. of the 5th IEEE International Symposium on RE (2001)
37. Firesmith, D.: Prioritizing Requirements. Journal of Object Technology (2004)
38. Gorschek, T., Wohlin, C.: Requirements Abstraction Model. Requirements Engineering. Springer Verlag, 11, pp. 79-101 (2006)
39. Mead, N.: Requirements Prioritization Introduction. Software Engineering Institute (SEI), Carnegie Mellon University (CMU) (2006)
40. Gilb, T.: Principles of Software Engineering Management. Addison Wesley. ISBN 0201192462 (1988)
41. Brodie, L., Woodman, M.: Towards a Rational Prioritization Process for Incremental and Iterative Systems Engineering. Procs. of the 1st International Workshop on Requirements Analysis, Pearson (2008)
42. Daniels, J., Werner, P.W., Bahill, A.T.: Quantitative Methods for Tradeoff Analyses. Systems Engineering, John Wiley & Sons, Inc., 4, 3 (2001)

Using Metrics to Express Quality

Lindsey Brodie & Mark Woodman

Middlesex University e-Centre
School of Engineering & Information Sciences
The Burroughs, Hendon,
London NW4 4BT
l.brodie@mdx.ac.uk & m.woodman@mdx.ac.uk

Abstract

Many IT projects focus mainly on functionality and struggle in their handling of the system's quality attributes, such as availability, adaptability and usability. This often means that system quality is treated as a 'bolt-on' rather than being designed in. The underlying problem is the inadequate support for specifying performance attributes within many systems development methods. However, one method that addresses this area is Planguage, which handles performance attributes by the use of quantified system information – i.e. metrics. This paper explores Planguage's use of metrics, and also investigates how it integrates the handling of basic system concepts such as function, performance (including quality), and design. The advantage of such an integrated and quantified approach is then illustrated by describing Planguage's impact estimation (IE) method. Within IE, quantified performance attributes are fundamental to the project design, planning and control processes. Further, building on this approach, we propose an extension to IE to express stakeholder viewpoints and stakeholder value more explicitly. Examples from a case study are given to support the discussion.

1.0 Introduction – The problem

Many IT projects struggle in their handling of the system's quality attributes, such as reliability, usability, and security [1]. Too often, precedence is given to designing to meet the functionality and the quality requirements are "retrofitted late in the development process" or "pursued in parallel with, but separately from, functional design" [2]. The underlying problem is the inadequate support for the specification of performance attributes within many systems development methods. As we shall explore in this paper, the issues are deep-rooted and involve reconsidering the relationships amongst functions and performance attributes. Here (and in the rest of this paper) we are deliberately using the Planguage term "performance attribute", which includes the quality attributes, as we shall discuss later.

1.1 Current specification of performance attributes

Quality requirements are commonly known as non-functional requirements (NFRs). As Woodward [3] has suggested, the term NFR is too negative and can be taken to imply such attributes have a secondary role to functionality. Certainly, the handling of quality requirements is often poor. Firesmith [4] considers "these types of requirements are often given far too little priority, are not specified at all, or are specified in a vague untestable manner." Sometimes there are simply indirect references to the required attributes scattered amongst the requirements text. For example, in a specification for bank system that we shall use as a case study throughout this paper, phrases such as "up-to-date view", "easy to use rules administration", "low overhead cost", "in a timely manner" and "high performance" are present without any quantification of how they should be interpreted. Alternatively, in many requirements specifications there is a false separation of the quality attributes from the system functionality: often there is a substantial list of system functions followed by a separate shorter section covering the quality attributes using a few high-level, often ambiguous, statements.

Rarely is any attempt made to express quantified performance attributes for specific functions. Instead, performance attributes are frequently treated as if they always applied uniformly across the entirety of a system. For example, a quality performance attribute of "24 x 7 availability" is rarely needed for all the functions of a system; functionality used mainly during office hours can often be unavailable for maintenance purposes at some time overnight without incurring any major problems. Aside from the issue of not accurately capturing the requirements, an apparently small distinction like this can have a major effect in reducing operational costs – itself potentially a quantifiable performance attribute. In fact, even further qualifications can be placed on the scope of a performance attribute, such as by location, stakeholder role, time or event, all having a potential impact on the costs of the system. Returning to the bank system specification, there was no breakdown relating specific performance requirements to specific functions or any further qualifiers discussed in the specification.

Possibly, as a result of the inadequate performance requirements specification, requirements prioritization – the ordering of requirements for implementation – is carried out in several prioritization methods mainly using the "functional aspects" [5]. Once again there is inadequate consideration of the performance attributes. Performance can alter independently of function: for example, a server can be given increased processing power by adding extra memory – its functionality remains the same. Moreover, the problems do not stop there: consider the use of functional specifications, function points, work breakdown structure (WBS) and Earned Value Management (EVM); they are all task-orientated, ignoring the contributions of the performance attributes.

1.2 Why performance attributes matter

In addition to the reasons already given, there are more fundamental reasons why failing to adequately specify the performance attributes matters. The requirements for change are often driven by the need to improve the performance attributes, rather than to modify the basic (high-level) functionality: for example, a bank system requires "greater security" or "improved efficiency". In the same way, when selecting or designing a system, it is the level of ambition of the performance attributes that will most likely dictate the final choice of design – that is, different designs can apply as the performance levels vary, while the system's functionality remains constant. In the bank system case study, the quality of interfacing with customers will primarily be judged by the number of transactions the system processes in a set time, the capacity of its database, and the responsiveness of its customer-facing website; these are all performance attributes. Arguably, the quality of a system is judged largely on the performance attributes, rather than the basic functionality.

If the performance attributes and their respective levels are not explicitly stated, there is a risk of some stakeholders misunderstanding the requirements, which could result in the serious consequence of inadequate designs being implemented. Additional costs are then likely to be incurred when trying to address any shortfalls: changing designs to cater for enhanced performance levels midway through system implementation is likely to be costlier than designing with those levels in mind from the start. At worst, there could be failure to identify upfront any high risks associated with performance attribute levels that are too challenging (moving towards state-of-the-art levels).

If we recognise the importance to a system of the performance attributes, the question becomes how should we specify them to better reflect their contribution? One potential solution, as we shall outline in this paper, is to use Planguage [6] [7]. Gilb has developed Planguage over the last three decades in industry and the handling of performance attributes using quantified system information – metrics – has been fundamental to this approach. Planguage is a specification language and set of methods for planning and controlling systems development. Note the emphasis on its being a *planning* method, Planguage is not intended as a method for detailed specification of system design.

We shall first explain how performance attributes (including quality attributes) are specified using Planguage, then explore how the method integrates the handling of such attributes with the other basic system concepts. To illustrate how Planguage actively uses its specifications for planning, including prioritization, and control purposes, a description of its impact estimation (IE) method will be given. Finally, we shall propose an extension to IE, building on its use of metrics and its integrated approach, to improve support for multiple stakeholder viewpoints and stakeholder value. Some examples from the bank system case study are used to explain the various points. Note that for space reasons, we are not describing all the features of Planguage and IE in this paper.

2.0 Planguage Specification

We start by exploring how performance attributes are specified. Planguage arranges its performance attributes into a hierarchy, where high-level attributes are decomposed into their component parts. See Figure 1: it shows a selection of generic performance attributes that can be used to describe any system (obviously there are more performance attributes than shown here). It also shows a small selection of performance *requirements* specific to the bank system case study, which breaks the hierarchy down into less abstract concepts. We shall continue to use this selection throughout the rest of the paper.

Planguage identifies three basic types of performance attribute: quality, resource saving, and workload capacity. This division recognises how stakeholders in practice state their requirements. "Quality" aims to capture all the attributes, not expressed in terms of resource utilisation, which stakeholders identify as being specifically valuable in a system. In addition to a generic set of system qualities (such as availability and usability), there need to be specific qualities that reflect the specific system objectives, for example, customer satisfaction, staff development and innovation. Notice that Planguage therefore defines "quality" in a slightly narrower way than the term is normally used. Planguage's use of "performance" as the umbrella term (incorporating quality) is a somewhat similar usage to USA MIL-STD 499B [6][8].

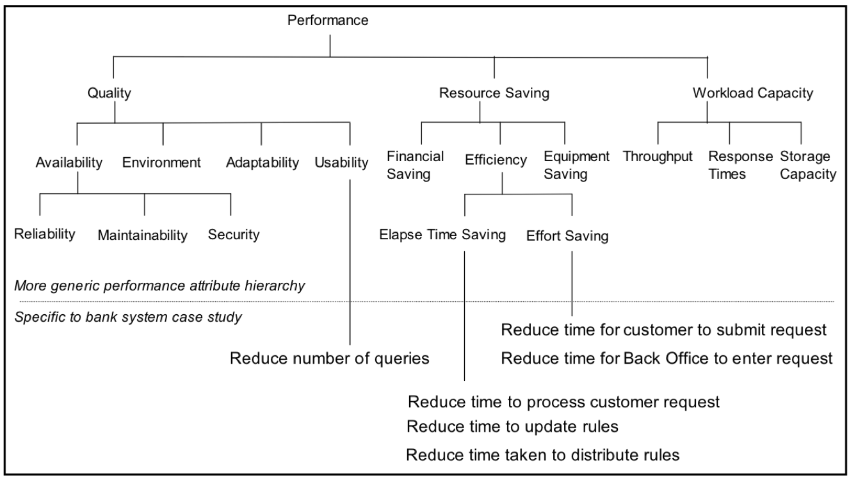

Performance
Quality
Resource Saving
Workload Capacity
Availability
Environment
Adaptability
Usability
Financial Saving
Efficiency
Equipment Saving
Throughput
Response Times
Storage Capacity
Reliability
Maintainability
Security
Elapse Time Saving
Effort Saving
More generic performance attribute hierarchy
Specific to bank system case study
Reduce time for customer to submit request
Reduce number of queries
Reduce time for Back Office to enter request
Reduce time to process customer request
Reduce time to update rules
Reduce time taken to distribute rules

Figure 1: Hierarchy of performance attributes

"Resource saving" (which could be extended to cater for gains as well as savings) recognises that often stakeholders express performance requirements by stating the levels of ambition for changes in resource utilisation (for example, by requiring specific levels of cost reduction). Often such resource savings can be interlinked with the quality attributes, so while specific attention has to be paid to ensuring the savings are achieved, care has also to be taken that the resulting benefits are not counted more than once. To give an example, "improving reliability" links to resource savings regarding "staff effort on recovery activities". In practice, for some systems the stakeholders will express their requirements as better reliability, for others as reducing staff effort (perhaps leaving it open as to what is improved to reduce effort), and indeed for some systems might even specify a mixture of both. Typically, without a performance-related resource saving categorisation, such requirements become muddled with the cost-related resource requirements, that is, the resources consumed, also known as the budgets. (See also later discussion in Section 3, especially Figure 2.)

Finally, the "workload capacity" category is used to specify the loading and storage volumes for a system. The workload capacities are key to understanding the system size and its needs for growth.

The overall aim of the performance attribute decomposition is to identify a set of key attributes that are representative of the objectives and that are capable of having scales of measure defined for them. Often when there are problems with identifying scales it is because the attributes are still too high-level and therefore too abstract. In order to specify performance attributes using Planguage, the following seven steps are needed.

1. Establish the vision: The senior management stakeholders should provide the vision. Stakeholder expectations might need to be managed accordingly and the feasibility of new ideas explored. For example, returning to the bank system case study, its vision was as follows:

> Vision: To reduce the elapse time to respond accurately to customer loan requests.
> Rationale: A faster response will result in more business.

2. Identify all the stakeholders: Often not all the requirements are identified because too narrow a view is taken of who is impacted by a system. Note we use the term 'stakeholder' to represent any grouping by role and/or location. For example in the case study, we identified several distinct stakeholders, but use of location to subgroup them further was not relevant.

> Stakeholders: Regulator, IT Department, Customer, Rules Administration, Business Units, Back Office.

3. Obtain and analyse the requirements: Ask the key stakeholders about their requirements. Then arrange the performance requirements into a hierarchy. Ensure these requirements are within the scope of the project to address and that the project can therefore take responsibility for achieving them. For the bank system, we chose to focus on the efficiency of the process and one attribute of usability, the number of queries received about the problems with using the process. We include the functions here to show the links to functionality. For example:

> Function: Process a customer request.
> Performance requirement: Efficiency.Elapse Time Saving.Reduce time to process customer request.
> Performance requirement: Usability.Reduce number of queries.

4. Determine relevant scales of measure for the performance requirements: Looking for any measurement already in use within the organisation is a good place to start. It is important that measurement is practical and cost-effective. It is also worth considering whether you are specifying a leading or a lagging indicator (How soon will any change in levels be detected? How directly does this measure what we want?) We chose monitoring the time taken and the number of queries raised. Planguage suggests you embed qualifiers within a scale of measure to make it more specific and also more reusable. For example:

```
Function: Process a customer request.
...
Performance requirement: Usability.Reduce number of queries.
Scale: Average number of defined [query type: Default = Loan] in defined [Time unit]
    from defined [Stakeholder].
```

5. Establish the levels on the scales of measure: There is a need to establish the actual current level and the required future level(s): Planguage uses 'Past' to denote current levels and 'Goal' for future levels. In additional these levels need putting in context by adding the specific [time, place, event] qualifiers. 'Time' qualifiers can be used to show either the date on which a level was measured or the date by which a level is required. 'Place' can be mapped to function, stakeholder, geographical location and other things as required. 'Event' qualifiers can make the levels conditional on certain events having occurred (such as if this contract is won then the level is set at 'x', otherwise it will be set at 'x-5'). The example below shows different levels being specified for two different stakeholders, Back Office and Customer.

```
Function: Process a customer request.
...
Performance requirement: Usability.Reduce number of queries.
Scale: Average number of defined [query type: Default = Loan] in defined [Time unit]
    from defined [Stakeholder].
Past [Back Office]: 10 per week.
Goal: <1 per week.
Past [Customer]: 25 per week.
Goal: 5 per week.
```

Notice the requirements are not simply about providing functionality, but rather the functionality with specific performance attributes at specific performance levels in a specific context. It is this fuller picture that equates to Planguage metrics.

6. Identify some potential design solutions: Here we are defining a design solution ("design") as anything that could make an impact towards fulfilling at least one of the requirements. Identifying designs might seem slightly strange at this stage, but there are several reasons for this. First, many stakeholders accidentally state designs in their requirements, and so there is a need to capture such stakeholders' input as design. Secondly, that the proposed designs might help identify additional requirements, and finally, because it is useful to discuss some proposed designs with stakeholders and establish what impacts they perceive they will have on their requirements (we shall return to this last point in Section 3). An example of specifying an initial design solution need not be too detailed:

```
Design: APTM: Automate the rules & test manually.
Rationale: Speed up the distribution to Back Office staff.
```

Putting all the above steps into practice for the bank system case study gives the following specification (The following is a much simplified version of Planguage, for example all dates have been removed and no source information is given):

Stakeholders: Regulator, IT Department, Customer, Rules Administration, Business Units, Back Office.

Requirements:
Function: Submit request.
Performance requirement: Efficiency.Effort Saving.Reduce time for customer to submit request.
Scale: Average time taken for defined [request type: Default = Loan].
Past: 30 minutes.
Goal: 10 minutes.

Function: Enter customer request details.
Performance requirement: Efficiency.Effort Saving.Reduce time for Back Office to enter request.
Past: 30 minutes.
Goal: 10 minutes.

Function: Process a customer request.
Performance requirement: Efficiency.Elapse Time Saving.Reduce time to process customer request.
Past: 5 days.
Goal: 20 seconds.
Performance requirement: Usability.Reduce number of queries.
Scale: Average number of defined [query type: Default = Loan] in defined [Time unit] from defined [Stakeholder].
Past [Back Office]: 10 per week.
Goal: <1 per week.
Past [Customer]: 25 per week.
Goal: 5 per week.

Function: Update the business rules.
Performance requirement: Efficiency.Elapse Time Saving.Reduce time to update rules.
Scale: Average time taken for defined [request type].
Past: 1 month.
Goal: 1 day.

Function: Distribute business rules.
Performance requirement: Efficiency.Elapse Time Saving.Reduce time taken to distribute rules.
Scale: Average time taken.
Past: 2 weeks.
Goal: 1 day.

Designs:
APTM: Automate the rules & test manually.
Rationale: Speed up the distribution to Back Office staff.

BD: Back Office loan decisioning system.
Rationale: Automating applying the rules will save time.
Dependency: APTM.

WSS: Web self-service.
Rationale: Customers can get a rapid response.
Dependency: BD, APTM.

APAT: Automate the rules & test automatically.
Rationale: Speed up the distribution to Back Office staff.
Dependency: APTM.

7. Obtain agreement from the relevant stakeholders: The specifications of the performance attributes and their performance levels must reflect accurately the requirements. Also ask the relevant stakeholders about the likely impacts of each of the proposed designs on the requirements. Planguage uses its impact estimation (IE) table to capture this information. As we shall discuss in the next section, this enables the likelihood of the requirements being achieved, the strengths of the proposed designs and the performance to cost ratios to be assessed.

3.0 Impact Estimation (IE)

3.1 Planguage System Model

Let's first very briefly look at the Planguage system model; see Figure 2. The main purpose of discussing this model is to show how Planguage integrates the performance attributes and other context attributes with functionality. From the previous discussion, many of the concepts in Figure 2 should be understood.

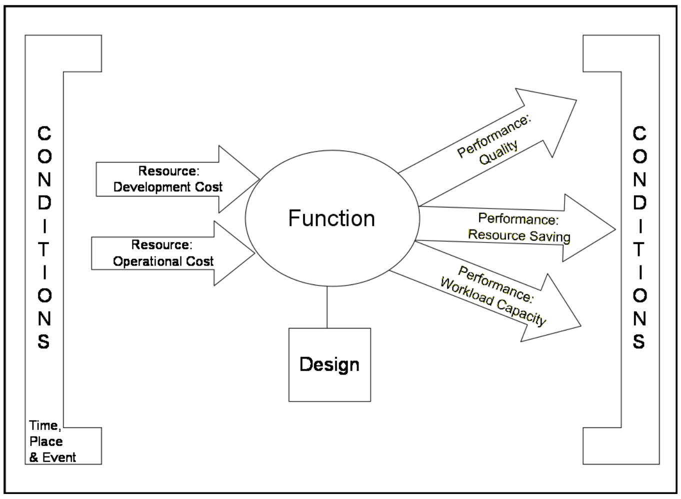

CONDITIONS
Resource: Development Cost
Resource: Operational Cost
Function
Design
Performance: Quality
Performance: Resource Saving
Performance: Workload Capacity
CONDITIONS
Time, Place & Event

Figure 2: Planguage system model

The highest-level function equates to the total system functionality. Any functionality implies a certain level of resource needed to support it (the costs) and enable delivery of certain levels of performance. While the existence of a function is binary in nature (either present or absent in a system), the performance and resource attributes are scalar (they can be at varying levels). A design (when implemented) provides the functionality with its specific resource and performance attributes within the specific context. As we described previously in Section 2, the [time, place, event] conditions set the system context for each function As the system context changes, the requirements are likely to alter and so too are the design solutions.

An Impact Estimation (IE) table is a matrix of requirements against designs. See Figure 3, which shows how the Planguage system concepts are captured within an IE table. An IE table captures a 'snapshot' of a system (or part of a system) at a specific point in time. So, typically, the time and event conditions are assumed as a given. Place conditions are very varied and tend to be dealt with by a combination of specific requirements and the choice of the proposed designs. Functionality is also considered in the choice of proposed designs in so far. The designs are chosen by their suitability to address some aspect of the required functionality and any unsuitable ones are filtered out. So the prime focus of an IE table is on the required levels of the performance attributes (that is the quality, resource saving and workload capacity attributes), and the contribution of the proposed designs towards meeting them.

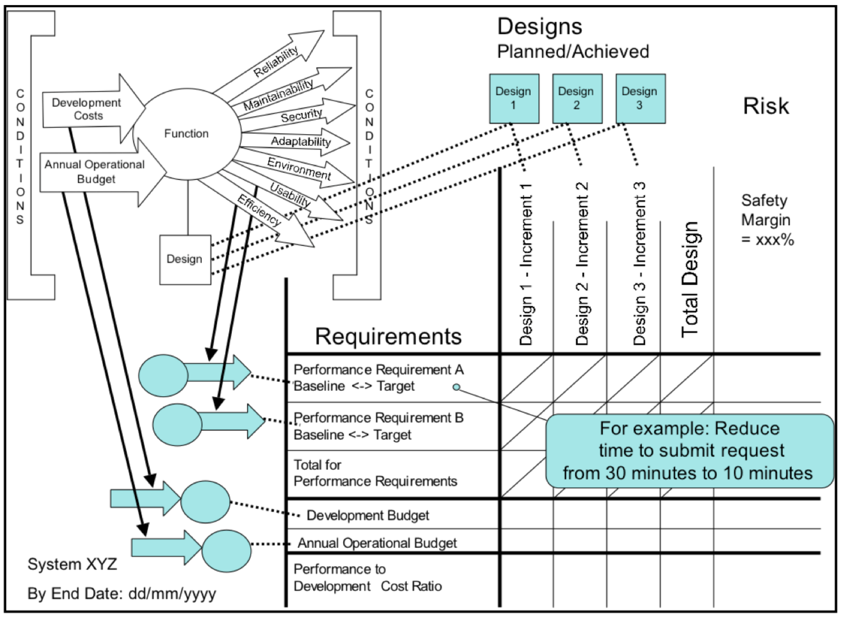


Figure 3: Mapping of the system concepts to an IE table

3.2 Creating an IE table

To create an IE table, the performance requirements are placed down the left-hand column. Each performance requirement shows its current baseline (Past) level and the required target (Goal) level. Beneath the performance requirements, the resource requirements are listed. Typically this is the development budget. Then across the top row, the designs are listed with any dependent designs being placed to the right-hand side of the design it is dependent on. If the designs are complementary, then the sequencing should be roughly in the proposed implementation order. If the designs are alternatives then order is not important. See the non-shaded area of Figure 4, which shows an IE table for the bank system. The dependencies amongst the designs are shown as bold arrows linking the relevant designs.

Next the estimated impact of each of the designs on each of the performance and resource impacts can be filled in as a level on the scale of measure. If the baseline level is taken as 0% and the target level is taken as 100%, then the estimated impact can be calculated as a percentage and the percentage change can be determined. For example, see Figure 4 and the impact of the design, "Automate Rules + Manual Testing" with the performance requirement, "Number of Back Office queries". The design is estimated to reduce the average number of queries from 10 per week down to 5, which is a 50% impact. The percentage change is 50% (i.e. from 0% to 50%).

The benefit of converting to percentages is that it enables some checking, and simple arithmetic to compare designs. By checking the percentage impacts across a row for a given requirement, you can assess the likelihood of achieving the target level (100%). By looking down a column for a given design, you can see which requirements it is contributing towards meeting. By totalling the percentage changes in the performance requirements down a column for a given design, and then dividing by the estimated development cost of the design, a cumulative performance to cost ratio for each design can be calculated. Here the designs are complimentary so the aim is to sequence the implementation order to deliver the highest value as early as possible. Looking at Figure 4, it can be seen that the designs are in the right order regarding the figures for the cumulative performance to (development) cost ratios.

Stakeholder Value

Key:
s = seconds
m = minutes
d = days
w = week

Bank System

By End Date: dd/mm/yyyy

Designs by expected Increment with design dependencies

Regulator	IT Dept.	Customer	Rule Admin.	Business Unit	Back Office	Total Value / Benefit	Requirements	1 Automate Rules + Manual Testing	2 Back Office Loan Decisioning	3 Web Self-Service	4 Automate Rules + Automate Testing
		4				4	Time for customer to submit request 30 min <-> 10 min	-	-	10 m 100%	-
					3	3	Time for Back Office to enter request 30 min <-> 10 min	-	-	0 m 150%	-
		9		9		18	Time to respond to customer request 5 days <-> 20 seconds	-	1 d 80%	20 s 100%	-
					1	1	No of Back Office complaints 10 per week <-> 0	5 50%	<1 90%	0 100%	(2) (80%)
		1			5	6	No of customer complaints 25 per week <-> 5	-	15 50%	5 100%	-
1			5	4	8	18	Time to update business rules 1 month <-> 1 day	2 w 50%	-	-	1 d 100%
1			3	4	6	14	Time to distribute business rules 2 weeks <-> 1 day	1 d 100%	-	20 s 103%	-
2		14	8	17	23	64	Cumulative Total for Performance Requirements	200%	170%	280%	50%
							Design Cost (M)	0.2	0.3	1.0	0.5
							Development Budget 2.5M <-> 300K	2.3	2.0	1.0	0.5
							Cumulative Perf. to Devt. Cost Ratio	1000	567	280	100
							Cumulative Stakeholder Value to Development Cost Ratio	23.5/0.2 =117.5	17.8/0.3 =59.3	13.7/1.0 =13.7	9/0.5 =18

Figure 4: An IE table for the bank system. The shaded area represents the extensions to IE

As implementation proceeds and increments are completed, the actual results can be measured and captured in the IE table alongside the estimates. This allows deviations from the planned levels to be identified and future plans adjusted as appropriate.

4.0 Extending IE to Cater for Multiple Stakeholder Viewpoints and Stakeholder Value

4.1 The need to improve IE

There are several issues with the basic IE table as described in Section 3. For brevity these are not all discussed here, but they include not catering sufficiently for multiple stakeholder viewpoints and not explicitly capturing the stakeholder value associated with each performance requirement. In other words, each performance requirement can be said to express what the development project has to achieve, but fails to represent the resulting different stakeholder values across the different stakeholders. For example, reducing the time taken to update the business rules from 1 month to 1 day has a value to the Back Office, the business units, the rule administration and the regulator. The precise value differs depending on the stakeholder.

4.2 A proposed simple extension to IE

To address the multiple stakeholder viewpoints and stakeholder value problems, we decided to extend the basic IE table to the left-hand side to capture the different stakeholder viewpoints of stakeholder value, see the shaded areas of Figure 4. The values given in the stakeholder columns represent the financial value of the estimated time saving resulting from achieving 100% of the requirement.

This extension in turn enabled cumulative stakeholder value to development cost ratios to be calculated for the different designs as shown in the bottom row of Figure 4. This is closer to a measure of the return on investment than before when measuring the performance to cost ratios. There is, however, still an additional issue in that the utility graph for increasing value against performance is not always linear. Especially in cases where 100% of a high-value requirement was not reached very rapidly, there would be a need to determine the utility graphs by stakeholder. Despite this, for the bank system this was not considered a major issue as 100% levels for all the performance requirements in the IE table are likely to be achieved within four increments. Looking at the results for the value to cost ratios, it appears that maybe the implementation order of "web self-service" and "automate rules and automate testing" should be reversed.

5.0 Conclusions

In this paper, we have discussed how Planguage specifies and utilises performance attributes, in particular the quality attributes. We have shown how Planguage integrates the performance attributes into the system model, and how they can be used within IE for prioritization and other planning, and control purposes. In addition, we have outlined an extension to IE, and argued that it improves support for multiple stakeholder viewpoints and stakeholder value, and so enables enhanced prioritization using explicit stakeholder value.

It is the inclusion of metrics - quantified (*numeric*) system information - that is the crucial contribution to support the evaluation of system quality (the assessment of the impacts of different designs on the performance attributes). Further, it is the use of metrics that allows the expression of explicit stakeholder value.

The overall result of using Planguage is that an IT project is not simply discussing what functionality it intends to implement for specific resource expenditure, it is also discussing what performance/quality levels it will deliver. In addition to these planning considerations, there is also the ability for enhanced project control - to monitor delivery of performance levels against estimations. Further, by enhancing IE, the possibility of strengthening the evaluation of what stakeholder value an IT project will deliver is being opened up. By using Planguage metrics, system quality can be much more comprehensively addressed.

6.0 References

1. Barbacci M, Klein M, Longstaff T & Weinstock C, *Quality Attributes*. Software Engineering Institute 1995. Technical Report CMU/SEI-95-TR-021 ESC-TR-95-021
2. Chung L, Nixon, B, Dealing with Non-Functional Requirements: Three Experimental Studies of a Process-Orientated Approach, *Proceedings of the 17th International Conference on Software Engineering* (ICSE ’95), pp 25-37, Seattle, Washington USA, 1995
3. Woodward S (2003). Quantified Objectives: Managing Software Development Projects, *Methods and Tools*, Fall 2003. This article can be found at: www.methodsandtools.com/archive/archive.php?id=6 (visited February 2009)
4. Firesmith, D (2004). Prioritizing Requirements, *Journal of Object Technology*, September-October 2004, Vol. 3, No. 8, 35-47. This article can be found at: http://www.jot.fm/issues/issue_2004_09/column4 (visited February 2009)
5. Regnell B, Svensson R & Olsson T (2008). Supporting Roadmapping of Quality Requirements, *IEEE Software*. 42-47.
6. Gilb T, *Competitive Engineering*. Elsevier Butterworth-Heinemann 2005, ISBN 0-7506-6507-6
7. Gilb T, *Principles of Software Engineering Management*. Addison Wesley 1988, ISBN 0-2011-9246-2
8. US Department of Defense, *MIL-STD 499B: Systems Engineering*, 1994. Available at: http://www.afit.edu/cse/docs/guidance/MS499BDr1.pdf. (visited February 2009)

TOWARDS A RATIONAL PRIORITIZATION PROCESS FOR INCREMENTAL AND ITERATIVE SYSTEMS ENGINEERING

LINDSEY BRODIE
School of Engineering and Information Sciences, Middlesex University, London, UK

MARK WOODMAN
School of Engineering and Information Sciences, Middlesex University, London, UK

Prioritization is a fundamental part of requirements analysis, but the current prioritization processes implemented within existing prioritization methods are underdeveloped. As a result, many projects often pay only lip service to carrying out prioritization and fail to capture adequate prioritization data during requirements specification. This shortfall becomes more pressing given the recent adoption within many IT projects of incremental and iterative methods, which makes prioritization more important (as it is now needed for each increment). Specifically, delivery of stakeholder value should be a prime concern and yet the published prioritization methods identified to date all fail to demand any direct estimates of stakeholder value. For example, the Planning Game in Extreme Programming (XP) relies on ratings of functionality by how essential the functionality is to the project (termed 'value'). This paper outlines the requirements for a prioritization process and briefly reviews the shortfalls in several existing prioritization methods.

1 Introduction

1.1. The Need for Reassessment of the Prioritization Process

In 1995, Zave [1] identified "understanding priorities and ranges of satisfaction" as an issue for requirements engineering to address. Since then a considerable body of research work has been carried out addressing prioritization methods. However, we propose that there are flaws in many of the current approaches to prioritization and that the fundamental nature of the prioritization process needs re-examining to enable progress.

In some ways the prioritization process can be considered as a 'gap' in systems development. Certainly within most of the commonly used systems development methods, prioritization can be considered to have had far too low a profile in the past. Also industry standards such as the Integrated Capability Maturity Model (CMMI) [2] and SWEBOK [3] do not offer any specific guidance on the prioritization process.

In the literature, the discussion of prioritization is fragmented over several distinct areas. These areas include requirements prioritization [4] [5] [6], release planning [7], architecture selection [8], financial management [9] [10], and decision-making and negotiation methods [11]. Little discussion of an overall prioritization process integrating these areas has been found to date; this is a weakness in systems development. Even if, in practice, a series of distinct sub-processes were recommended for prioritization, it would be helpful if there were an umbrella process consolidating an approach to prioritization, as it is unlikely that industry would adopt numerous distinct prioritization methods. In any case, a more consolidated and integrated approach to prioritization is possible – as we shall show later in this paper.

Numerous papers discuss and compare prioritization methods, for example [4] [5] [6]. The stance taken in this paper differs because the main focus is on the expression of stakeholder value and the support that using metrics can provide.

1.2. The Drivers for Improving the Prioritization Process

The main drivers for an improved prioritization process in systems development include:

- Identifying stakeholder value
- Supporting iterative and incremental development [12]
- Improving support for stakeholder communication when negotiating tradeoffs.

Note a stakeholder is any interested party in a system: for example, the customers, the software developers, maintenance, marketing, or the product retailers. The term "stakeholder" as used in this paper denotes any relevant grouping of stakeholders, usually by role or by role and location.

Delivering stakeholder value must be seen as the ultimate aim of IT projects. However, as Boehm and Sullivan [13] point out software developers fail to address "value added" in adequate depth when designing systems. They suggest the need for further research into principles, models, methods and tools for the "dynamic management of software development as an investment activity" and "for resolving multi-attribute decision issues in software design and development." To date there has been little progress. Sullivan [14] remarks on a lack of "formal, testable and tested theories, methods, and tools to support economic-based analysis and decision-making (and value-based analysis more broadly)."

Supporting incremental and iterative development becomes increasingly important as use of agile methods grows within IT projects [15]. In addition to the increased emphasis on identifying stakeholder value at each increment, the main additional needs within incremental and iterative development for prioritization are seen as support for dynamic prioritization and improved handling of the requirements' and designs' interdependencies.

Improving support for stakeholder communication has to be addressed in order to ensure stakeholders are provided with relevant, timely information to help their decision-making. The issues involved encompass both the decision-making process in the system's environmental context (including consideration of the organizational structures), and the quality of the data involved in the prioritization process. For example, are the results of existing prioritization methods misleading to the stakeholders?

1.3. Existing Prioritization Methods

To date, we have identified more than 60 prioritization methods in the IT literature. However, it is not clear to what extent these methods are used in industry, or indeed how successful they have been [16]. Indeed, there appear to be some problems with the take-up and continued use of the well-known prioritization methods, such as Quality Function Deployment (QFD) [17] and Analytic Hierarchy Process (AHP) [18].

We consider the main flaw in the existing prioritization methods as being that stakeholder value is often expressed indirectly. For example, it is often treated within the umbrella term of 'importance' [6] [4] or by means of requirements being classified using such terms as 'mandatory', 'desirable' or 'inessential' [19 quoting Brackett 1990]. The majority of these terms are expressing stakeholder value in an ambiguous way. The problem is compounded when there is no means of capturing multiple stakeholder values [6] [5] or inappropriate aggregation and/or weighting of stakeholders or stakeholder values is carried out. Such methods fail to cater adequately for stakeholders having different system viewpoints and/or system usage.

To further discuss the above and additional problems, subsequent sections of this paper identify the requirements for a prioritization process and then consider how a selection of the more commonly used existing prioritization methods measure up to meeting these requirements.

2 Requirements for a Prioritization Process

The requirements for a prioritization process have been classified below using two categories. There are the different prioritization *factors* that have to be taken into account within the prioritization process (sometimes termed criteria [20] or aspects [21] [16], and there are the needs associated with *structuring the prioritization data* to enable the subsequent, relevant prioritization processing.

2.1. Prioritization Assumptions

First, let's briefly consider some basic assumptions that we need to make about the need for prioritization and its practice:

- Prioritization to identify stakeholder value is required because either there are limited resources (such as budget, timescales and/or staff availability) and/or incremental and iterative development is involved
- There's agreement that a rational prioritization process shall be followed, maybe to provide evidence and explanation to support an intuitive decision
- The amount of effort expended on the prioritization process shall be proportional to the level of IT investment and/or the stakeholder value involved
- Stakeholder conflicts over requirements and/or designs are assumed as a given. There is no assumption that stakeholders will always be able to negotiate an agreed solution. Stakeholder value hopefully will differentiate, but if not, it will come down to strategic alignment and ultimately to senior management decisions/tradeoffs

- No prerequisite requirement for optimum solutions exists; any satisfactory solution will suffice unless otherwise determined.

Perhaps the most significant of these assumptions is the one concerning the prioritization process being accepted as a rational process. The implication of this assumption is that prioritization decisions are based on hard data and that the prioritization data justifies the decisions made, or at least provides evidence supporting the decisions made.

2.2. Prioritization Factors

The prioritization factors discussed in the literature [4] [5] [7] [18] [20] [22] [23] include:

Stakeholder opinion: This factor is concerned with personal opinions, preferences, biases and intuition.

Strategy: Strategic alignment, product strategy, architectural strategy and competitive strategy can all be taken into account.

Time: Urgency, time to market, short-term versus long-term timescales.

Legal obligations: There can be mandatory legal obligations that have to be met.

Financial value: Potential financial benefits (or on the negative side, financial penalties), cost/benefit ratios and the cost of not implementing can all be assessed.

Cost: There are many different types of cost to consider: for example, development costs, installation costs and operational costs. These should be associated with specific designs.

Fit: This is the fit with existing components, such as existing products, current staff skill levels, current business processes, existing organizational culture and current workload.

External dependency: There can be dependencies with external organizations. These may be customers and/or suppliers.

Risk: Risk can take many forms: for example, business risk, sales barriers, volatility of requirements, the degree of change involved and technical risk. It depends on individuals and/or the organization's attitudes towards risk as to how much risk can be considered acceptable. Often risk is seen as a negative concept, though there is an emerging acceptance that risk can be seen as positive – and that 'No risk, no gain' is often the case when pursuing high gains.

Not all factors are relevant to all types of stakeholder: the different stakeholders will have different viewpoints on each of these factors and they will have expertise in different areas. For example, a product's price impacts the customers and the systems developers in different ways: it is a customer's cost, but it also impacts the systems developers' profit margin or cash flow. To give another example, assessing the contribution towards competitive strategy will most likely be the responsibility of Marketing, while the architectural strategy will be the responsibility of the systems architect.

Tentatively, the prioritization factors can be seen as mapping to the different types of stakeholder value. Certainly 'strategic value' and 'financial value' exist. Further research work is needed to determine if such a mapping proves useful when considering stakeholder value.

2.3. Structuring the Prioritization Data

An initial list of requirements for structuring the prioritization data to support the prioritization process has been identified by extrapolating from discussions in the literature, for example from [6] [22] [5] [7] [23] [18] [24] [25]. The requirements include:

Explicit stakeholder value: Expression of stakeholder priority to capture the stakeholder value. Also the capture of any associated stakeholder value and the rationale supporting the choice made. Utility functions [26] are likely to be needed to express how stakeholder value changes with the requirement level. We consider more explicit stakeholder value should be captured.

Multiple stakeholder viewpoints: There is a need to handle different areas of interest/expertise and capture the different priorities together with the associated stakeholder values. We consider mapping to organizational structure (business unit/divisional and hierarchical organizational structure) including location data is an additional consideration. The ability to identify and flag areas requiring work and/or stakeholder conflicts/overlaps requiring further detailed negotiation is also needed.

Links to business processes: The mapping of requirements to business processes (Note this is not considered further in this paper).

Links to programme/portfolio/other projects: There is a need to understand how other proposed IT investment interacts with the project under consideration. There should be no unknown counting of any proposed stakeholder value more than once and a project needs to be alerted if there is a risk of substantial change by any another activity in an area within its scope of development or influence.

Performance attributes: Failure to handle the performance attributes (often known when requirements as non-functional requirements (NFRs)) is seen as a major drawback for a prioritization method. Changes in the levels of the performance requirements are typically seen as the major system requirements dictating design, so failing to capture how they impact stakeholder values means the loss of significant system information.

Integrated functionality and performance attributes: As the previous bullet point. Few prioritization methods enable appropriate integration of functional and performance requirements and yet this is significant as performance levels are often not required to be set at uniform levels across all the system functionality.

Requirement interdependencies: The ability to express interdependencies among requirements. This becomes increasingly important with incremental and iterative development.

Design interdependencies: The ability to express interdependencies among designs. This becomes increasingly important with incremental and iterative development.

Increment handling: There is a need to capture the mapping of requirements and designs to increments. In other words to identify the outcomes of the prioritization process.

Results/feedback data: the ability to capture the actual data after implementation into operational use has occurred.

Requirements abstraction: the ability to handle requirements at different levels of abstraction in a manner that assists prioritization processes.

Dynamic prioritization: the priority data must be captured in order that it can be reused in subsequent prioritizations without needing further inputs from stakeholders unless something significant has changed in the system and/or its environment.

Scaling up: the ability to scale up to cope with large numbers of requirements. Some existing prioritization methods become impractical when the number of requirements begins to grow. In fact, for most large-scale projects, there probably is no need to go into the detailed level of handling every single requirement in the prioritization process as it is likely any analyst would get lost in the detail and expend too much time and effort.

3 Handling of Prioritization within Selected Existing Prioritization Methods

In this section, we consider how the existing prioritization methods meet the prioritization process requirements. Specifically, we examine the handling of the prioritization factors and the requirements for the structuring of the prioritization data discussed in the previous sections. For brevity, a selection of only 7 out of the 60 existing prioritization methods identified to date in the literature is discussed, and only brief descriptions of these methods are given below. The prioritization methods were selected on the basis of representing different categories of prioritization method. They are as follows:

MoSCoW [27]: MoSCoW separates requirements into ordinal groups of 'must have', 'should have', 'could have' and 'want to have but will not have this time round'.

Requirements Triage [28]: "Triage is the process of determining which requirements a product should satisfy given the time and resources available." The aim is to establish what requirements need to be in the next baseline, what requirements will not be in the next baseline, and which are optional and should be triaged - "requirements that the product could incorporate but that the development team must first carefully weigh against available resources". These requirements have to be prioritized by relative importance and any interdependencies noted, the resources needed for each of the requirements have then to be estimated and any resource interdependencies noted, and then a subset of the requirements has to be chosen that "optimizes the probability of the product's success in its intended market". Relative importance is determined using the 100-dollar test [4] or by the stakeholders voting for each requirement on whether it should be included or excluded based on how useful they see a requirement.

Analytic Hierarchy Process (AHP) [29]: In AHP, an aim and a criteria hierarchy are set up and then some potential designs (solutions) are identified. A set of comparison values is selected, typically in the range of 1-9 with 9 being the highest importance. The meanings are 1= equal importance, 3=slightly more important, 5=essentially more important, 7=demonstrated importance, 9=extremely more important [5]. Using the comparison values, pair-wise comparisons are worked through for all the criteria, then for the sub-criteria within the criteria, and this continues until the bottom of the criteria hierarchy is reached. Then normalized relative weightings are calculated for each of the criteria/sub-criteria (so the relative importance of each criteria/sub-criteria is known). Next pair-wise comparison of the designs is carried out against each of the lowest level of each branch's criteria/sub-criteria. Normalised relative values are then calculated. The normalized relative values are then multiplied by the relevant criteria/sub-criteria weighting and the results summed to give the next level's comparison figure. This multiplying and summing continues up the criteria hierarchy until the top is reached and a set of comparison values for the potential designs is obtained.

Quality Function Deployment (QFD) [30]: QFD consists of four phases, but often only the first phase known as the House of Quality (HoQ) is used. HoQ consists of a matrix structure of requirements against designs. The relative importance of the requirements is first established (AHP is sometimes used to achieve this). Then the relationship matrix is filled in by determining the impact of each of the designs on each of the requirements. The impacts are expressed using relationship values such as 'strong positive', 'medium positive', 'medium negative' and 'strong negative'. In addition to filling in the relationship matrix, the baseline values for the designs and relevant competitors' designs can be captured. The target values and additional information about difficulty of implementing and cost are also sometimes added for the designs. Additional information can also be captured against the requirements giving customer and sales preferences.

Impact Estimation (IE) [31]: IE uses a matrix structure of requirements against designs. The requirements are expressed using the performance and resource attributes of the system concerned. Each attribute is described using a scale of measure and details captured of baseline and target levels. The estimated impact of each of the designs on each of the requirements is determined and expressed as the resulting level on the scale as an absolute value and as a percentage given the baseline level is 0% and the target level is 100%. Evidence to support each impact estimate is asked for and also the uncertainty in the estimate (a credibility rating (0.0 -1.0) ranging from 'This is a guess' (0.0), to 'Another project has achieved this before' (0.8) to 'I have done this' (1.0). The percentages for performance can then be summed vertically to see the contribution of the individual designs towards meeting the requirements, and performance to cost ratios can be calculated using the data on the design's resource attributes (the costs). By considering the contribution of the designs towards meeting the requirements some idea of the level of risk involved in meeting the performance and resource requirements can be assessed.

Cost-Value Approach [32]: Karlsson and Ryan developed a cost-value approach for prioritizing requirements based on AHP. "Customers and users" carry out pair-wise comparisons of the requirements and then software engineers estimate the relative cost of implementing each requirement. AHP calculations are then carried out and a diagram showing relative value against implementation cost for each requirement is created (value is on the y-axis and implementation cost on the x-axis).

Planning Game [33]: In the Planning Game, the business people write a set of stories. Then the stories are classified by value into 'essential', 'less essential, but of significant business value' and 'nice to have'. Development people then estimate the amount of effort (story points: 1 story point = 1 day) and classify the stories by risk into 'can estimate the effort precisely', 'can estimate reasonably well' and 'cannot estimate at all'. Then development set the velocity for the implementation, and business select the scope and select the appropriate stories.

See Table 1, which shows how these prioritization methods cater for the prioritization factors identified in Section 2.2.

Table 1. Mapping of Prioritization Methods to the Prioritization Factors.

PRIORITIZATION FACTORS	PRIORITIZATION METHODS						
	MoSCoW	Requirements Triage	AHP	QFD	IE	Cost-Value Approach	Planning Game
Stakeholder opinion/preference	Y	Y	Y	Y	Y	Y	Y
Strategy	N	N	N	(Y)	(Y)	N	N
Time	N	Y	N	N	Y	N	Y
Legal obligations	N	N	N	N	N	N	N
Financial value	N	N	N	N	(Y)	(Y)	N
Cost	N	Y	(Y)	(Y)	Y	Y	Y
Fit	N	N	N	N	N	N	N
External dependency	N	N	N	N	N	N	N
Risk	N	N	N	N	(Y)	N	(Y)

Key: Y = Yes, N = No, (Y) = Some coverage

To explain the background to some of the entries in Table 1:

- MoSCoW captures the stakeholder opinion about the individual requirements at a high, aggregated level with no indication of what factors should be considered. The method has no means of specifically addressing any of the other factors
- Requirements Triage captures the importance of each requirement and estimates the effort and timescales required. The issues are seen as a lack of definition of what 'importance' consists of and, that effort is being estimated against a specific requirement rather than against a specific design
- In AHP, both requirements and designs are considered. The weighting of the requirements is carried out by pair-wise comparison and it is not clear what factors are being taken into account during the comparison. Subsequently the designs are pair-wise compared for their impact on the individual requirements
- QFD's HoQ evaluates the impact of designs on the requirements. Once again it is not specified what factors have to be considered in categorizing the impact. There is specific consideration of comparison to other competitors' products, customer and sales preferences. Also cost and ease of implementation can be addressed, though in practice this is not usually done
- IE also evaluates the impact of designs on the requirements. Given the requirements are expressed using metrics any evaluation is specifically based on asking about the resulting change – by how much does the design impact on altering the level of the

metric from the baseline level towards the target level. Development costs (and maybe other costs) are captured and performance to cost ratios are calculated. These ratios give some idea of value (and relative value among the designs) in terms of meeting the requirements. The requirements are expressed as the required target level by a specific date – so time is taken into account. If increments are being determined, more detailed consideration of elapse time is involved. Risk is handled at the level of whether the designs cover the pre-determined safety-margin. So not all risk is explicitly considered

- The Cost-Value Approach considers only the requirements. It uses AHP and so suffers from the same problems as AHP. However, the Cost-Value Approach method does explicitly consider cost and evaluates value against cost by plotting a graph of value against cost.
- The Planning Game takes much the same approach as Requirements Triage – the 'value' that is assessed is essentially the same as the 'importance' assessed in Requirements Triage – once again there is no specific definition of what should be taken into account. The Planning Game labels the ability to estimate effort as 'risk', but this is only one aspect of risk and once again, it is the effort to deliver a requirement rather than a specific design that is being assessed. It does take into account the increment loading – so timescales are being considered.

See Table 2, which shows how the prioritization methods cater for the prioritization data requirements identified in Section 2.3. To explain the background to some of the entries in Table 2:

- For expressing stakeholder priority, assignment using priority groups is often used. AHP uses pair-wise comparison and IE uses metrics expressing the required levels. IE does not use weightings as reaching all the required target levels is the aim (otherwise different targets would be set)
- Only IE handles performance attributes with functionality in an integrated way – it is the use of metrics that enables this

  Some requirements interdependencies can be captured by use of requirements hierarchies, but not all
- IE can cater for sequencing designs over increments and this enables design sequencing to reflect ordering interdependencies. QFD can capture design interactions, but in practice this is not usually carried out
- Dynamic prioritization: The 100-dollar test used in Requirements Triage would need additional input from stakeholders (redistribution of the 100 dollars) if the requirements changed. QFD suffers similar issues as it relatively weighs requirements. AHP also captures relative data and requires the input of further comparisons as requirements change to enable reprioritization, but it depends on the precise areas of the impacts of the changes made to the system – sometimes it is possible to ignore very small changes and continue using the original data. Of course, over time the data quality deteriorates.
- Scaling up: Priority groups suffer from overloading of the groups when there are large numbers of requirements – how do you prioritize among a large number of entries in one group? Also AHP's pair-wise comparison requires excessive effort as

the number of requirements grows. Methods such as QFD and IE cope with scaling up by selecting and focusing on the key requirements (so reducing the amount of data being handled).

Table 2. Mapping of prioritization methods to requirements for structuring the prioritization data

PRIORITIZATION DATA	PRIORITIZATION METHODS						
	MoSCoW	Requirements Triage	AHP	QFD	IE	Cost-Value Approach	Planning Game
Stakeholder priority	Priority groups – ordinal scale	100-dollar test or, Next Release Voting = Priority groups	Pair-wise comparisons - ratio scale	Priority groups – ordinal scale	Target levels using metrics – absolute scale	Cost-Value Diagram Pair-wise comparisons - ratio scale	Priority groups – ordinal scale
Explicit stakeholder value	N	N	N	N	N	N	N
Multiple stakeholder viewpoints	N	N	N	N	N	N	N
Links to programme/ portfolio/other projects	N	N	N	N	N	N	N
Performance attributes	N	Y	Y	Y	Y	Y	N
Integrated functionality and performance attributes	N	N	N	N	Y	N	N
Requirements interdependencies	N	(Y)	(Y)	N	(Y)	N	N
Design interdependencies	N	N	N	(Y)	(Y)	N	N
Increment handling	N	Y	N	N	Y	N	Y
Results/ feedback data	N	(Y)	N	N	Y	N	(Y)
Requirements abstraction	N	Y	Y	(Y)	Y	N	N
Dynamic Prioritization	Y	N or Y	N	N	Y	N	Y
Scaling up	N	N	N	(Y)	(Y)	N	N

From Tables 1 and 2, it can be seen that the selected existing prioritization methods do not provide much support to address the prioritization process requirements. Of the selected methods, IE provides slightly more support. IE's integrated handling of performance attributes and functionality, and its use of metrics to support evaluation of stakeholder priority are considered of particular interest. For these reasons, IE was chosen as the basis for further research work. Extensions to IE to cater for multiple stakeholders and to capture stakeholder value explicitly have now been designed (see later) and initially tested using case study data.

4 Impact Estimation and the Prioritization Process

In this section a more detailed explanation of how IE addresses the prioritization process is given. The aim is to show how IE handles Planguage metrics [31], to highlight some of the requirements for a prioritization process and to show how we have extended IE. It is based on an example from [31].

4.1. Basics of Impact Estimation

Consider the following brief example of an objective (a requirement), 'Learning', which specifies how long it should take specific users to learn to use the user interface to carry out specific tasks**.**

Learning:

Gist: Make it substantially easier for our users to learn tasks <- Marketing.

Scale: Average time for a defined [User Type: Default UK Telesales Trainee] to learn a defined [User Task: Default = Response] using <our product's instructional aids>.

Response: Task: Give correct answer to simple request.

Meter: Average time taken by defined [Number: Default = 10] of defined [User Types] to carry out the defined [User Task].

Past [Last Year]: 60 minutes.

GN: Goal [User Type = UK Telesales Trainee, User Task = Response, By Start of Next Year]: 20 minutes.

GA: Goal [UK Telesales Trainee, Response, By Start of Year After Next]: 10 minutes.

Also that you have developed some proposed alternative designs as follows:

On-line Support: Gist: Provide an optional alternative user interface, with the users' task information for defined task(s) embedded into it.

On-line Help: Gist: Integrate the users' task information for defined task(s) into the user interface as a 'Help' facility.

Picture Handbook: Gist: Produce a radically changed handbook that uses pictures and concrete examples to instruct, without the need for any other text.

Access Index: Gist: Make detailed keyword indexes, using experience from at least ten real users learning to carry out the defined task(s). What do they want to look things up under?

The IE information captured to support the Learning objective would be as given in Table 3, which shows the impacts of the design ideas on the objective. (Note the objective's data would be more summarized in an IE table and scale impact, scale uncertainty, and some other details would not appear in separate cells.) This example is a case of comparing alternative design ideas.

Table 3. Example of IE data collected and basic calculations carried out. The design idea of Picture Handbook seen as very cost-effective, but it doesn't on its own meet the goals. Maybe there is a complementary design idea that could be found? On-line Support is seen as achieving the goals (though the safety margin is not extremely comfortable) but, it is not very cost-effective compared to On-line Help and the development timescales need considering. Overall, there is a need to review the long-term strategy. Short term, On-line Help seems an ideal design idea to start considering further.

	On-line Support	**On-line Help**	**Picture Hand-book**	**On-line Help + Access Index**
Learning Past: 60 minutes <-> Goal: 10 minutes				
Scale Impact	5 min.	10 min.	30 min.	8 min.
Scale Uncertainty	±3min.	±5 min.	±10min.	±5 min.
Percentage Impact	110%	100%	60%	104%
Percentage Uncertainty	±6% (3 of 50 minutes)	±10%	±20%?	±10%
Evidence	Project Ajax: 7 minutes	Other Systems	Guess	Other Systems + Guess
Source	Ajax Report, p.6	World Report, p.17	John B	World Report, p.17 + John B
Credibility	0.7	0.8	0.2	0.6
Development Cost	120K	25K	10K	26K
Performance to Cost Ratio	110/120 = 0.92	100/25 = 4.0	60/10 = 6.0	104/26 = 4.0
Credibility-adjusted Performance to Cost Ratio (to 1 decimal place)	0.92*0.7 = 0.6	4.0*0.8 = 3.2	6.0*0.2 = 1.2	4.0*0.6 = 2.4
Notes: Time Period is two years.	Longer timescale to develop			

To cater for stakeholder value, we have extended the basic IE table to include cells capturing stakeholder value, the stakeholder value to cost ratios and the credibility-adjusted value to cost ratios. For a given stakeholder, Stakeholder A for the Learning objective:

Stakeholder Value [Financial, Stakeholder A]: Reduction in Learning Time x Average Hourly Rate of Stakeholder A x No. of Stakeholder A Staff x No. of User Tasks that Stakeholder A has to learn.

Note assumptions have been made that all user tasks take the same amount of time to learn and that the time saving per user task is the same. It is the use of metrics that enables the financial stakeholder value to be calculated. Note in addition, the impact on stakeholder value of altering the requirement target level can be discussed and assessed. See Figure 1 for an overview of the extended IE table showing how the different stakeholder values can be captured. Stakeholder value is linked to each requirement and expressed for 100% achievement of the requirement. Utility functions will be needed if the relationship between the requirement level and stakeholder value is not linear.

4.2 IE Prioritization Process including Stakeholder Value

A process to create an IE table, which includes multiple stakeholders and stakeholder value is given in Figure 2 Note this process is iterative and the sub-processes are not necessarily carried out in sequence.

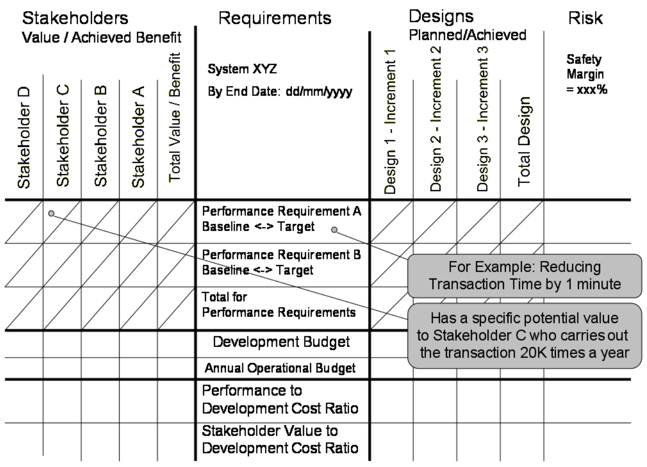

Stakeholders
Value / Achieved Benefit
Stakeholder D
Stakeholder C
Stakeholder B
Stakeholder A
Total Value / Benefit
Requirements
System XYZ
By End Date: dd/mm/yyyy
Designs
Planned/Achieved
Design 1 - Increment 1
Design 2 - Increment 2
Design 3 - Increment 3
Total Design
Risk
Safety Margin = xxx%
Performance Requirement A Baseline <-> Target
Performance Requirement B Baseline <-> Target
Total for Performance Requirements
Development Budget
Annual Operational Budget
Performance to Development Cost Ratio
Stakeholder Value to Development Cost Ratio
For Example: Reducing Transaction Time by 1 minute
Has a specific potential value to Stakeholder C who carries out the transaction 20K times a year

Figure 1. Overview of the extended IE table showing stakeholder value being captured against requirement.

5 Conclusions

This paper has shown by the evaluation of a selection of the existing prioritization methods that there is inadequate support for the prioritization process: there are significant shortfalls in the existing prioritization methods. Specifically, the general lack of concern about the ambiguous expression of stakeholder value needs to be addressed.

Unless resources are unlimited, there must always be a process by which priority decisions are made [34]. The growth of incremental and iterative development serves to reinforce the urgency for improvement: better handling of stakeholder value and dynamic prioritization are seen as essential.

To date, we regard IE as the best starting point for including stakeholder value in prioritization. IE integrates the handling for prioritization of requirements and designs with increment plans and results. The extended IE table maps requirements to stakeholders and provides an initial framework for capturing stakeholder value. This provides improved support for negotiation and decision-making. However, additional case studies are required to improve and extend the method further.

Prioritization Process for Extended IE

1. Specify the time period(s) involved. Also specify the required safety margins.
2. Identify the stakeholders.
3. Identify the main performance requirements.
4. Create a requirements hierarchy that breaks down the performance requirements into their component parts.
5. Identify/develop scales of measure for the lowest-level performance requirements.
6. Identify/develop meters to measure the levels in a practical, cost-effective way.
7. Map each performance requirement to stakeholders and determine the associated stakeholder values. Consider how the stakeholder value varies as the performance requirement's level varies.
8. Establish the baselines (current levels) for the performance requirements.

9. Consider competitors' levels and trends.
10. Set target levels with lowest acceptable and highest acceptable levels (= acceptable ranges) for the performance requirements.
11. Considering the constraints and the required functionality (ideally start with the high-level functionality), identify/develop some proposed designs. Note any interdependencies and if possible, sequence interdependent designs accordingly and/or group dependent designs into supersets (this is dependent on when the designs can deliver stakeholder value). Are any designs alternative choices?
12. Sequence designs into a potential series of increments.
13. Estimate the impact of each design on each performance requirement. Specify as a level on the scale and as a percentage (the impact percentage - using the baseline level as 0% and the target level as 100%). Give an estimate of the uncertainty in the estimate (+/- error margins) and state the credibility rating for the estimate (0.0 = wild guess and 1.0 = certainty). Document the evidence and source of the information. Multiply each estimated impact by its credibility rating.
14. Estimate the costs associated with each design.
15. Sum the relevant percentage impacts vertically for each design to assess the contribution towards meeting the requirements.
16. Calculate the performance to cost ratio for each design.
17. Calculate the value to cost ratio for each design. Where a design delivers some impact on a requirement, consider the stakeholders affected by the requirement and claim the estimated amount of stakeholder value for each stakeholder. For example, if Design D impacts 70% on a requirement that gives Stakeholder A and Stakeholder C potential benefit of 10K and 20K respectively, then you can use a utility function to work out the stakeholder values involved. Assuming a linear function and that Design D is implemented with both stakeholders, Design D can claim as stakeholder value 70% of 10K and 70% of 20K. Divide stakeholder value by cost to obtain the value to cost ratio.
18. Calculate safety deviations for each requirement by summing the estimated impacts horizontally for each performance requirement.
19. Check for sufficient design to meet all the requirements. Is the declared safety margin met for all performance requirements?
20. Also check that there are no selected designs that achieve little impact on the performance requirements – if there are, then maybe there are missing performance requirements?
21. Schedule designs with highest value to cost ratios early. Ensure any design interdependencies are met.
22. Implement the next increment.
23. As each increment is delivered, enter the actual results against the estimated results and observe any deviations. Enter any known changes and recalculate as required, then decide the next increment.

Figure 2. Prioritization process within extended IE expanded to include stakeholder value.

References

1. P. Zave, 'Classification of Research Efforts in Requirements Engineering', *Proceedings of the 2nd IEEE International Symposium on Requirements Engineering*, 214-216 (1995).
2. B. Boehm, D. Port and V.R. Basili, "Realizing the Benefits of the CMMI with the CeBASE Method", *Systems Engineering*, Vol. 5, No. 1, pp.73-88 (2002).
3. 3. P. Bourque and R. Dupuis (eds.), SWEBOK: Guide to the Software Engineering Body of Knowledge 2004 Version, IEEE Computer Society (2004).
4. P. Berander and A. Andrews, A "Requirements Prioritization" (Chapter 4), *Engineering and Managing Software Requirements*, Aurum and Wohlin (Eds.), Springer-Verlag (2005). ISBN 3540250433.
5. F. Moisiadis, "The Fundamentals of Prioritising Requirements", *Proceedings of the Systems Engineering, Test and Evaluation Conference*, October 2002 (2002).
6. J. Karlsson, C. Wohlin and B. Regnell, "An Evaluation of Methods for Prioritizing Software Requirements", *Information and Software Technology*, Elsevier Science B.V., 39, pp. 939-947 (1998).

7. D. Greer and G. Ruhe, “Software release planning: an evolutionary and iterative approach”, *Information and Software Technology*, 46, 4, pp. 243-253 (2004).
8. R. Kazman, J. Asundi and M. Klein, “Quantifying the Costs and Benefits of Architectural Decisions”, *Proceedings of the 23rd International Conference on Software Engineering (ICSE 2001)*, IEEE Computer Society, pp. 297-306 (2001).
9. S.V. Sivzattian, *Requirements as Economic Artifacts: A Portfolio-Based Approach*, Ph.D. Thesis, Department of Computing, Imperial College of Science, Technology and Medicine, University of London, October 2003 (2003).
10. J. Favaro, “Managing Requirements for Business Value”, *IEEE Software*, March/April 2002, pp. 15-17 (2002).
11. J. Park, D. Port and B. Boehm, “Supporting Distributed Collaborative Prioritization for Win-Win Requirements Capture and Negotiation”, *Proceedings of the International Third World Multi-conference on Systemics, Cybernetics and Informatics (SCI'99)*, International Institute of Informatics and Systemics (IIIS), 2, pp. 578-584 (1999).
12. C. Larman and V.R. Basili, “Iterative and Incremental Development: A Brief History”, *Computer*, June 2003, 36, 6, IEEE Computer Society, pp. 47-56 (2003).
13. B.W. Boehm and K.J. Sullivan, “Software Economics: A Roadmap”, The Future of Software Engineering, 22nd International Conference on Software Engineering, ACM Press, pp. 319-344 (2000).
14. K. Sullivan, “Introduction to the First Workshop on the Economics of Software and Computation”, In Companion to the Proceedings of the 29th International Conference on Software Engineering, IEEE Computer Society, Washington, DC, pp. 125-126 (2007).
15. M. Cusumano (Massachusetts Institute of Technology), A. MacCormack (Harvard University), C.F. Kemerer (University of Pittsburgh) and W. Crandall (Hewlett-Packard), ‘Software Development Worldwide: The State of the Practice’. IEEE Software, November/December 2003, 20, 6, 28-34 (2003).
16. L. Lehtola, *Providing value by prioritizing requirements throughout software product development: State of practice and suitability of prioritization methods*, Licentiate thesis, Helsinki University of Technology, Department of Computer Science and Engineering (2006).
17. A. Martins and E. Aspinwall, ‘Quality Function Deployment: an empirical study in the UK’, Total Quality Management, August 2001, 12, 5, 575, 14 pages (2001).
18. L. Lehtola and M. Kauppinen, “Suitability of Requirements Prioritisation Methods for Market-driven Software Product Development”, Software Process Improvement and Practice, John Wiley and Sons, Ltd., February-March 2006, 11, pp. 7-19 (2006).
19. J. Karlsson, ‘Software Requirements Prioritizing’, *Proceedings of the Second International Conference on Requirements Engineering (ICRE'96),* IEEE Computer Society Press (1996).
20. S. Barney, A. Aurum and C. Wohlin, “A product management challenge: Creating software product value through requirements selection”, *Journal of Systems Architecture*, 54, Elsevier B. V., pp. 576–593 (2008).
21. P. Berander, *Evolving Prioritization for Software Product Management*, Blekinge Institute of Technology (2007). Doctoral Dissertation Series, No 2007:07. ISSN 16532090. ISBN 9789172951082. Available via

http://www.bth.se/tek/pba.nsf/pages/8c17d1e959f1576dc12571e20031bdbd!OpenDocument/ [Last Accessed: August 2008].

22. P. Carlshamre, K. Sandahl, M. Lindvall, B. Regnell and J. Natt och Dag, "An Industrial Survey of Requirements Interdependencies in Software Product Release Planning." *Proceedings of the Fifth IEEE International Symposium on Requirements Engineering*, Toronto, 2001, pp. 84-91.
23. D. Firesmith, "Prioritizing Requirements", *Journal of Object Technology*, September-October 2004, 3, 8, 2004, pp. 35-47. http://www.jot.fm/issues/issue_2004_09/column4/ [Last Accessed: August 2008].
24. T. Gorschek and C. Wohlin, "Requirements Abstraction Model", *Requirements Engineering*, Springer Verlag, 11, 79-101 (2006).
25. N. Mead, "Requirements Prioritization Introduction", *Software Engineering Institute (SEI), Carnegie Mellon University (CMU),* (2006). See https://buildsecurityin.us-cert.gov/daisy/bsi/articles/best-practices/requirements/545.html/. Extracted and adapted from a report by Chung, Hung, Hough and Ojoko-Adams, *Security Quality Requirements Engineering (SQUARE): Case Study Phase III* (CMU/SEI-2006-SR-003).
26. J. Daniels, P. W. Werner and A. T. Bahill, "Quantitative Methods for Tradeoff Analyses", *Systems Engineering*, John Wiley & Sons, Inc., 4, 3, 190-212 (2001).
27. J. Stapleton (Editor), *DSDM: Business Focused Development* (2nd Edition), Addison Wesley (2003). ISBN 0321112245. First edition published in 1997.
28. A. Davis, "The Art of Requirements Triage", *IEEE Computer,* March 2003, 36, 3, pp. 42-49 (2003).
29. T.L. Saaty, "How to Make a Decision: The Analytic Hierarchy Process"*, European Journal of Operational Research*, 48, pp. 9-26 (1990).
30. L. Cohen, *Quality Function Deployment: How to Make QFD Work for You*, Addison Wesley Longman (1995). ISBN 0201633302.
31. T. Gilb, *Competitive Engineering: A Handbook For Systems Engineering, Requirements Engineering, and Software Engineering Using Planguage*, L. Brodie (Ed.), Butterworth-Heinemann (2005). ISBN 0750665076.
32. J. Karlsson and K. Ryan, "A Cost-Value Approach for Prioritizing Requirements", *IEEE Software*, September/October 1997, 14, 5, pp. 67-74 (1997).
33. K. Beck, *Extreme Programming: Explained,* Addison-Wesley (2000).
34. T. Gilb and M. W. Maier, "Managing Priorities: A Key to Systematic Decision-Making", Proceedings of INCOSE (2005).