The Effectiveness of Requirements Prioritization Techniques for a Medium to Large Number of Requirements: A Systematic Literature Review

Qiao Ma

A Dissertation Submitted to
Auckland University of Technology as a Part of the Requirements for the
Degree of Master of Computer and Information Sciences

November 2009
School of Computing and Mathematical Sciences

Table of Contents

List of Figures

List of Tables

Attestation of Authorship

"I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person (except where explicitly defined in the acknowledgements), nor material which to a substantial extent has been submitted for the award of any other degree or diploma of a university or other institution of higher learning."

Signature

September, 2009

Acknowledgements

It has been a long journey for me to complete this dissertation. During this time, I was encouraged and supported by a lot of people. They let me have faith to continue until this dissertation was finished.

I would like to thank to my supervisor Jim Buchan, for giving me lots of support and encouraging me a lot.

I would like to thank to my family, my friends, and everyone who helped me.

Abstract

In software system development, it can be a challenge for people to select the ‘right’ requirement among several or many options if it is not obvious which requirement is desirable. Requirements prioritization helps people to discover the most desirable requirements. It seems that most requirements prioritization techniques work well on a small number of requirements, but many of them have constraints on medium to large numbers of requirements. This directly leads to a question: are there prioritization techniques that are suitable for people to prioritize medium to large numbers of requirements? In order to find an answer to this question, this research investigates the strength of evidence for the effectiveness of different requirements prioritization techniques for medium to large numbers of requirements.

The methodology used for this research is a Systematic Literature Review. A Systematic Literature Review investigates research questions through identifying, evaluating and interpreting all relevant studies. It summarises the existing evidence for a certain technology. The reason a Systematic Literature Review was used to conduct this research is because it matches the purpose of this research, which is to systematically assess current studies in requirements prioritisation techniques as reported in literature, and analyse and draw together the results.

After conducting the Systematic Literature Review, prioritization techniques that have been applied to medium to large numbers of requirements are identified and the strength of evidence for effectiveness of each technique is evaluated. It is found that the strength of evidence for effectiveness is weak for most prioritization techniques for large numbers of requirements. More studies on prioritization techniques for large numbers of requirements are needed. Stronger evidence presented for prioritization techniques for medium sized numbers of requirements shows the techniques are more mature. However, all the studies in the medium-size category use a subjective measure of improvement based on the users’ perceptions of level of improvement. It seems that the evaluations are still not strong for these studies.

1. Introduction

Making decisions among several options is common through people's lives. People may decide what kind of food they will eat at their lunch time; people may decide which movie they will watch on their next Saturday evening; or people may decide what kind of vegetables they will prepare for their dinner. When people have several options and it is not obvious which option is the best one, decisions can become hard to make. For example, when purchasing a cell-phone, it is relatively easy to make a choice if only considering one desirable function, since one only needs to evaluate which cell-phone contains the desirable function. But when considering more aspects such as function, quality, price, and size, the decision might become harder to make, since a cell-phone with desirable functions and quality may not have desirable price and size, or a cell-phone with desirable functions, price and size may not be of desirable quality. One way to make the right decision is to prioritize the different options.

In the software development process, making decisions among several or many options is also very common. Projects are often faced with constraints such as budgets, time to market and human resources. Within these constraints, projects often cannot implement all the requirements in one product release. When projects contain more requirements than can be implemented in one product release, stakeholders need to make decisions on which requirements need to be implemented first.

Firesmith (2004) says that one project may contain hundreds or even thousands of requirements. Generally, not all the requirements affect users' satisfaction equally. Further, it is often not obvious which requirement strongly affects users' satisfaction among hundreds or thousands of requirements. When only one stakeholder is involved in the project, it is relatively easy to make decisions since only one stakeholder's opinion needs to be considered. When more than one stakeholder is involved in the project, decisions can be harder to make, since different stakeholders have different perspectives. For example, project developers look for the requirements which can be implemented fast, financial managers look for the requirements with low cost, market managers look for the requirements with high market value, and end users look for the requirements which are easy to use. One requirement may have a low cost, and short time to be implemented, but also have low market value and be hard to use. Conversely, one requirement may have a high cost, but a short time to be implemented, high market value and ease of use. It can be a challenge for stakeholders to decide which requirements lead to high stakeholders' satisfaction and need to be implemented first. Requirements prioritization is an approach that can uncover the most important requirements to maximize the stakeholders' satisfaction. Ngo-The and Ruhe (2005) note that requirements prioritization has been recognised as one of the most important decision making processes in the software development process.

Due to the importance of requirements prioritization, numerous methods on how to prioritize requirements have been developed such as AHP, cost-value approach, simple ranking, hundred dollar method, and minimal spanning tree. These methods contribute a lot to software development. But through further study, it is found that most prioritization methods work well on small numbers of requirements, but many of them seem to have constraints for people prioritizing medium to large numbers of requirements (for details, please refer to section 2.3).

One project may contain a large number of requirements, and good management of these requirements can be important for making projects successful. Requirements prioritization techniques that are suitable for people to prioritize large numbers of requirements can be helpful for managing large numbers of requirements. An example is the $170 Million FBI Virtual Case File (VCF) project (Goldstein, 2005). This project involved nearly six months of requirements gathering and ultimately produced 800 pages of requirements specification. This project was finally written off as a total failure. Glenn A. Fine, the U.S. Department of Justice's inspector general, summarizes the factors responsible for the project's failure. Some of the factors are: poorly defined and slowly evolving design requirements, overly ambitious schedules, and the lack of a plan to guide hardware purchases, network deployments, and software development. Cleland-Huang and Mobasher (2008) think the VCF project's failure was at least partially due to problems in managing and prioritizing requirements.

Since nowadays project developments often face limited resources, this requires the requirements prioritization methods to be simple, easy to use and to generate accurate results. However, most prioritization methods seem to have constraints for people prioritizing medium to large numbers of requirements. They generally either can produce accurate results but are complicated, time consuming, and/or require more resources, or they can be easy and fast to perform but produce less accurate results (for details, please refer to section 2.3). Khan’s (2006) research shows that most of the prioritization methods are evaluated with less than 20 requirements. It is seen that there is a lack studies on prioritization methods for medium to large numbers of requirements. This leads to a question: are there prioritization methods that are suitable for people to prioritize a medium to large number of requirements? This question is the primary motivation for this research.

The objective of this research is to investigate the strength of evidence for the effectiveness of different requirements prioritization techniques for medium to large numbers of requirements. This research uses a Systematic Literature Review methodology to investigate the requirements prioritization techniques which have been applied to medium to large numbers of requirements.

Before further elaboration of this research, some specifications are now defined, to reduce the possibility of confusion.

- The prioritization methods here are restricted to the methods that can help people (not computers) to make decisions. It is no doubt that within current technologies, computers can easily prioritize hundreds or thousands of requirements. But a computer does not “think”, it does the work as it is coded. In many situations, computers cannot make decisions instead of people.

- To the author’s knowledge, no previous literature defines what number of requirements is medium and what number is large. Since Hatton (2007) said that it would be difficult and probably incorrect for people to rank 15 or more elements, this research treats 1 to 14 (inclusive) requirements as a “small” number of requirements, 15 to 50 (inclusive) requirements as a “medium” number of requirements, and more than 50 requirements as a “large” number of requirements.

- This research is to investigate the strength of evidence for the effectiveness of different requirements prioritization techniques for medium to large numbers of requirements. Evidence here means reasoned argument and an evaluation of some defined measures of effectiveness. Strength means rigour of the evaluation method and detail of its description.

The rest of this dissertation is organized as follows. Section 2 presents the background on the area of requirements prioritization. Section 3 provides the methodology of this research. Section 4 presents how the research is conducted, as well as the findings of this research. Finally, section 5 presents the conclusion.

2. Literature Review

Requirements prioritization is one of many critical steps in the software development process. The context of requirements prioritization is introduced first to present what the role of requirements prioritization is in software engineering and how it relates to other critical steps to produce software systems. Then some well-known prioritization methods are presented to show what the prioritization techniques are and how they prioritize requirements. Some evaluations and findings which are done by previous people are presented next, and followed with the motivation and the research question of this research.

2.1 The Context of Requirements Prioritization

The following show the context of requirements prioritization regarding its role and how it relates to other critical activities, the definition of requirements prioritization, the usefulness of requirements prioritization, and the factors that can influence the effectiveness of requirements prioritization.

2.1.1 Requirements Engineering and the Role of Requirements Prioritization

Pressman (2001) notes that software engineering is a systematic, disciplined application that encompasses processes, management, methods and tools in order to develop, operate, and maintain software. Software engineering contains vast and varied domains. Reifer (2003) lists a number of domains such as software processes, requirements engineering, reverse engineering, testing, software maintenance and evolution, software architecture, software analysis, and software design. One domain contained in software engineering is requirements engineering.

Zave (1997) says that requirements engineering is one of many domains in software engineering concerned with the real-world goals, functions and constraints on software systems. Many authors such as Bergman and Klefsjö (2003), Doerr, Hartkopf, Kerkow, Landmann and Amthor (2007), and Nuseibeh and Easterbrook (2000) think the challenge faced with software engineering is whether the software system truly reflects the customers' needs. There is no fixed solution to deal with that challenge. However, the requirements engineering process identifies stakeholders and their needs, and documents these in order to analyse them. It is designed to focus on the customers' needs and integrate the customers' needs into the newly released software system.

Requirements prioritization is one of many critical activities of requirements engineering contributing towards making good decisions for software systems. Figure 1 presents the relationship between requirements prioritization, requirements engineering and software engineering. Requirements prioritization has a close relationship with many other critical activities in requirements engineering.

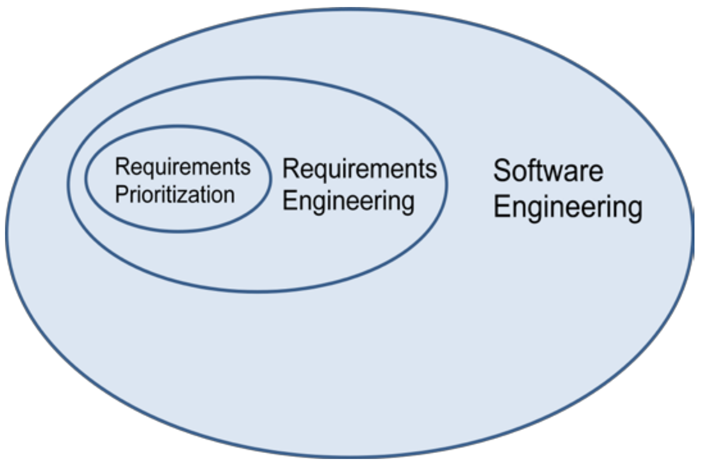


Figure 1: The relationship between requirements prioritization, requirements engineering and software engineering.

Many factors influence the process of requirements engineering, such as technical maturity, disciplinary involvement, organizational culture, application domain, and market situation. These factors make the requirements engineering process vary with different products or organizations. Therefore the activities involved in the requirements engineering process can be varied with different products or organizations. Nuseibeh and Easterbrook (2000) introduce some general requirements engineering activities, which are: eliciting requirements, modelling requirements, communicating requirements, agreeing requirements, and evolving requirements. This paper takes these activities as examples of how requirements prioritization can be involved with other critical activities.

Requirements elicitation is one critical activity in requirements engineering. Zowghi and Coulin (2005) note requirements elicitation is concerned with learning and understanding the users' and the project sponsors' needs. It is generally referred to as the first step of the requirements engineering process. Goguen and Jirotka (1994) say requirements can be collected by asking the right questions. Requirements prioritization can be performed after most requirements are elicited, in order to quickly direct resources. Requirements gathered often need to be interpreted, analysed, modelled and validated until the engineers feel that a complete enough set of requirements has been collected.

Modelling requirements is another fundamental activity in requirements engineering which constructs abstract descriptions for interpretation. It provides a method for analysing requirements. Machado, Ramos and Fernandes (2005) say it constitutes the first system representation at the early design phase. Many kinds of modelling approaches have been developed. Which modelling approach should be chosen depends on which one fits the problem best. Requirements prioritization can be performed at this stage to ensure the valuable requirements are delivered to the clients as early as possible.

Communicating requirements involves facilitating communication of requirements among different stakeholders. In order to achieve this, requirements need to be documented to ensure they can be read, analysed, written, rewritten, and validated. Requirements management is often needed to make sure the requirements are readable and traceable. Thayer and Dorfman (1997) note a variety of documentation standards have been developed to provide guidelines to achieve the readability of requirements. Gotel and Finkelstein (1994) note requirements traceability traces the life of a requirement in both forwards and backwards directions in order to analyze the consequences and impact of change. Since requirements are known better at this stage, and stakeholders may know better what they want, requirements prioritization can be performed to derive a more accurate result.

Agreeing requirements involves maintaining agreements with all stakeholders. Maintaining agreements can be difficult especially when the stakeholders have divergent goals. Requirements prioritization can be performed to help people identify the different goals held by different stakeholders. Trade-off decisions can be made during the prioritization process to achieve consensus among different stakeholders.

Software systems always evolve when the environment and the stakeholder’s requirements change. Evolving requirements involves managing changes in requirements such as adding requirements, modifying requirements, or deleting requirements. Requirements prioritization can be performed when there is a change made to requirements. During the activities of requirements engineering, requirements are studied and both clients and system developers may have a better understanding of requirements. Requirements can be added, modified, or deleted during any of these stages. These changes may result in the prioritizations done earlier becoming obsolete. Requirements may need to be reprioritized in order to update resources. Since requirements are always evolving, Lehtola, Kauppinen and Kujala (2004) suggest that requirements prioritization needs to be taken iteratively through the entire software development process to keep the resources up to date.

When should we perform requirements prioritization? This can be difficult to decide. After examining a number of software development projects, Hatton (2008) suggests that requirements prioritization can be performed whenever the project developers believe it is appropriate. If the system developers prioritize requirements too early, they will run the risk of misdirecting project time and resources since the requirements are likely to be changed. If the developers wait until very late to prioritize requirements, they risk spending more project resources on low value requirements due to failure to identify high value requirements early enough. The system developers need to find the right time to perform prioritization to maximise the benefit while minimising the risk.

Requirements prioritization should be performed iteratively through the software development process. However, using the same technique to perform requirements prioritization for different stages may result in limited benefits. Different prioritization techniques contain different properties (for details, please see section 2.2). Choosing the most suitable technique for different stages can maximise the benefits achieved. After examining a number of software development projects, Hatton (2008) advises that in the early stage of the requirements engineering process, large numbers of requirements are likely to be added from the clients' side. Clients may have a general idea of what they want, but they may not have a clear idea of what exactly they want. Requirements are likely to be changed later. Requirements prioritization can be performed at this stage in order to quickly direct resources. The prioritization techniques which are fast to perform and easy to deal with additional requirements may be the most suitable techniques to choose at this stage, and the level of detail can be sacrificed. In the mid stage, requirements are not very likely to be added but rather become more clearly defined and precise. Requirements prioritization can be performed at this stage to ensure the valuable requirements are delivered to the clients as early as possible. Prioritization techniques which are fast to perform but can provide more detailed information may be required. At the later stage, for a fairly small project, the previous prioritization may provide enough information. But for larger projects, conflicting requirements may be discovered during the design phase. Requirements prioritization can be performed to solve the conflicts. Prioritization techniques that can solve the conflicts may be required.

2.1.2 The Meaning of “Requirement”

In requirements prioritization, the object for people to prioritize is sets of requirements. Different people may see the meaning of a requirement differently in different contexts. For example, people may see a requirement as something necessary or people may see a requirement as some condition they need to meet. It is better to present the meaning of a requirement first to ensure consistency.

What does a requirement mean in the context of software engineering? IEEE 610.12-1990 standard (p. 62) provides a formal definition for a requirement. It defines a requirement as:

> (1) A condition or capability needed by a user to solve a problem or achieve an objective. (2) A condition or capability that must be met or possessed by a system or system component to satisfy a contract, standard, specification, or other formally imposed documents. (3) A documented representation of a condition or capability as in (1) or (2).

A requirement can be seen as something that a user needs or certain conditions that a system is required to meet or possess.

Machado, Ramos and Fernandes (2005) mention that since clients and system developers naturally have different points of view on requirements, requirements can be categorised as user requirements and system requirements.

Maiden (2008) refers to a user requirement as an instruction which a user provides that expresses a property of a domain or a business process that a new system will bring about. User requirements generally come from the requirements elicitation process. They are often described in natural language and mainly focus on the problem domain. User requirements are often prioritized by customers in order to know which requirements contain high values for the customers. A system requirement expresses a desirable system property that will lead to the achievement of at least one user requirement. System requirements mainly focus on the solution domain. System requirements are often prioritized by the system developers in order to determine the system implementation order.

Requirements can be categorised in other different ways. Table 1 shows some categories of requirements. Requirements prioritization can be performed on different categories of requirements to achieve different purposes.

Table 1: Types of requirements (Aurum & Wohlin, 2005, p. 4)

Requirements Classification
• *Functional requirements* — what the system will do • *Non-functional requirements* — constraints on the types of solutions that will meet the functional requirements e.g. accuracy, performance, security and modifiability
• *Goal level requirements* — related to business goals • *Domain level requirements* — related to problem area • *Product level requirements* — related to the product • *Design level requirements* — what to build
• *Primary requirements* — elicited from stakeholders • *Derived requirements* — derived from primary requirements
Others classifications, e.g. • *Business requirements* versus *technical requirements* • *Product requirements* versus *process requirements* —- i.e. business needs versus how people will interact with the system • *Role based requirements*, e.g. customer requirements, user requirements, IT requirements, system requirements, and security requirements

2.1.3 What is Requirements Prioritization?

Different people may see the meaning of requirements prioritization from different angles. Sommerville (1996) defines requirements prioritization as the activity during which the most important requirements can be discovered. Firesmith (2004) defines requirements prioritization as the process to determine the implementation order of the requirements for implementing the system or the process to determine the order of importance of the requirements to the stakeholders. Compare the definitions provided by the two authors, Sommerville (1996) mainly focuses on the importance of the requirements to the users. But sometimes requirements contain dependencies. This may result in the implementation order of a system being different from the order of importance to the stakeholders. For example, in order to implement the high value requirement A, requirement B needs to be implemented first even though requirement B is of low value in itself. System developers can prioritize requirements to see which requirement can be implement first. The definition provided by Firesmith (2004) systematically defines the meaning of requirements prioritization. This paper takes Firesmith's (2004) definition as the meaning of requirements prioritization.

2.1.4 The Usefulness of Requirements Prioritization

Why do we need requirements prioritization? The definition provided by Gilb and Maier (2005, p. 3) may answer the question. They define: "priority is relative right of a requirement to the utilization of limited (or scarce) resources." "Resources" in this definition are taken to mean all types of resources, including time, money and human resources. In other words, if the resources are unlimited, we can have all, therefore there is no need to prioritize requirements. But generally, projects face limited resources such as short timelines, small budgets, restricted human power, and limited technology. As a result, projects often contain more candidate requirements than can be implemented in one product release time. Stakeholders need to decide which requirements should be implemented. Requirements prioritization helps the project developers to select the final candidate requirements within resource constraints.

Firesmith (2004) notes that one project may contain hundreds or even thousands of requirements. Generally, not all the requirements contain equal user satisfaction. It is often not obvious which requirement contains high user satisfaction among hundreds or thousands of requirements. When only one stakeholder is involved in the project, it is relatively easy to make decisions since only one stakeholder's opinion needs to be considered. When more than one stakeholder is involved in the project, decisions can be harder to make, since different stakeholders have different perspectives. For example, project developers look for the requirements which can be implemented fast, financial managers look for the requirements with low cost, market managers look for the requirements with high market value, and end users look for the requirements which are easy use. One requirement may be of low cost, with short implementation time, but also have low market value and be hard to use. Conversely, another requirement may have a high cost, but short time to be implemented, high market value and be easy to use. It can be a challenge for stakeholders to decide which requirements need to be implemented first. Requirements prioritization is a technique that can uncover the most important requirements to maximize the stakeholders' satisfaction.

Nowadays, projects are still suffering low success rates. Chaos Report 2009 (as cited in New Dawn Technologies, 2009) releases its recent research results, indicating only 32% of all projects are "successful" in the sense they are delivered on time, on budget, and with the required features and functions. 44% are described as "challenged" meaning they are delivered late, over budget, and/or with less than the required features and functions. The remaining 24% failures, being terminated before completion or delivered but never used. Ten main factors causing challenged or failed projects are unveiled. Four of them are lack of user involvement, lack of resources, unrealistic expectations, and changing requirements and specifications. Requirements prioritization increases user involvement by letting the stakeholders decide which requirements the project should contain. It helps stakeholders hold realistic expectations by letting the stakeholders understand the current constraints on resources and accepting the trade-off decisions on conflicting perspectives. Karlsson and Ryan (1997) think it helps stakeholders allocate resources based on the priorities of the requirements. Karlsson, Wohlin and Regnell (1998) think it helps stakeholders detect requirements defects, such as misunderstanding or ambiguous requirements, to reduce the number of changes to requirements and specifications in the later stage of projects. Hatton (2007) says requirements prioritization has become an essential step in the software development process in order to reduce software failure. Ngo-The and Ruhe (2005) note requirements prioritization has been recognised as one of the most important decision making processes in the software development process.

2.1.5 Aspects of Requirements Prioritization

Berander and Andrews (2005) define an aspect as a property or attribute that can be used to prioritize requirements. Synonymous words used by other authors are "factor" (Henry & Henry, 1993) and "criteria" (Hatton, 2007). Requirements can be prioritized based on different aspects such as importance, time, cost, penalty, and risk. Some aspects are introduced below.

Importance

Stakeholders can prioritize requirements to find out which requirement is most important to them. However, the word “importance” can be a multifaceted concept which may have different meanings to different people. For example, importance could mean high market value, high quality of the product, or urgency of implementation among other things.. It is essential to specify the meaning of “importance” first to reduce the possibility of confusion when letting the stakeholders prioritize the requirements.

Time

Time can be the time spent on successfully implementing the candidate requirement. Wiegers (1999) notes time can also be influenced by other factors such as degree of parallelism in development, or staff training time.

Cost

Cost can be the money spent on successfully implementing the candidate requirement. Cost can be directly influenced by staff hours. Cost can also be influenced by other factors such as the extra resources needed in order to implement the requirements.

Penalty

Penalty is how much needs to be paid if a requirement is not fulfilled. Penalty is an important aspect that needs to be evaluated. Sometimes requirements may have low values, but failing to fulfil these requirements may cause a high penalty.

Risk

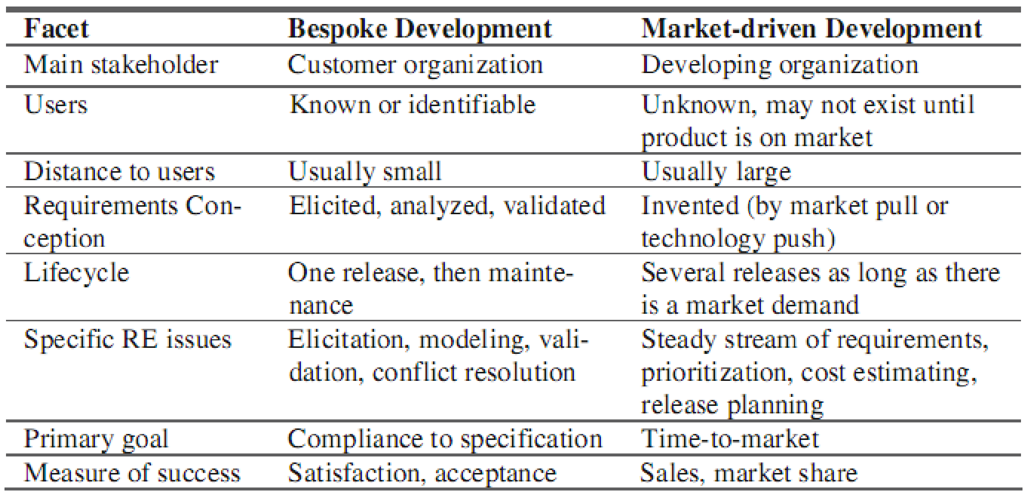
Mustafa and Al-Bahar (1991) define risk as the degree of likelihood that a project will fail to achieve its time, cost or quality goals. Each project contains a certain amount of risk. Boehm (1991) notes the risk contained in project development could be things such as personnel shortfalls, unrealistic schedules and budgets, developing the wrong functions and properties, continuing stream of requirements changes, or gold plating (adding more functionality or features than is necessary). Aurum and Wohlin (2003) say that the risk of each requirement of a project can be estimated in order to get an estimation of the risk level of the project. Risk can be prioritized to find which requirement contains the lowest risk.

Other aspects which can be considered include volatility, strategic benefit, market value, and available resources. Stakeholders can prioritize requirements based on a single or multiple aspects. Generally, prioritizing requirements based on a single aspect is easier than prioritizing requirements on multiple aspects. For example, when only considering a single aspect such as market value, a requirement that contains a high market value will have a high priority. But when considering more aspects such as cost, a high market value requirement may involve high cost. In this case, stakeholders may change their minds, and the high priority requirement may become a low priority requirement. Ruhe, Eberlein and Pfahl (2003) note that aspects are often not independent of each other, and may interact with each other (for example, high quality may require high cost). Change in one aspect may result in a change in another aspect. Since each aspect may have an influence on the degree of successful of the final product, it is essential to consider multiple aspects in order to increase the degree of success of the final product. Several aspects can be considered when prioritizing requirements, but generally it is not practical to consider all the aspects. Which aspects should be considered depends on the real situation.

2.1.6 Customers in Different Types of Markets

Two extreme types of software development markets are bespoke development and market-driven development. Regnell and Brinkkemper (2005) specify that in bespoke development (also known as customer-specific development) the product is developed based on the wishes and needs of a particular customer, and that customer pays the development cost. Market-driven development is specified as where the product is aimed to be offered to an open market, the development cost is divided among many buyers and the producer receives the potential profit. The differences between bespoke development and market-driven development are shown in Table 2.

Table 2: Differences between market-driven software development and bespoke software development (Carlshamre, 2001, p. 59)



Facet	Bespoke Development	Market-driven Development
Main stakeholder	Customer organization	Developing organization
Users	Known or identifiable	Unknown, may not exist until product is on market
Distance to users	Usually small	Usually large
Requirements Con-ception	Elicited, analyzed, validated	Invented (by market pull or technology push)
Lifecycle	One release, then mainte-nance	Several releases as long as there is a market demand
Specific RE issues	Elicitation, modeling, vali-dation, conflict resolution	Steady stream of requirements, prioritization, cost estimating, release planning
Primary goal	Compliance to specification	Time-to-market
Measure of success	Satisfaction, acceptance	Sales, market share

Bespoke development and market-driven development are two extremes in software development. Generally, real cases fall somewhere in between. Different types of markets involve different kinds of customers. Berander and Andrews (2005) show three types of general situations: single customer, several customers, and mass-market.

In a single customer situation, the product is developed for one customer, and therefore only one customer’s opinions need to be considered. The project developers only need to consider one customer’s priorities. One issue that needs to be considered is that the customer who prioritizes the requirements may not be the end user. For example, the customer could be the employer but the actual end user could be the employee. In this case, only considering the customer’s priorities may not enough since the customer’s priorities may be different from the end user’s priorities. Not considering the end user’s needs may reduce the usability of the product. One way to solve this problem is to include the end user as well when performing requirements prioritization.

In a several customers situation, customers may have conflicting viewpoints and preferences. It can be a challenge to join the different customers’ views together. The ultimate goal is to get the final priority and make every customer satisfied with the final priority. The prioritization process may need to be performed iteratively in order to get consensus between the different customers (requirements with conflicting priority need to be identified, argued, and probably reprioritized between stakeholders to get final decisions). Different techniques such as win-win (Boehm, Grünbacher & Briggs, 2001), Quantitative Win-Win (Ruhe, Eberlein & Pfahl, 2002), and EVOLVE (Greer & Ruhe, 2004) are designed to solve the conflicts between stakeholders.

In a mass-market situation, it is not possible to get all the customers to prioritize requirements. The actual customer may not even be known until he/she purchases the product. Prioritization can be done based on the analysis of market demands. Scenarios (letting people be the fictional customer to prioritize requirements) can also be used to get the users' needs.

2.2 Techniques of Requirements Prioritization

Several basic techniques on how to prioritize requirements are introduced in the following section. The prioritization techniques can be categorized as nominal scale, ordinal scale, and ratio scale.

2.2.1 Nominal Scale

For nominal scale methods, requirements are assigned to different priority groups, with all requirements in one priority group being of equal priority.

Numerical assignment

Numerical assignment is mentioned by a number of studies such as Berander and Andrews (2005), Bradner (1997), IEEE-STD 830-1998 (1998), Karlsson, Host and Regnell (2006), Leffingwell and Widrig (2000) and Sommerville and Sawyer (1997). It is a simple requirements prioritization technique based on grouping requirements into different priority groups. The number of priority groups can vary, but three is common. For example, requirements can be grouped as "critical", "standard", and "optional". The results of numerical assignment are on a nominal scale. All requirements contained in one priority group represent equal priority. No further information shows that one requirement is of higher or lower priority than another requirement within one priority group.

MoScoW

MoScoW is a kind of numerical assignment and it is mentioned by DSDM Consortium (2009), Hatton (2007, 2008) and Tudor and Walter (2006). MoScoW currently incorporates into the software development methodology DSDM (Dynamic Systems Development Method). The idea of MoScoW is that it groups all requirements into four priority groups “MUST have”, “SHOULD have”, “COULD have”, and “WON’T have”.

- “MUST have” means that requirements in this group must be contained in the project. Failure to deliver these requirements means the entire project would be a failure.
- “SHOULD have” means that the project would be nice if it contains the requirements in this group.
- “COULD have” also means that the project would be nice if it contains these requirements. But these requirements are less important than the requirements in the “SHOULD have” group.
- “WON’T have” is like a “wish list”. It means that the requirements in this group are good requirements but they will not be implemented in the current stage. They may be implemented in the next release.

The results of MoScoW are on a nominal scale. All requirements contained in one priority group represent equal priority. No further information shows one requirement is of higher or lower priority than another requirement within one priority group.

2.2.2 Ordinal Scale

Ordinal scale methods result in an ordered list of requirements.

Simple ranking

Ranking elements is quite intuitive for most people as it can happen in people’s lives. Berander and Andrews (2005) and Hatton (2008) mention simple ranking is that n requirements are simply ranked from 1…n, with the most important requirement ranked 1 and the least important requirement ranked n. This is a common requirements prioritization technique based on an ordinal scale.

Bubble sort

Bubble sort is mentioned by Aho, Hopcroft and Ullman (1983). It is a method for sorting elements. Karlsson et al. (1998) introduce this technique to the requirements prioritization area for ranking requirements. The idea of the bubble sort method for sorting requirements is that the users compare two requirements at a time and swap them if the two requirements are in the wrong order. The comparisons continue until no more swaps are needed. The result of bubble sort is a list of ranked requirements. The average and worst case complexity for bubble sort is O (n^2).

Binary search tree

Another method for sorting elements that is mentioned by Aho et al. (1983) is binary search tree. A binary search tree is a tree in which each node contains at most two children. Karlsson et al. (1998) introduce this technique to the requirements prioritization area for ranking requirements. The idea of the binary search tree method for ranking requirements is that each node represents a requirement, all requirements placed in the left subtree of a node are of lower priority than the node priority, and all requirements placed in the right subtree of a node are of higher priority than that node priority. When performing the binary search tree method, first choose one requirement to be the top node. Then, select one unsorted requirement to compare with the top node. If that requirement is of lower priority than the top node, it searches the left subtree, but if that requirement is of higher priority than the top node, it searches the right subtree. The process is repeated until no further node needs to be compared and at that time the requirement can be inserted into the right position. The average complexity for binary search tree is O (n log n).

Simple ranking, bubble sort and binary search tree methods are all used for ranking requirements. Simple ranking method is quite intuitive for people, bubble sort and binary search tree methods seem harder for people to use to rank requirements. One question which may come out is that if people can do simple ranking easily, why are bubble sort and binary search tree methods needed? The answer is that when there are a fairly small number of requirements needing to be prioritized, simple ranking seems easy for people to perform. But as the number of requirements increases, people may have difficulty remembering all the requirements. Psychology research (reviewed by Miller (1956)) shows that people have difficulty remembering more than seven (plus or minus two) elements. Hatton (2007) said that it would be difficult and probably incorrect for people to use the simple ranking method to rank 15 or more elements. If a large number of requirements need to be ranked, in order to get a high degree of accuracy, bubble sort and binary search tree seem more suitable than simple ranking.

2.2.3 Ratio Scale

The results of ratio scale methods can provide the relative difference between requirements.

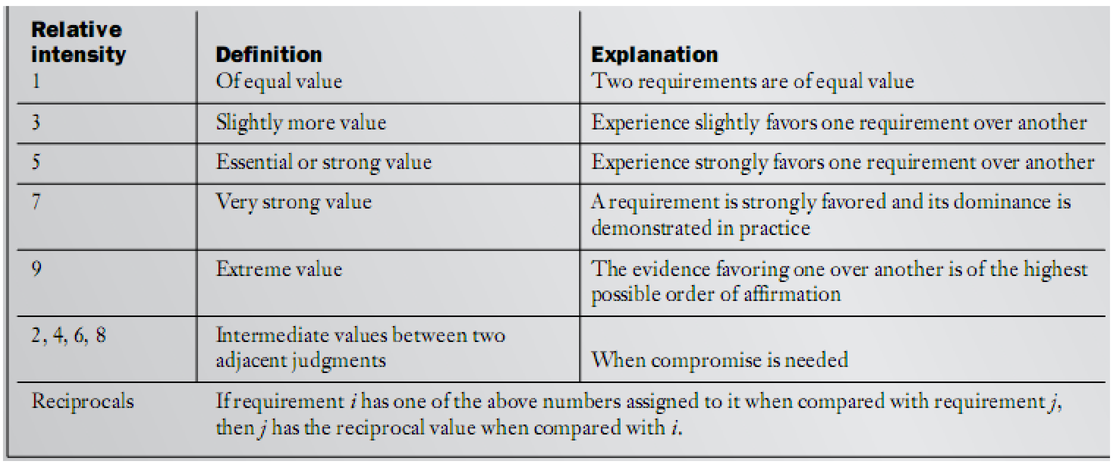
Hundred Dollar Method

Hundred dollar method (also called cumulative voting) which is mentioned by Berander and Andrews (2005) and Hatton (2008) is a simple method for prioritizing requirements. The idea of the hundred dollar method is that each stakeholder is asked to assume he/she has $100 to distribute to the requirements. The result is presented on a ratio scale. The ratio scale result can provide the information on how much one requirement is more/less important than another one.

AHP

Another well-known prioritization technique based on ratio scale results is called Analytic Hierarchy Process (AHP). AHP is developed by Saaty (1980) and it is designed for complex decision making. The idea of AHP is that it compares all possible pairs of hierarchical requirements to determine the priority. When using AHP, the user first identifies the attributes and alternatives for each requirement and uses them to build a hierarchy. Then the user specifies his/her preference to each pair of the attributes by assigning a preference scale which is generally 1 to 9, where 1 indicates equal value and 9 indicates extreme value. The scale is shown in Table 3. After that AHP converts the user's evaluations to numerical values and a numerical priority is derived for each element of the hierarchy. Note that a redundancy might exist when using the AHP method to prioritize requirements, therefore a consistency ratio should be calculated after using the AHP method to judge if the prioritization is valid. If n requirements need to be prioritized, $n*(n-1)/2$ pair-wise comparisons are required when using the AHP method. Therefore the complexity of AHP is $O(n^2)$.

Table 3: Fundamental scale used for AHP (Karlsson & Ryan, 1997, p.70)



Relative intensity	Definition	Explanation
1	Of equal value	Two requirements are of equal value
3	Slightly more value	Experience slightly favors one requirement over another
5	Essential or strong value	Experience strongly favors one requirement over another
7	Very strong value	A requirement is strongly favored and its dominance is demonstrated in practice
9	Extreme value	The evidence favoring one over another is of the highest possible order of affirmation
2, 4, 6, 8	Intermediate values between two adjacent judgments	When compromise is needed
Reciprocals	If requirement *i* has one of the above numbers assigned to it when compared with requirement *j*, then *j* has the reciprocal value when compared with *i*.	

Empirical studies performed by Karlsson and Ryan (1997) and Karlsson et al. (1998) show that AHP is time consuming. Some techniques try to reduce the number of comparisons in order to reduce the time consumed. Hierarchy AHP and minimal spanning tree have been developed for that purpose.

Hierarchy AHP

Davis (1993) notes that in large projects, requirements are often structured in a hierarchy, with the generalized requirements placed at the top of the hierarchy and the more specific requirements placed at the lower levels of the hierarchy. Hierarchy AHP, which is introduced by Karlsson et al. (1998), uses the AHP method to prioritize requirements only at the same level of hierarchy. This method can reduce the number of decisions compared with the AHP method, since not all the requirements are compared pair-wise. This can reduce the number of redundant comparisons, but the trade-off is that the ability to identify inconsistent judgments is also reduced.

Minimal Spanning Tree

Minimal spanning tree is another prioritization method which is introduced by Karlsson et al. (1998). The idea of minimal spanning tree method is that if the decisions are made perfectly consistent, the redundancy will not exist, and in this case the number of comparisons will reduce to only n-1 comparisons (n is the number of requirements). A minimal spanning tree constructs unique pairs of requirements. It is a directed graph which is minimally connected. Minimal spanning tree can reduce the number of pair-wise comparisons dramatically compared with AHP. However, the ability to identify inconsistent judgments is low.

Cost-Value Approach

Karlsson and Ryan (1997) provide a method which is called the Cost-Value approach for prioritizing requirements. The basic idea of the Cost-Value approach is that each individual requirement is determined on two aspects: the value to the users and the cost of implementing the requirement. It uses the AHP technique to compare requirements pair-wise according to the relative values and costs. Empirical studies performed by Karlsson and Ryan (1997) show that the Cost-Value approach is time consuming.

2.2.4 Combining Techniques

Planning Game

Beck (1999) introduces a prioritization method, named Planning Game, which is based on a combination of prioritization techniques. Planning Game is mostly used in agile projects. The idea of Planning Game is that it combines the numerical assignment technique and ranking technique together to perform the requirements prioritization. Requirements are first prioritized into three groups: (1) those without which the system will not function, (2) those that are less essential but provide significant business value, and (3) those that would be nice to have. After assigning the requirements into three groups, requirements are simply ranked in each group.

2.2.5 Tools

Some tools have been developed to make the prioritization process easier for people, such as Tool-Supported PWC and Case-based ranking. Tool-Supported PWC which is mentioned by Karlsson, Thelin, Regnell, Berander and Wohlin (2007) is a tool to facilitate the use of the pair-wise comparison technique. Avesani, Bazzanella, Perini and Susi (2005) present a novel framework called Case-based ranking to overcome the scalability issue. The idea of the Case-Based Ranking method is that the system first lets the users do a limited elicitation effort, and then the system uses machine learning techniques to generate an approximation of the final ranking.

It is also possible to use some simple tools (for example, spreadsheets) to make the prioritization process easier or faster.

2.2.6 The Level of Detail

Generally speaking, ratio scale results provide more detail than ordinal scale results, and ordinal scale results provide more detail than nominal scale results. For nominal scale methods, all requirements in one priority group represent equal priority. No further information shows that one requirement is of higher or lower priority than another requirement within one priority group. Ordinal scale results can provide an important ordered list of requirements, but they cannot show how much one requirement is more/less important than another requirement. Ratio scale results can provide the relative difference between requirements.

Prioritization methods that produce highly detailed results may be complicated and time consuming. Some simple and easy to use methods may only produce less detailed results. The project developers need to make the decision on how “quick and dirty” the approach can be without giving up the quality of the decisions.

2.3 Techniques Evaluation

The following section presents the findings from previous authors’ evaluations of some prioritization techniques.

2.3.1 The Evaluation of AHP, Hierarchy AHP, Spanning Tree, Bubble Sort and Binary Search Tree Methods

Karlsson et al. (1998) perform an experimental study to evaluate six prioritization methods: analytic hierarchy process (AHP), hierarchy AHP, spanning tree matrix, bubble sort, binary search tree, and priority groups.

The priority groups method was not introduced in section 2.2 because it is different from general grouping requirements. For more details on priority groups method and why it was not introduced, please refer to Appendix A.

Karlsson et al. (1998) use 13 quality requirements to evaluate the six prioritization methods. These requirements are prioritized by three authors independently. Each prioritization method is assigned randomly to each author. Each author only studies one method per week in order to minimize the risk of remembering the priorities of the requirements. Only importance to customers is considered as an aspect when prioritizing requirements.

Two kinds of measurements are evaluated: objective measures and subjective measures. An ordinal scale from 1 to 6 is used for the measurement, where 1 indicates the best and 6 indicates the worst. The evaluation results are shown in Table 4 and Table 5.

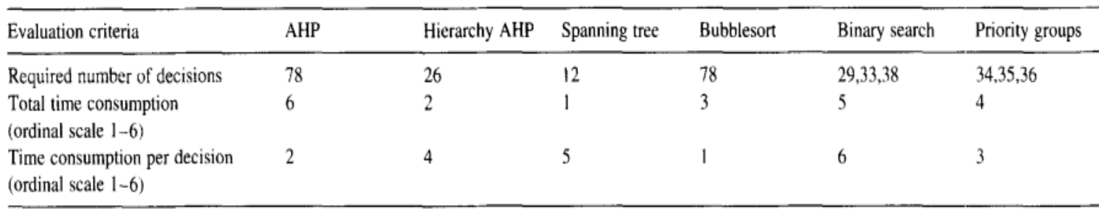
The objective measures are: required number of decisions, total time consumption, and time consumption per decision.

- Required number of decisions measures the total number of decisions which need to be made for each prioritization method. The numbers of decisions for the first four methods are pre-defined. The numbers of decisions for the last two methods are counted by each author (therefore three numbers are presented in the result shown in Table 4).
- Total time consumption compares the time spent to complete the prioritization process among the six methods.
- Time consumption per decision compares the average time spent on each decision for each prioritization method among the six methods.

The subjective measures are: ease of use, reliability of results, and fault tolerance.

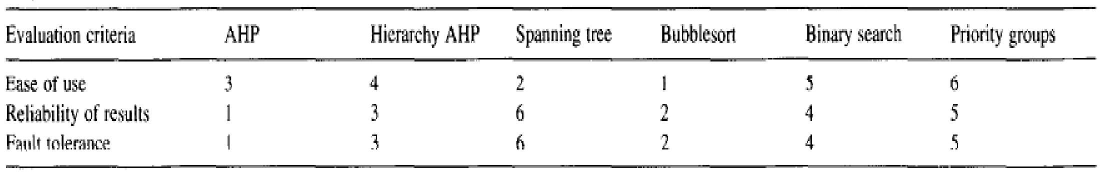
- Ease of use measures how easy a particular prioritization method is to use.
- Reliability of results measures how reliable the result of a particular prioritization method is.
- Fault tolerance means the ability to identify inconsistent judgments.

Table 4: Objective measures (Karlsson et al., 1998, p. 945)



Evaluation criteria	AHP	Hierarchy AHP	Spanning tree	Bubblesort	Binary search	Priority groups
Required number of decisions	78	26	12	78	29,33,38	34,35,36
Total time consumption (ordinal scale 1–6)	6	2	1	3	5	4
Time consumption per decision (ordinal scale 1–6)	2	4	5	1	6	3

Table 5: Subjective measures (Karlsson et al., 1998, p. 945)



Evaluation criteria	AHP	Hierarchy AHP	Spanning tree	Bubblesort	Binary search	Priority groups
Ease of use	3	4	2	1	5	6
Reliability of results	1	3	6	2	4	5
Fault tolerance	1	3	6	2	4	5

From the results provided in Table 4 and Table 5, it is seen that AHP can provide the most reliable result of the six methods, but it requires the largest number of decisions and the longest time consumption. Minimum spanning tree involves the smallest number of decisions and the shortest amount of time consumption, but it provides the least reliable result and the lowest fault tolerance. Bubble sort is the easiest method to use and it can provide relatively reliable results and relatively good fault tolerance, but it involves the largest number of decisions (same as AHP). Hierarchy AHP and binary search tree reside in the middle. They produce less reliable results than AHP and bubble sort, but also take fewer decisions and less time to perform than AHP and bubble sort.

It is seen that no one prioritization method is perfect among these six methods. Minimum spanning tree requires less effort and time to perform the prioritization process, but it contains a high risk of misdirecting project resources and time since it provides low reliable results. AHP and bubble sort methods can provide reliable results, but they need large amounts of effort and time to perform. When dealing with a small number of requirements, the amount of effort and time spent can be relatively small. But since the complexity of the AHP and bubble sort methods is high (both are O (n^2)), when dealing with large numbers of requirements the amount of effort and time spent may become unmanageable. Karlsson et al. (1998) also admit that AHP and bubble sort both contain a scale up problem. Among the six prioritization methods, it seems that no method is perfect for large numbers of requirements (AHP and bubble sort contain a scale up problem, and other methods contain a certain degree of accuracy problems).

2.3.2 The Evaluation of Cost-Value Method

Cost-Value approach uses the AHP method to compare requirements pair-wise according to their relative value and cost. Karlsson and Ryan (1997) use two case studies to evaluate the Cost-Value approach. One case study is performed in Ericsson’s Radio Access Network (RAN) project. Fourteen high-level requirements are prioritized. The second case study is performed in the Performance Management Traffic Recording (PMR) project. Eleven high-level requirements are prioritized. Two case studies show that the Cost-Value approach is a useful approach. However, they also find that the users find comparing all requirements in a pair-wise manner tedious. It is found that the Cost-Value approach contains a scale-up problem.

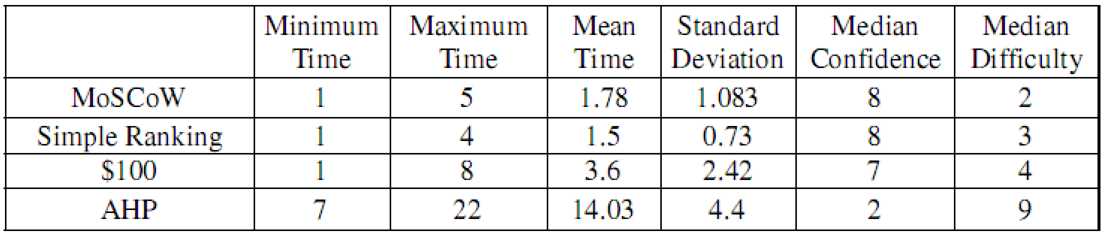
2.3.3 The Evaluation of Simple Ranking, MoSCoW, AHP and Hundred Dollar Methods

Hatton (2007) conducts a case study to examine four prioritization methods: simple ranking, MoSCoW, AHP and Hundred dollar method. Each method is examined by three criteria: ease of use, the time to complete the prioritizing process and the user's confidence (user's confidence here means how deeply the user believes the prioritization result actually reflects his/her real priority).

Twelve requirements related to mobile phone features are provided to the users. Each user is asked to use each method to prioritize requirements. Each user is also asked to record the time taken to use each method to perform the prioritization, the difficulty of each prioritization method, and the user's confidence rate for each method. A wide range of people from different ages, genders, levels of education and occupations are selected to be participants. Thirty one studies are completed and the results are used for data analysis.

The time taken for each method is recorded by letting the user record the time he/she started and finished each prioritization process. An ordinal scale from 1 to 10 is used for the measurement of difficulty of the method, where 1 indicates "very easy" and 10 indicates "very difficult". The user's confidence rate is also measured on an ordinal scale from 1 to 10, where 1 indicates "not confident" and 10 indicates "very confident". The research findings are shown in Table 6 and Table 7.

Table 6: Times taken (minutes), median confidence and median difficulty (1-10 Scale) (Hatton, 2007, p. 240)



	Minimum Time	Maximum Time	Mean Time	Standard Deviation	Median Confidence	Median Difficulty
MoSCoW	1	5	1.78	1.083	8	2
Simple Ranking	1	4	1.5	0.73	8	3
$100	1	8	3.6	2.42	7	4
AHP	7	22	14.03	4.4	2	9

Table 7: Properties of prioritization methods (Hatton, 2007, p. 242)

	Simple Ranking	MoSCoW	$100	AHP
Ratio Scale Information			√	√
High Confidence From User	√	√	√	
Consistent	√	√	√	√
Low Difficulty	√	√	√	
Low Effort	√	√	√	
Able to handle large numbers of alternatives		√		

From the results in Table 6, it is seen that AHP contains the longest completion time range of the four methods. For the mean time values, AHP takes much longer time to complete than any of the other methods. MoSCoW and simple ranking contain the highest degree of confidence and low difficulty rate. AHP contains the highest difficulty rate and the lowest degree of confidence.

The results show that the results of the four methods are all consistent (see Table 7). This means the results of the four methods accurately indicate the client’s priorities. MoSCoW is the easiest to use of the four prioritization methods. It takes less time to perform and it provides high user confidence. Simple ranking is also an easy method to use. It also takes less time to perform and it provides high user confidence. Hundred dollar method takes longer to perform and contains less user confidence than MoSCoW and simple ranking, but it is still relatively easy to use, takes less time to perform, and contains high user confidence. AHP is the hardest to use of the four methods. It takes the longest time to perform and it contains the lowest user confidence.

From the data in this study, Hatton (2007) derives a positive linear relationship between confidence and difficulty where the less difficult the prioritization method, the more confident the user. This relationship is logical and has been investigated in marketing and psychology by Macintosh and Gentry (1999). It is seen that even though some complicated prioritization methods can produce accurate results, people may not believe it if people have low confidence in it.

Hatton (2007) only uses 12 requirements to perform the research. When dealing with a relatively small number of requirements, all the four methods seem to work well (since the results of the four methods are all consistent). What about dealing with a large number of requirements?

Since people may have difficulty remembering a large number of requirements, it would therefore be difficult and probably error-prone for people to use the simple ranking method to rank relatively large numbers of requirements.

When dealing with relatively large numbers of requirements, the hundred dollar method (also be known as cumulative voting) also contains some potential problems. Berander and Svahnberg (2008) argue that the stakeholders may lose the overview as the number of requirements increases when using hundred dollar method.

Regnell, Höst, Dag, Beremark and Hjelm (2001) use an industrial case study to evaluate hundred dollar method. Ten stakeholders evaluate 17 groups of requirements. During their research, they find that hundred dollar method contains some potential problems. One potential problem is that stakeholders may assign the same priority to many requirements, especially the requirements with low priorities. This makes it difficult to discriminate among the requirements. Another potential problem is that in order to get their favourite requirement, some stakeholders might put all their money on a favourite requirement that others do not put high priority on, and in this case the total result would be highly influenced.

For AHP, the evaluation result that Hatton (2007) gets is similar to the result that Karlsson et al. (1998) get. AHP contains a high difficulty rate and takes a long time to perform. Karlsson and Ryan (1997) also find that users find making all the pair-wise comparisons between requirements tedious. AHP is complicated, and people also feel low confidence in the results of the AHP method.

Hatton (2007) also does not think simple ranking, hundred dollar method and AHP are suitable for large numbers of requirements (see Table 7). He only thinks MoSCoW suits large numbers of requirements. But MoSCoW cannot provide as meaningful a result as ordinal scale or ratio scale methods. It prioritizes requirements into different priority groups, but requirements that are in the same priority group represent equal priority; it cannot provide the relative difference between the requirements within one priority group. Some tools can help people with the scale up problem, but more resources are required and people need to spend time to learn how to use these tools.

2.3.4 Motivation for this Study

Analysing the evaluations of prioritization techniques discussed in this chapter , it seems that most prioritization techniques generally work well on small numbers of requirements, but many of them have limitations on medium to large numbers of requirements. This leads to a question: are there prioritization methods that are suitable for medium to large numbers of requirements? This question is the primary motivation for this research. The objective of this research is to investigate the strength of evidence for the effectiveness of different requirements prioritization techniques for medium to large numbers of requirements. The research question is specified as:

What is the strength of evidence for the effectiveness of different requirements prioritization techniques for medium to large numbers of requirements in the software engineering domain?

2.4 Chapter Conclusion

This chapter describes the context related to requirements prioritization. Several well-known prioritization techniques are presented to show what the prioritization techniques are and how they prioritize requirements. The research problem is justified by presenting and discussing the evaluation of prioritization techniques of previous people. Finally, the research question is presented.

The next chapter describes and justifies the selection of a Systematic Literature Review as a research methodology that is well suited to the research question.

3. Methodology

This chapter describes and justifies the research methodology used in this investigation. The research methodology should be selected according to the research aims and the first part of this chapter (3.1) explains the rationale for using a Systematic Literature Review for the research purpose identified in chapter 2. Having established the motivation for using a Systematic Literature Review, the process stages are then described in detail. The design and implementation of each methodology stage for this study is also described stage by stage. This includes a description of the data gathering techniques and data analysis approach, as suggested by the methodology.

The unit of analysis for this research is a requirements prioritization technique, and the main measure under investigation is the effectiveness of the technique, and the strength of the evidence of this effectiveness.

3.1 Justification of Methodology

The purpose of this dissertation is to systematically assess current studies in requirements prioritization techniques as reported in literature, and analyse and draw together the results. In particular the aim is to identify techniques which have been applied to large requirements sets and evaluate the evidence presented for their efficacy to draw more general conclusions than any one study. It is expected that this study will be a prelude to further research in this area. These research objectives closely align with the stated purpose of a Systematic Literature Review, which Brereton et al. (2007, p. 571) describe as “assessing and aggregating research outcomes in order to provide a balanced and objective summary of research evidence of a particular topic”. Kitchenham (2004) notes that a Systematic Literature Review investigates research questions through identifying, evaluating and interpreting all relevant studies. It summarises the existing evidence for a certain technology. So, the Systematic Literature Review provides a *systematic* and tested approach to identifying and investigating all the relevant studies to derive evidence on the effectiveness of different prioritization techniques.

Other similar studies in the software engineering domain also use a Systematic Literature Review, strengthening the case for using it in this investigation. For example, MacDonell and Shepperd (2007) conduct a Systematic Literature Review to review the current evidence for the effectiveness of two software effort estimation models. This is very similar to the objectives of the investigation proposed in this dissertation.

A comprehensive search of current literature did not reveal any similar systematic reviews in this area of requirements prioritization techniques for medium to large requirements sets, so this research should provide a valuable contribution to the body of knowledge in this area. It was expected that either the review will identify some prioritization techniques that are shown to be applicable to medium to large requirements sets, or identify that there is a need for such techniques and gathering empirical evidence of their efficacy.

The next section explains the accepted phases and stages of the Systematic Literature Review approach and describes the design and implementation for this study.

3.2 Description of Methodology

The process for undertaking a Systematic Literature Review in the software engineering domain is described and justified in some detail in Kitchenham (2004). Brereton et al. (2007) reflect on lessons learned from several Systematic Literature Reviews and conclude that the Systematic Literature Review process is relevant to software engineering studies. Many authors follow the guideline provided by Kitchenham (2004) to conduct their Systematic Literature Reviews (see for examples Brereton et al. (2007), MacDonell and Shepperd (2007), and Staples and Niazi, (2007)). This dissertation follows a synthesis of the guidelines and recommendations provided by Brereton et al. (2007), Kitchenham (2004), MacDonell and Shepperd (2007) and Staples and Niazi (2007) to conduct this review. The stages of the Systematic Literature Review are described in Figure 2.

Figure 2 presents an overview of the process of the Systematic Literature Review which was followed by this research. It is worth noting that although these stages are depicted as sequential, the execution of many stages involves iteration to previous phases. Each phase and stage is now discussed in more detail and related to this study.

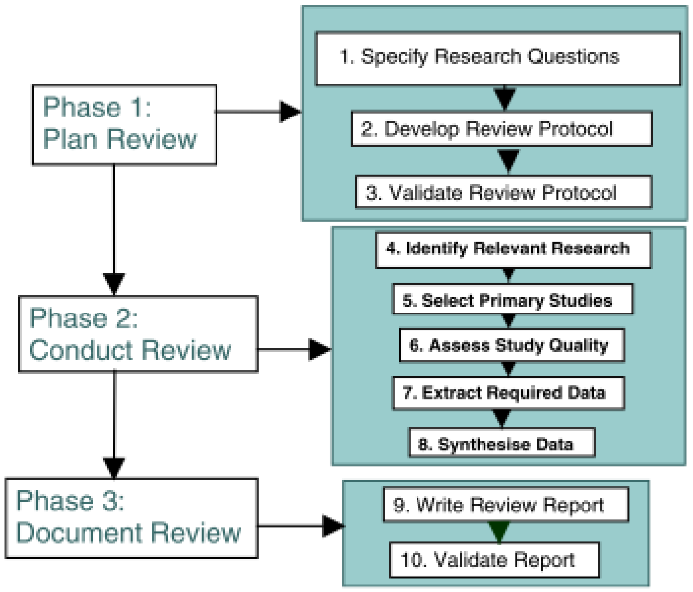


Figure 2: The process of the Systematic Literature Review (Brereton et. al, 2007, p. 572)

3.2.1 Plan Review Phase

This phase defines the research objectives and how the review will be conducted by formulating the research question, and producing a defined, documented and strict procedural review protocol. Each stage will now be described to provide a clear understanding of how this methodology guided the undertaking of the research.

A research question is the main question that a research aims to answer. Kitchenham (2004) recommends using three viewpoints to formulate the research question. The three view points are: population, intervention, and outcomes. Population means somebody or something that can be affected by the intervention. Interventions are software techniques that perform special tasks. Outcomes are factors of importance to practitioners. The three viewpoints are used for better analysing the research question when designing a research strategy. The research question for this research has already been introduced and justified in chapter 2 and it is analysed in more detail in section 3.3.

The "develop review protocol" stage involves designing a protocol on how to conduct a Systematic Literature Review for this research. This review protocol addresses factors such as the research questions, research strategy, search string(s), data sources, quality assessment criteria, data extraction plan, and data synthesis plan. Kitchenham (2004) and Brereton et al. (2007) both emphasise the benefits of designing a review protocol before conducting a Systematic Literature Review and particularly note that this helps to reduce the possibility of research bias. Without a review protocol, researchers' expectations might drive the selection of studies. Brereton et al. (2007), MacDonell and Shepperd (2007) and Staples and Niazi (2007) all developed their review protocols before conducting their Systematic Literature Reviews.

The "validate review protocol" stage involves evaluating and justifying the review protocol itself. Kitchenham (2004) and Brereton et al. (2007) recommend peer review and piloting the review protocol with a reduced set of identified sources. If the obtained results are not suitable, the review protocol can be revised and validated. Brereton et al. (2007), MacDonell and Shepperd (2007) and Staples and Niazi (2007) all reviewed and piloted their review protocols.

For details on the review protocol for this research, please refer to section 3.3.

3.2.2 Conduct Review Phase

After deriving an appropriate review protocol, the Systematic Literature Review can be conducted. During the "conduct review" phase, the search in the pre-identified search engines and journals is executed and the studies obtained are evaluated according to the study selection criteria which are defined in the review protocol. After identifying the studies, the quality of each identified study is assessed to ensure a minimum level of quality is used in this study. The data which are useful for answering the research question are then extracted from each identified study and finally, the extracted data are synthesised for analysis and answering of the research question.

The first stage in this phase, "identify relevant research", involves performing the search in pre-identified search engines and journals and retrieving the candidate relevant studies. This is the first step in identifying the relevant studies. During this stage the search string(s) defined in the review protocol are customised so that they suit each specific search engine used. This is because different search engines deal with logical operators differently and may have restrictions on the number of terms and their combinations. This involves some trial and error with different queries in each search engine.

The next stage “select primary studies” involves deciding which primary studies from all the retrieved possible studies are included in the review. It is good practice to have the study selection process reviewed by a second person to reduce the possibility of selection bias.

The third stage “assess study quality” entails evaluating the quality of each identified primary study to ensure the result of this review contains a minimum level of quality. Since the result of the review is based on the analysis of each primary study, if the primary study contains bias, the result of the review will be biased. The results of each study’s quality assessment should be reviewed to minimize the bias on the assessment.

The stage “extract data” concerns extracting data from identified primary studies. Once the relevant studies are selected and their quality assessed, the data which are useful for answering the research question can be extracted from each identified study. The data will be extracted according to the data extraction form which is defined in the review protocol. Kitchenham (2004) suggests it is good practice to have the data extraction process reviewed.

The final stage of conducting the review, “synthesise data” involves tabulating the extracted quantitative and qualitative data and identifying trends and themes using an appropriate technique. This is then used to answer the research question.

3.2.3 Document Review

The final phase of the Systematic Literature Review methodology “document review” contains two stages: writing the review report and validating the review report. This stage is intended to trigger planning on the purpose of the report and match the publication, format and dissemination of the results to the intended audience and purpose. In the case of this study, the review will be reported in this dissertation and will be reviewed by the examiners and supervisor.

The design of the review protocol is central to this methodology and is now described in detail for this study.

3.3 Review Protocol

The review protocol of this research follows a synthesis of the guidelines and recommendations provided by Brereton et al. (2007), Kitchenham (2004), MacDonell and Shepperd (2007) and Staples and Niazi (2007). This review protocol was developed iteratively through a process of design, test, review, modify. The initial design was peer reviewed by the author's supervisor and certain modifications were made after getting significant feedback. Then several pilot searches were conducted to identify potential relevant studies as well to test the appropriateness of the review protocol. After several iterations of such refinement, the final Systematic Literature Review protocol was specified.

This review protocol addresses factors that include: the research question(s), research strategy, search string(s), data sources, quality assessment criteria, data extraction plan, and data synthesis plan. What each part of the protocol means and how it was designed in this study are discussed in the following sections.

3.3.1 Research Question

A research question is the main question that a research aims to answer and has been formulated in Chapter 2. Following Kitchenham's (2004) suggestion, the research question has been analysed using a framework based on the three viewpoints: population, intervention, and outcomes. Population means somebody or something that can be affected by the intervention. Interventions are software techniques that perform special tasks. Outcomes are factors of importance to practitioners. This structure helps to guide the design of the research strategy and also helps to identify the data to be extracted.

As discussed in Chapter 2, the research question driving this research is:

What is the strength of evidence for the effectiveness of different requirements prioritization techniques for medium to large numbers of requirements in the software engineering domain?

Using the above framework of analysis on the research question gives:

Population: the newly released software systems that contain medium to large numbers of requirements that can be affected by the prioritization.

Intervention: the requirements prioritization techniques that have been used on medium to large numbers of requirements.

Outcomes: suitable prioritization techniques can be chosen to prioritize medium to large numbers of requirements.

This clarifies that the emphasis is on identifying and describing the intervention, the prioritization technique. The outcome is clearly providing developers with well evidenced techniques for prioritizing medium to large requirements sets for different software systems.

3.3.2 Research Strategy

The research strategy describes a set of constraints or criteria that allow the selection of primary studies that provide information directly answering the research question. The research strategy was designed by considering the viewpoints of the research questions. It was piloted to ensure it is reliably interpreted and identifies studies correctly.

This review is limited to meet the following criteria:

- The publications are primary studies.
- The publications are studied in the software engineering domain.
- The publications are published in English.
- The publications are published within the last 5 years (from 2004 to 2008).
- The prioritization technique either explicitly claims to be designed for a medium to large number of requirements or the study is evaluated with 15 or more requirements.

These criteria are now explained and justified.

The first two criteria refer to the fact that this research is looking for empirical evidence from primary studies on the efficacy of prioritization techniques in the area of software engineering.

Kitchenham (2004) notes that the study should not be excluded based on the language of the study. That is, studies written in different kinds of languages such as French, Germany or Japanese all need to be considered to minimize research bias. But due to the limitation of the linguistic knowledge of the author, studies only written in English are included in this review.

Since software development practices have changed rapidly over time, studies that were published more than 5 years ago may be out of date. In order to be sure the results of this research are relevant to today, this research reviews the studies that were published within the last 5 years (from 2004 to 2008).

It is difficult to design key words to represent the meaning "medium to large number" for a search string, since any number above 14 can be relevant. Therefore the author would search the studies manually to select the relevant studies.

Kitchenham (2004) recommends that the research strategy should be developed in consultation with librarians. The author developed the research strategy without the assistance of librarians. The author feels confident in developing a research strategy that would retrieve all relevant studies.

Parts of this research strategy are now transformed into a search string that is likely to return a subset of articles in this area that meets some of these constraints.

3.3.3 Search String

The aim of the search string is to minimize the research effort but identify as many of the relevant studies as possible. This research mainly focuses on the effectiveness of prioritization techniques that have been applied to medium to large numbers of requirements in the software engineering domain. Thus two kinds of key words "requirements" and "prioritization" in the search string are designed to locate studies that study requirements prioritization techniques. Articles would be selected if they contain the key words (or alternative spelling or different forms of the key words). Two search strings have been designed, one for the search engines, and one for the journal indexes. For the search string designed for search engines, the two kinds of key words were designed to be searched only in title and abstract fields. Brereton et al. (2007) and

Staples and Niazi (2007) follow this technique also, to retrieve possible relevant studies from search engines. For the search string designed for journals, the two kinds of key words were designed to search the entire article.

For the identified search engines (see the next section for more details), the search string is designed as the following:

In title or abstract field:
(requirements OR requirement) AND (prioritization OR prioritisation OR prioritize OR prioritise OR prioritizing OR prioritising OR prioritizes OR prioritises OR prioritized OR prioritised)

In any field:
AND (software)

AND (pyr >= 2004 AND pyr <= 2008)

For the identified journals (see the next section for a list), the search string is designed as following:

In any field:
(requirements OR requirement) AND (prioritization OR prioritisation OR prioritize OR prioritise OR prioritizing OR prioritising OR prioritizes OR prioritises OR prioritized OR prioritised)

AND (pyr >= 2004 AND pyr <= 2008)

Note: "pyr" means the year when the publication is published.

Two kinds of search strings are designed to locate possible relevant studies from identified search engines and identified journals. "*In title or abstract field*" means a search engine locates articles that contain the key words in the title or abstract of the article. "*In any field*" means a search engine locates articles that contain the key words anywhere in the article.

This research studies prioritization techniques under the software engineering environment. Studies that are studied under other environments such as medicine, architecture and hardware design are excluded. The key word “software” in the search string is used to locate studies which are studied under the software engineering domain.

Having identified the selection criteria and related search string, the next section describes the sources that will be searched to find the candidate articles that meet these criteria.

3.3.4 Data Sources

Data sources specify the places where the possible relevant studies can be retrieved. Two kinds of data sources were identified: search engines and journals. Since there is a possibility that relevant studies are contained in other search engines which are not identified in the review protocol, relevant journals need to be identified to ensure high quality studies are included.

The identified search engines include the main search engines that cover the domain of computer science and information systems.

- ACM Digital library
- Google scholar
- IEEE Xplore
- ProQuest
- ScienceDirect
- Springer

The identified journals were obtained by surveying the articles from the literature review (Chapter 2) and their cited references.

- Empirical Software Engineering
- IEEE Transactions on Software Engineering
- IEEE Software
- Information and Software Technology
- Journal of System and Software

- Requirements Engineering
- Software Process Improvement and Practice
- Software Quality Journal

The relevant studies are then selected based on applying the selection criteria to the candidate articles returned by running the search string queries. The design of this process is described in the following section.

3.3.5 Selection Process

The search process specifies how to identify relevant studies from those retrieved by the queries on the search engines and individual journals specified in the previous section.

The title and abstract of each retrieved paper is read by the author to determine whether that paper is relevant or not, with particular emphasis on checking that the prioritization technique is evaluated with 15 or more requirements. If the relevance of the paper cannot be determined by its title and abstract, the full text of the paper is read to determine its relevancy.

It is expected that the searches may return the same article in different search engines, and duplicates will be removed.

Similarly articles with ambiguous or unclear relevance, for example where the evaluation of the technique is not discussed in sufficient detail, will be discarded.

The selected papers are then assessed for quality and the appropriate data extracted from them to answer the research question. These processes are described in the next two sections.

3.3.6 Quality Assessment Criteria

Quality assessment criteria specify how to assess the quality of the study. Since the result of the review is based on the analysis of each primary study, if the primary study contains bias, the result of the review will be biased. The results of each study’s quality assessment should be reviewed to minimize the bias on the assessment. This quality assessment is designed through the use of a quality assessment form (refer to Appendix B, Table 15). This is designed for assessing the quality of each identified study. The style and quality assessment questions were inspired by the guideline provided by Kitchenham (2004) and Lötter (2000). Kitchenham (2004) recommends a checklist of factors and Lötter (2000) provides a broad framework of ideas on how to judge the scientific value of a research report. The internal and external validity of the studies are the main measures of quality for each study, as suggested by Kitchenham (2004). Internal validity means that the design and implementation of a study are likely to prevent systematic error. External validity means that the outcome of the study can be generalized to a population. Internal validity is gauged for each study by assessing the clarity of the research problem, the clarity and justification for the research methodology, links to related literature, and the strength of the evidence and its support for the results of the study. External validity is assessed by firstly confirming internal validity, which is a necessary but not sufficient condition for external validity. The context of the study, the research methodology and the characteristics of the participants all influence the external validity score. A six point scale is used on the Quality Assessment Form (Appendix B, Table 15) to capture most of the quality evaluations. If the internal validity of a selected paper is below the threshold of scoring a "3", then it should be carefully considered whether to include it or not. The external validity scores are used to indicate the extent to which the prioritization technique can be generalized to other situations.

3.3.7 Data Extraction

Data extraction plans what kind of data will be extracted from each relevant study to address the research questions. A data extraction form (refer to Appendix B, Table 16) is designed to capture the data that needs to be extracted from each study.

Firstly, some publication details are captured along with the date of the data extraction. Next, some general information is captured that includes details of the context of the study, for better understanding of a particular prioritization technique. Some details of the applicability of the prioritization technique are then extracted ("Evaluation" section of the form). This is needed because the effectiveness of a prioritization technique might not only depend on the type of technique, but may also depend on other factors such as the number of requirements used, the number of stakeholders who prioritize requirements, when the prioritization is performed or the occupation of each participant. The actual effectiveness of a technique is also recorded, together with how it is measured. The strength of evidence of effectiveness is used to directly answer the research question. It is judged based on the score of internal validity of each identified study, whether the evaluation is thorough, and the subjective/objective metric. An answer style: “strong”, “medium”, and “weak” is used to capture the degree of the strength of evidence for each identified study.

The data extracted and recorded on the data extraction forms are then analysed and synthesised to provide a holistic view of the studies and to identify any patterns or themes. This data synthesis plan is described in the next section.

3.3.8 Data Synthesis

Data synthesis specifies the plan on how to collate and summarize the data of the primary studies. Since the different studies might involve different populations, interventions, research methods, research contexts, study qualities, and other variables, the results of the primary studies could be heterogeneous. Therefore data needs to be tabulated to display the impact of heterogeneity. A descriptive synthesis is needed to interpret the impact of heterogeneity.

3.4 Chapter Conclusion

This chapter describes and justifies the selection of a Systematic Literature Review as a research methodology that is well suited to the research question. The different stages of the methodology are described in detail with explanations and examples from similar studies. The design of the Review Protocol is described in detail with rationale for the design decisions.

The next chapter describes the next stage of the Systematic Literature Review process, the implementation of this Review Protocol and how it was conducted. It includes reporting of the data extraction and synthesis stages and a discussion of the results.

4 Conducting the Review, Results and Discussion

4.1 Research Identification and Study Selection

The research identification process was systematically executed according to the review protocol. Search strings defined in the review protocol were applied to the identified data sources (search engines and journals) to retrieve potentially relevant papers. Appendix C displays the customised search string details for each pre-selected search engine. The customisation and running of the search strings in each pre-selected search engine and journal was completed over a two week time period (see Appendix F for a detailed timetable). This returned 273 potentially relevant papers from the pre-selected search engines and 317 potentially relevant papers were from the journals (see Table 8 and Table 9).

The 590 candidate papers were reduced by firstly eliminating duplicate papers, then by further evaluation according to the selection criteria outlined in the research strategy described in the Review Protocol. For all the retrieved papers, the title and abstract of each paper were carefully read by the author to determine if it is a relevant paper or not. If the relevance of a paper could not be determined by its title and abstract, the full text of the paper was read to assess its relevancy. In this way many candidate studies were rejected because they were not primary studies, were not in the software engineering domain or were related to inappropriate topics. Many more candidate studies were discarded because they did not satisfy the criterion that the prioritization technique studied in the paper was not explicitly evaluated with 15 or more requirements. Note that even if a study uses a prioritization technique that could be a suitable method for a medium to large number of requirements, if that study does not explicitly specify that the technique is designed for large numbers of requirements, or it is clearly evaluated with 15 or more requirements, then that study was excluded since there is no clear evidence that the technique is suitable (or not).

Eighteen papers were originally selected from all the retrieved papers. Nine papers were further discarded due to either duplicate contents contained in different papers or little information provided on the effectiveness of requirements prioritization techniques. Nine relevant primary studies (3 conference papers, 4 journal papers, and 2 workshop papers) were finally identified for the review. Tables 8 and 9 show the results of the search and selection process for the identified search engines and journals, respectively. The journal search and evaluation did not contribute any papers that had not been discovered through search engines. The 9 papers selected for inclusion in the final study are shown in Table 10.

Table 8: Search results for identified search engines

Search engines	**Found**	**Total Discarded**	**Finally Included**
ACM Digital library	22	21	1: [4]
Google scholar	50	45	5: [1] [2] [5] [7] [8]
IEEE Xplore	105	101	4: [1] [2] [6] [8]
ProQuest	16	16	0
ScienceDirect	35	33	2: [3] [9]
Springer	45	44	1: [5]
Total	**273**	**260**	**9 distinct papers**

Table 9: Search results for identified journals

Journals	**Found**	**Total Discarded**	**Finally Included**
Empirical Software Engineering	18	17	1: [5]
IEEE Transactions on Software Engineering	44	44	0
IEEE Software	74	74	0
Information and Software Technology	57	57	0
Journal of System and Software	66	64	2: [3] [9]
Requirements Engineering	24	24	0
Software Process Improvement and Practice	8	7	1: [7]
Software Quality Journal	26	26	0
Total	**317**	**313**	**4**

Table 10: The selected papers for the review

The 9 identified papers for the review:

[1] Avesani, P., Bazzanella, C., Perini, A., & Susi, A. (2005). *Facing Scalability Issues in Requirements Prioritization with Machine Learning Techniques*. Proceedings of 13th IEEE International Conference on Requirements Engineering, 297-305.

[2] Beg, R., Abbas, Q., & Verma, R. P. (2008). *An Approach for Requirement Prioritization Using B-Tree*. First International Conference on Emerging Trends in Engineering and Technology (ICETET'08), 1216-1221.

[3] Berander, P., & Svahnberg, M. (2008). Evaluating Two Ways of Calculating Priorities in Requirements Hierarchies - An Experiment on Hierarchical Cumulative Voting. *Journal of Systems and Software, In Press, Corrected Proof.*

[4] Cleland-Huang, J., & Mobasher, B. (2008). *Using Data Mining and Recommender Systems to Scale Up the Requirements Process*. Proceedings of the 2nd international workshop on Ultra-large-scale software-intensive systems.

[5] Karlsson, L., Thelin, T., Regnell, B., Berander, P., & Wohlin, C. (2007). Pair-wise Comparisons Versus Planning Game Partitioning-experiments on Requirements Prioritisation Techniques. *Empirical Software Engineering, 12*(1), 3-33.

[6] Laurent, P., Cleland-Huang, J., & Duan, C. (2007). *Towards Automated Requirements Triage*. 15th IEEE International Requirements Engineering Conference, 131-140.

[7] Lehtola, L., & Kauppinen, M. (2006). Suitability of Requirements Prioritization Methods for Market-driven Software Product Development. *Software Process Improvement and Practice, 11*(1), 7-19.

[8] Perini, A., Susi, A., Ricca, F., & Bazzanella, C. (2007). *An Empirical Study to Compare the Accuracy of AHP and CBRanking Techniques for Requirements Prioritization*. Fifth International Workshop on Comparative Evaluation in Requirements Engineering, 23-35.

[9] Pettersson, F., Ivarsson, M., Gorschek, T., & Öhman, P. (2008). A Practitioner's Guide to Light Weight Software Process Assessment and Improvement Planning. *Journal of Systems and Software, 81*(6), 972-995.

4.2 Quality Assessment of Selected Papers

The quality assessment form specified in the review protocol (see Table 15 in Appendix B) was applied to each of the 9 selected primary studies to assess their quality. A scale of 1 to 5 was used, with 1 indicating poor quality and 5 indicating a high level of quality. Where the study has insufficient information to assess its quality, it is assigned a code “N”. The individual assessment details are presented in tabular form in Appendix D.

The overall results of the quality assessments for the 9 papers, as measured by internal and external validity, are summarised in Table 11.

Table 11: Results of study quality assessment

Questions / Paper No.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
Does the study contain high internal validity?	N	N	5	N	5	5	4	5	4
Does the study contain high external validity?	N	N	3	N	1	3	3	2	3

The internal validity and external validity of each paper was judged on the answer of the quality assessment questions (see Appendix D) and also more holistically. For papers [5], [7] and [9], only part of the paper is considered to be relevant for this review. In this case the quality of the paper was judged by that part, not the whole paper.

In the 9 identified papers, 4 papers ([3] [5] [7] [9]) are journal papers, 3 papers ([1] [2] [6]) are conference papers, and 2 papers ([4] [8]) are workshop papers. The 4 journal papers were considered to have some base level of quality, since they have already been thoroughly reviewed by the editorial panels. From the results it is seen that papers [6] and [8] also are of a fairly good standard of quality. Although these 6 papers do not possess a high degree of external validity, they show a high level of internal validity. In reality, every experiment contains threats to validity to some degree. Good internal validity does not mean that there are no threats, but that the threats are small enough to be controlled well, and have an insignificant influence on the results. The 6 papers ([3] [5] [6] [7] [8] [9]) are used for this review.

No quality evaluation score is provided for papers [4], [2] and [1] since there is insufficient information in the paper from which to judge the level of their quality, therefore the quality of these 3 papers is uncertain. Despite this uncertainty in their quality, the information in these papers is still useful and has been used in this study, but with extra caution.

Three out of 9 studies are set in an industrial setting. It could be argued that the results of these 3 studies are more easily transferrable to another industrial context, compared to the studies set in an academic context. This could be interpreted as an expected higher external validity score for the 3 studies in an industrial setting.

4.3 Data Extraction and Presentation

Data were extracted according to the data extraction form specified in the review protocol (see Appendix E for the completed individual data extraction forms). For papers [5], [7] and [9], only part of each paper is considered to be relevant for this review, therefore the data were extracted and analyzed based on that part, not the whole paper. An initial data analysis identified two categories of data. One category applies to a "large" requirements set, with 100 or more requirements. These prioritization techniques are explicitly designed for this large number of requirements. Four studies ([1] [2] [4] and [6]) belong to this category, although one of them has no empirical evaluation of the prioritization technique. The other category applies to a "medium" number of requirements, where the prioritization techniques are evaluated with 15 to 30 requirements. Five papers are in this "medium" category, papers [3] [5] [7] [8] and [9].

The results of the data extraction for the large and medium requirements sets are tabulated in Tables 12 and 13, respectively. Note that where the number of requirements that a study claims to be applicable to is not specified as a number, the description of the size of the requirements set is quoted from that study. This approach was also used to extract data for the number of stakeholders that the study claimed to be applicable to.

4.3.1 Large Requirements Set

Firstly the idea of each new technique, the evaluation and the effectiveness of that technique are discussed for the four papers investigating techniques for "large" requirements sets. The techniques in question are Case-based ranking, B-Tree method, Pirogov, and an un-named technique in study [4].

Table 12: Data Results for Large Requirements Sets

Category: Large number of requirements	**[1]**	**[2]**	**[4]**	**[6]**
General information				
The name of the newly designed technique	Case-based ranking	B-Tree method	Not named	Pirogov
Name of the prioritization technique	Ranking	Balanced search tree	Numerical assignment	Any prioritization technique
What technique is introduced in order to manage requirements?	Machine learning technique	No other technique	Data-mining technique	Clustering technique
Type of requirements claimed to be suited to	Not specified	Not specified	"unstructured or semi-structured data"	Not specified
Number of requirements claimed to be suited to	"large"	"large"	"massive"	"large"
Number of stakeholders claimed to be suited to	"single and multiple stakeholders"	Not specified	"broad stakeholders"	"a large set of stakeholders"
Evaluation				
Is the technique evaluated in an industrial or academic setting?	Academia	Academia	N/A	Industry
What is the methodology of the evaluation?	Simulation	Not specified	N/A	Case study
What are the roles of the participants in the evaluation?	Not specified	Not specified	N/A	Not specified
Number of requirements used for prioritizing requirements	25, 50, and 100 requirements	121 requirements	N/A	202 requirements
Types of requirements used for prioritizing requirements	Not specified	Not specified	N/A	Functional and non-functional
Number of participants used for prioritizing requirements	Not specified	Not specified	N/A	Not specified
Type of the market the evaluation resides in	Not specified	Not specified	N/A	Bespoke
The purpose of the prioritization	Not specified	Not specified	N/A	Find important requirements
When does the prioritizing process perform?	Not specified	Not specified	N/A	Not specified
Outcome measurements and results				
What is the measurement of the effectiveness?	User effort and accuracy	Number of comparisons	N/A	User effort and accuracy
Effectiveness of the technique	Outperforms AHP with both user effort and accuracy.	The number of comparisons is dramatically reduced.	N/A	User's effort is reduced. But some requirements are incorrectly prioritized.
Strength of evidence of effectiveness	Weak	Weak	Weak	Medium

Study [1] proposes a semi-automatic prioritization technique named Case-based ranking (CBRanking). The idea of CBRanking is that it exploits machine learning techniques to reduce users' elicitation effort in the prioritization process. The system first selects a reduced number of pairs of requirements to let users make decisions on which requirements they prefer. At the same time it performs a requirements analysis, and then it uses machine learning techniques to generate an approximation of a final ranking.

Simulation of the prioritization process is used to evaluate the effectiveness of CBRanking. The measurements of effectiveness are user effort and accuracy of result. User effort is measured by the number of requirements elicited pairs as a percentage of the total number of requirements pairs. The degree of user agreement is used to measure the accuracy. The effectiveness of CBRanking was benchmarked against AHP in terms of effort and accuracy. Both CBRanking and AHP are running the simulation process. Since this paper does not provide much information on how the simulation reflects the real situation and how the simulation is performed on AHP, the degree of reliability of data is unknown.

Figure 3 shows one evaluation result of study [1]**.** Consider 5% of elicited pairs, where we see approximately 28% disagreement with AHP and 14% disagreement with CBRanking. When considering a low ratio of user effort, it is seen that CBRanking outperforms AHP with accuracy. This is significant because user effort is reduced with an acceptable accuracy rate. Since the internal validity of this study is unknown, the strength of evidence of effectiveness is weak.

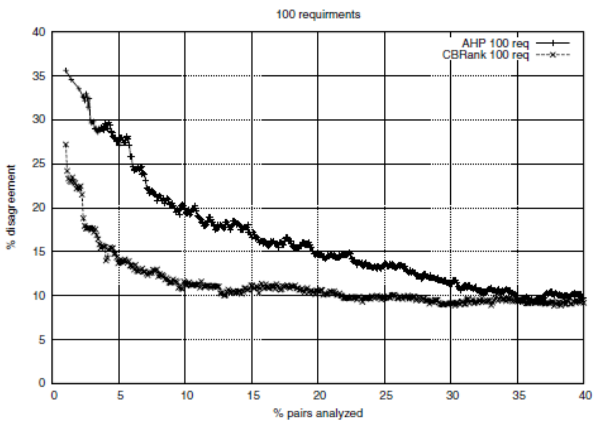


Figure 3: The plot of disagreement (y-axis) for a set of 100 requirements for an increasing number of elicited pairs (x-axis) (Avesani et al., 2005, p. 232)

Study [2] utilises the B-Tree method for prioritizing requirements. B-Tree (balanced search tree) is a well established method for sorting elements. The idea of B-Tree is that its internal nodes can contain a variable number of children nodes (see for example Cormen, Leiserson, Rivest and Stein (2001)). Since this paper does not provide much information on how this method is evaluated, the details on the evaluation are unknown. The measurement of effectiveness is the number of comparisons. Figure 4 shows the evaluation result of study [2]. The complexity of B-Tree is O ($t*\log_t n$) (t is some integer). With 121 requirements, the number of comparisons for B-Tree is 14. Compare this with AHP; since the complexity for AHP is O (n^2), the number of comparisons for AHP is 7260. It is seen that the number of comparisons is dramatically reduced for B-

Tree compared with AHP. This is significant when there are a large number of requirements that need to be prioritized, but only a small number of comparisons, and therefore user effort, is needed. But since this study does not evaluate the accuracy of B-Tree, the accuracy of B-Tree is unknown. Since the internal validity of this study is unknown, the strength of evidence of effectiveness is weak.

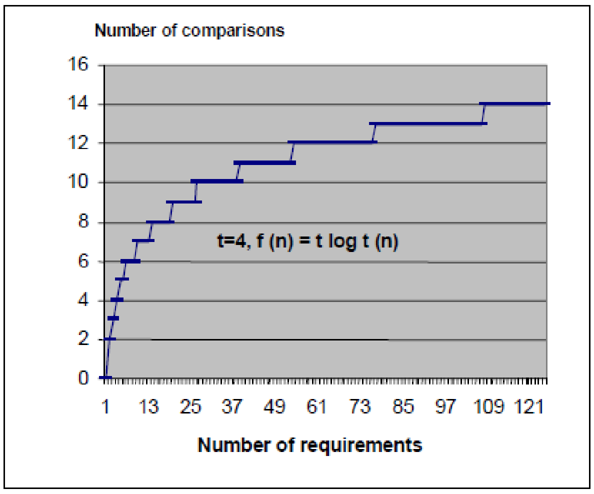


Figure 4: Number of comparisons of B-Tree (Beg et al, 2008, p. 1220)

Study [4] introduces a new technique (not named) to facilitate the requirements prioritization process. This technique adopts data-mining techniques to analyze and process large numbers of requirements and uses recommender systems to manage broad stakeholders. The idea of the technique is that it dynamically assigns stakeholders to appropriate forums to let them work collaboratively to prioritize requirements. Since this paper is a position paper, no evaluation is provided to assess the effectiveness of this technique. Since there is no evaluation contained in this study, the strength of evidence of effectiveness is weak.

Study [6] proposes a semi-automatic technique named Pirogov to facilitate the requirements prioritization process. The idea of Pirogov is that it adopts clustering techniques to automatically cluster requirements based on different goals. Stakeholders then manually prioritize the clusters by using any prioritization technique. The system then automatically generates a list of ranked requirements. The approach that Pirogov takes is similar to CBRanking in that both reduce the amount of user effort, and then generate an approximation of ranked requirements. A case study is used to evaluate the effectiveness of Pirogov. The measurements of effectiveness are user effort and accuracy. User effort is measured by the number of requirements that need to be prioritized by stakeholders, and accuracy is measured by comparing the results of their technique with a previously ranked “accurate” prioritized list of the same requirements from a completed project.

The evaluation uses 202 requirements. Forty one clusters are generated by the system and are manually prioritized by stakeholders. Then the system automatically generates a list of ranked requirements. It is seen that the user effort is reduced, since the users only need to prioritize 41 high-level abstractions instead of 202 requirements. Two of the researchers examine the accuracy of the results. Results show that 17% of requirements are incorrectly assigned to high priority value and 1 important requirement is assigned to low priority value.

This is significant because user effort is reduced with an acceptable accuracy rate. The internal validity of this study is high, the evaluation is thorough, but as it is a subjective measurement, the strength of evidence of effectiveness is medium.

4.3.2 Medium Requirements Set

Five studies ([3] [5] [7] [8] and [9]) use medium-sized requirements sets (15 to 30 requirements) to evaluate prioritization techniques. The prioritization techniques that are evaluated are: AHP, cumulative voting, hierarchical cumulative voting, tool-supported AHP, pair-wise comparisons, planning game, CBRanking, and tool-supported pair-wise comparisons. Chapter 2 has already introduced the AHP, cumulative voting, pair-wise comparisons, and planning game methods. CBRanking has also been introduced in study [1]. The idea of other prioritization techniques, the evaluations of all these prioritization techniques, and the effectiveness of these techniques are discussed as following.

Table 13: Data Results for Medium-Sized Requirements Sets

Category: Medium number of requirements	**[3]**	**[5]**	**[7]**	**[8]**	**[9]**
General information					
The name of the newly designed technique	Not named	No new technique	No new technique	No new technique	No new technique
Name of the prioritization technique	Hierarchical cumulative voting	PWC, Planning Game, TPWC	Pair-wise comparison	CBRanking and AHP	Cumulative voting
What technique is introduced in order to manage requirements?	Hierarchical technique	Computer tool (TPWC)	No other technique	Two web based tools (SCORE and JAHP)	No other technique
Type of requirements claimed to be suited to	Hierarchical requirements	Not specified	Not specified	Not specified	Not specified
Number of requirements claimed to be suited to	Not Specified	Not specified	Not specified	Not specified	Not specified
Number of stakeholders claimed to be suited to	Not Specified	Not specified	Not specified	Not specified	Not specified
Evaluation					
Is the technique evaluated in an industrial or academic setting?	Academia	Academia	Industry	Academia	Industry
What is the methodology of the evaluation?	Experiment	Experiment	Case study	Experiment	Case study
What are the roles of the participants in the evaluation?	Master student	PhD Students and professor	Users of system	PhD students and Junior Researchers	All development roles
Number of requirements used for prioritizing requirements	27 requirements	16 requirements	"ten categories of 20 or less requirements"	20 requirements	15 requirements
Types of requirements used for prioritizing requirements	High-level and lower-level, functional	High-level and independent	User requirements	User, high-level, and independent	Not specified
Number of participants used for prioritizing requirements	18	1st: 8, 2nd: 30	4	18	27
Type of market the evaluation resides in	Not specified	Market-driven	Market-driven	Not specified	Bespoke
The purpose of the prioritization	Find important requirements	Find high value and low cost requirements	Find important requirements	Find important requirements	Find important requirements
When is the prioritizing process performed?	Not specified	Not specified	Not specified	Not specified	The early stage
Outcome measurements and results					
What is the measurement of effectiveness?	User effort, scalability, and accuracy	User effort and accuracy	User effort	Accuracy	Accuracy
Effectiveness of the technique	HCV with compensation factor provides more accurate results than without compensation HCV is reasonably easy to use, scalable, and can provide reliable results.	PWC is time consuming and hardest to use. TPWC is the fastest and as easy to use as planning game. Similar accuracy among the three techniques.	Difficult to estimate how much one requirement contains more/less value than another. Difficult to prioritize more than 20 requirements. It is difficult to prioritize requirements at different levels of abstraction.	SCORE contains less accuracy than JAHP.	Participants disagree between each other. Satisfaction with the overall order varies between individual participants.
Strength of evidence of effectiveness	Medium	Medium	Medium	Medium	Weak

Note: Pair-wise comparisons (PWC), Tool-supported pair-wise comparisons (TPWC)

Study [3] introduces a compensation factor when using hierarchical cumulative voting (HCV) to prioritize unbalanced hierarchical requirements to make the prioritization results more accurate. Unbalanced hierarchical requirements mean that high-level requirements have different numbers of low-level requirements. In HCV the cumulative voting technique is used to prioritize requirements which are structured hierarchically.

An experiment is used to evaluate the effectiveness of HCV with and without a compensation factor. The measurements of effectiveness are user effort, scalability, and accuracy. Ease of use is used to measure user effort. Users' perception is used to measure ease of use, scalability, and accuracy. The 18 participants are randomly assigned to two groups. One group uses HCV with a compensation factor to prioritize requirements. The other group uses HCV without a compensation factor to prioritize requirements. Twenty seven requirements (6 high-level requirements and 21 low-level requirements) are used for prioritization. The evaluation results show that 83% of participants think HCV with a compensation factor produces accurate results, and 36% think HCV without a compensation factor produces accurate results. Besides that, most participants think HCV is reasonably easy to use, and is scalable if the number of low-level requirements does not grow considerably.

Study [5] evaluates three prioritization techniques: pair-wise comparisons (PWC), planning game and tool-supported pair-wise comparisons (TPWC). TPWC is where the PWC technique is built into a management tool to help users to prioritize requirements. Two experiments are conducted to evaluate the effectiveness of the three prioritization methods. The measurements of effectiveness are user effort and accuracy. Time-consumption and users' perception of ease of use are used to measure user effort. Users' perception is also used to measure the accuracy of each method.

The first experiment compares PWC with planning game. Eight participants are selected to prioritize 16 requirements. The results of the first experiment show that PWC takes 55% more time to complete than planning game. More participants (62.5%) think planning game is easier than PWC. There is no significant difference between the number of participants who think planning game is more accurate and the number of participants who think PWC is more accurate.

The second experiment compares planning game with TPWC. Thirty participants are selected to prioritize 16 requirements. The results show that TPWC takes 17% less time to complete than planning game. More participants (53%) think TPWC is easier than planning game. There is no significant difference between the number of participants who think planning game is more accurate and the number of participants who think TPWC is more accurate.

In summary, the results of the two experiments show that PWC is time consuming and the least easy to use compared with the other two. TPWC is the fastest technique and it is as easy to use as planning game. There is no significant difference regarding accuracy among these three techniques.

Study [7] uses an industrial case study to evaluate the effectiveness of pair-wise comparisons. The measurement of effectiveness is user effort. And the measurement of user effort is users' perception. Four users use "ten categories of 20 or less requirements" to prioritize requirements. After the prioritization process, the researcher interviews the practitioners about the usage of the method and asks them how they felt about the prioritization results. The findings are that participants feel it is difficult to estimate how much one requirement contains more/less value than another. It is difficult to prioritize more than 20 requirements. Some users feel that it would be easier for them just to select the requirements which are most important for them. This method also causes difficulty in prioritizing requirements at different levels of abstraction.

Study [8] compares two prioritization techniques: tool-supported AHP and tool-supported CBRanking. This study uses an experiment to evaluate the effectiveness of the two techniques. The measurement of effectiveness is accuracy. The measurement of accuracy is users' perception. Eighteen participants prioritize 20 requirements. Each participant uses both prioritization techniques to prioritize requirements. The order of using prioritization methods was randomly assigned to each participant to minimize the order effect.

Two post-tests are conducted to measure accuracy. The first post-test is that participants provide their opinions on which prioritization method is more accurate. The result shows that more participants (72%) think tool-supported AHP is more accurate than tool-supported CBRanking. The second post-test is that participants receive two lists of prioritization results without knowing which prioritization technique produced which list of results. Then the participants are asked to provide their opinions on which list of results is a better fit for their preferences. The result shows that 100% of the participants think tool-supported AHP is more accurate than tool-supported CBRanking.

The results show that tool-supported AHP produces more accurate results than tool-supported CBRanking. But the results of study [1] show that CBRanking outperforms AHP with accuracy when considering a low ratio of user effort. It seems that somewhat contradictory results are derived between study [1] and study [8]. Further investigation is needed to determine the reason for this.

Study [9] uses a case study to evaluate the effectiveness of the cumulative voting prioritization technique. Twenty seven participants use cumulative voting to prioritize 15 requirements. After finishing the prioritization process, disagreement charts and satisfaction charts are used for analyzing the level of agreement between participants. Since this study does not provide much detail on how to analyze the agreement between participants, the details of the analysis are unknown. The results show that there are disagreements between participants, but this disagreement does not influence the planning of the initial effort. Individual participants vary in their satisfaction with the overall order of departments.

The internal validities of studies [3], [5], [7] and [8] are high, the evaluations are thorough, but all of them are subjective measurements, thus the strength of evidence of effectiveness of the four studies is medium. Since study [9] does not provide enough detail on the analysis of the results, and it uses subjective measurements, the strength of evidence of effectiveness is weak for this study.

Overall the data show that there are a few promising candidate techniques for prioritizing medium to large requirements sets, although the evaluations and strength of evidence are quite variable.

The next section synthesizes the data to look for patterns and trends in the data and discusses the implications of this for prioritizing large requirements sets in practice.

4.4 Data Synthesis and Discussion

At this stage the research questions can be answered. Let's first review the research question:

> *What is the strength of evidence for the effectiveness of different requirements prioritization techniques for medium to large numbers of requirements in the software engineering domain?*

The remainder of this section is structured by synthesising and discussing the techniques used, the contexts and applicability of the techniques, metrics used for effectiveness, and the strength of the evidence for that effectiveness. The section concludes with an overall statement about the most promising techniques.

4.4.1 Approaches Used

From the data synthesis a number of general approaches for increasing the effectiveness of prioritization techniques for medium to large numbers of requirements can be identified. These approaches are summarized in Table 14. The categories are related to the specific techniques in the study in the following discussion.

Table 14: Some ways for increasing the effectiveness of prioritization techniques for medium to large numbers of requirements

Increase effectiveness

- Reduce user effort
    - Reduce the number of prioritization comparisons
        - Searching or ranking algorithms
        - Clustering technique
        - Machine learning technique
    - Reduce the number of requirements to be prioritized
        - Clustering technique
        - Data mining
        - Machine learning technique
        - Hierarchical technique
- Increase accuracy
    - Increase the knowledge of requirements
        - Data mining

One way to increase the effectiveness of prioritization techniques for medium to large numbers of requirements can be reducing the user effort. These approaches are categorised into the two main approaches of machine support to reduce either the number of comparisons or the number of requirements involved in the prioritization. Current techniques on reducing the number of comparisons are searching or ranking algorithms (e.g. B-Tree method), clustering techniques (e.g. Pirogov), and machine learning techniques (e.g. CBRanking). Current techniques for reducing the number of requirements needing to be prioritized are clustering techniques (e.g. Pirogov), data mining techniques (e.g. unnamed technique in study [4]), machine learning techniques (e.g. CBRanking), and hierarchical techniques (e.g. HCV). Note that clustering techniques and machine learning techniques can both reduce the number of comparisons and the number of requirements. This is because the two techniques let the users prioritize a reduced number of requirements, then the system automatically generates a ranked list of requirements. Both the number of comparisons and the number of requirements can be reduced.

Another way to increase the effectiveness of prioritization techniques for medium to large numbers of requirements can be by increasing accuracy. One way to increase accuracy is to increase knowledge of the requirements. Data mining is one technique to increase knowledge of requirements (e.g. unnamed technique in study [4]). Other ways to increase the accuracy of the prioritization techniques can be reducing the disagreements between stakeholders. Techniques for requirements negotiation can reduce the disagreements between different stakeholders. These techniques are out of scope of this dissertation.

4.4.2 Applicability

In the “large requirements set” category, all the 4 studies (100%) use machine support to help people prioritize requirements. It seems machine support is helpful to help people prioritize large numbers of requirements.

In the “large requirements set” category, all the 4 studies claim the techniques can be suitable for more than one stakeholder. In the “medium sized requirements” category, all the 5 studies use more than one participant to prioritize requirements. It seems these techniques can also be used for more than one stakeholder.

Most identified studies use “find important requirements” as the purpose of prioritizing requirements. The notion of “importance” isn’t specified in these studies and could mean any aspect such as cost, time, penalty, or risk.

Only a few studies are set in an industrial setting, with most of them using data generated from investigations in an academic context or using fabricated data. Evaluation of the techniques in an industrial setting provides practitioners with the most convincing evidence for the effectiveness of that technique. More empirical work needs to be done in the industrial setting to increase the body of convincing evidence for the effectiveness of different techniques.

A number of data collection types are not useful because the number of data sets is too small to draw conclusions or distinctions.

4.4.3 Measures of Effectiveness

Most studies use user effort and accuracy to measure the effectiveness of prioritization techniques. Scalability is also used to measure effectiveness.

User effort is measured in a number of different ways in the various studies. This includes: the number of decisions to be made, the number of requirements that need to be prioritized by the user, the time taken to prioritize the requirements, and the perceived ease of use. These measures are all related to some extent. For example, the number of requirements affects the number of decisions, which affects the time taken. It may be useful to have a more “standardised” metric for effectiveness so that comparisons of the effectiveness of techniques across different studies can more easily be made.

Users’ perceptions, user agreement and user satisfaction are the metrics used for the accuracy of the technique. Some papers confirmed accuracy by taking the approach of comparing the results of their technique with a previously ranked “accurate” prioritized list of the same requirements from a completed project. These metrics are quite subjective and open to interpretation. This may make it difficult for cross-study comparisons of accuracy improvement.

The scalability of a technique was judged by how many requirements could be prioritized with a “reasonable” effort. The prevalent notion is that machine support of prioritization will increase its scalability because some of the effort can be transferred to the machine.

Improvements in effectiveness of a technique were commonly estimated by comparing or “benchmarking” the results of the technique with the more traditional AHP prioritization process applied to the same number of requirements. This could be a useful standard approach allowing cross-comparison of studies and their effectiveness.

4.4.4 Level of Claimed improvements

There is no consistent measure for reporting levels of improvement in the studies since the metrics used are not the same. It is therefore difficult to compare studies. Most studies either report a quantitative improvement in effort, compared to a traditional method like AHP, or report a subjective improvement based on perception.

For the studies in the large number of requirements category, one study does not have any evaluation. CBRanking reports 5% of decision effort can achieve 85% accuracy. However, the reliability of this result is unknown. B-tree reduces the number of comparisons for N requirements to TLogN, less than N^2 for large N, but the accuracy is not mentioned. The evaluation shows Pirogov reduces the number of requirements by a factor of 4 (from 200 to 40) with 90% accuracy. However, this evaluation is a subjective measurement. It seems that most studies in the large number of requirements category are at the preliminary stage of evaluation. More studies on prioritization techniques for large numbers of requirements are needed.

All the studies in the medium-size category used a subjective measure of improvement based on the users’ perceptions of levels of improvement. It seems that the evaluations are still not strong for these studies.

4.4.5 Strength of Evidence of Effectiveness

For prioritization techniques for large numbers of requirements, most of the strength of evidence for effectiveness is weak. Stronger evidence presented for prioritization techniques for medium sized numbers of requirements shows the techniques are more mature than newly designed techniques.

4.5 Chapter Conclusion

This chapter describes each step on how the Systematic Literature Review was conducted in this study. The data synthesis shows that most of the strength of evidence of effectiveness is weak for prioritization techniques for large numbers of requirements. It also shows some inconsistency in how effectiveness is measured, and in reporting levels of improvement.

The next chapter provides the conclusion, the threats to validity, and proposes possible areas of future study that may be fruitful.

5. Conclusion

The objective of this research is to investigate the strength of evidence for the effectiveness of different requirements prioritization techniques for medium to large numbers of requirements. The methodology used for this research is a Systematic Literature Review. A Systematic Literature Review investigates research questions through identifying, evaluating and interpreting all relevant studies. It summarises the existing evidence for a certain technology. The reason for using a Systematic Literature Review to conduct this research is that it matches the purpose of this research, which is to systematically assess current studies in requirements prioritization techniques as reported in the literature, and analyse and draw together the results.

Before conducting the Systematic Literature Review, the review protocol is developed and reviewed to reduce the possibility of research bias. The review protocol defines the research objectives and how the review will be conducted. The research identification process, study selection process, study quality assessment, data extraction process, and data synthesis process are performed according to the review protocol. During these processes, the relevant primary studies are identified, the quality of each identified primary study is assessed, the data are extracted from the primary studies, and the extracted data are synthesised.

After conducting the Systematic Literature Review, prioritization techniques that have been applied to medium to large numbers of requirements are identified and the strength of evidence for effectiveness of each technique is evaluated. Some methods for increasing the effectiveness of prioritization techniques for medium to large numbers of requirements have been discovered and synthesised. The research question can be answered. It is found that for most prioritization techniques for large numbers of requirements, the strength of evidence for effectiveness is weak. More studies on prioritization techniques for large numbers of requirements are needed. Stronger evidence presented for prioritization techniques for medium sized numbers of requirements shows the techniques are more mature than newly designed techniques. All the studies in the medium-size category used a subjective measure of improvement based on the users' perceptions of levels of improvement. It seems that the evaluations are still not strong for these studies.

Some threats to research validity are contained in this study. In the data selection process only studies written in English are included. One threat to validity is that studies written in other languages might contain relevant studies, but these studies are not included. Another threat is that relevant studies may be contained in somewhere other than the identified search engines and journals. However, the identified search engines and journals are the main data sources in software engineering. These data sources would contain at least the major relevant studies. This threat can be controlled. Since only one person (the author of this paper) performed the study selection, study quality assessment and data extraction processes, personal opinion may affect the results of these processes. However, since the processes were rigidly performed according to the review protocol and any concern was discussed with the author's supervisor, these threats can be controlled.

It is found that there is a need for more studies on requirements prioritization techniques for large numbers of requirements. Future work could focus on further investigating the identified prioritization techniques for large numbers of requirements to get more evidence on the effectiveness of these techniques.

References

Aho, A. V., Hopcroft, J. E., & Ullman, J. D. (1983). *Data structures and algorithms*. Reading, MA: Addison-Wesley.

Aurum, A., & Wohlin, C. (2003). The fundamental nature of requirements engineering activities as a decision-making process. *Information and Software Technology, 45*(14), 945-954.

Aurum, A., & Wohlin, C. (2005). Requirements engineering: Setting the context. In A. Aurum & C. Wohlin (Eds.), *Engineering and managing software requirements* (pp. 1-16): Springer Berlin Heidelberg.

Avesani, P., Bazzanella, C., Perini, A., & Susi, A. (2005). *Facing scalability issues in requirements prioritization with machine learning techniques*. Proceedings of 13th IEEE International Conference on Requirements Engineering, 297 - 305.

Beck, K. (2000). *Extreme programming explained*. Reading, MA: Addison-Wesley.

Beg, R., Abbas, Q., & Verma, R. P. (2008). *An approach for requirement prioritization using B-Tree*. First International Conference on Emerging Trends in Engineering and Technology (ICETET'08), 1216 - 1221.

Berander, P., & Andrews, A. (2005). Requirements prioritization. In A. Aurum & C. Wohlin (Eds.), *Engineering and managing software requirements* (pp. 69-94): Springer Berlin Heidelberg.

Berander, P., & Svahnberg, M. (2008). Evaluating two ways of calculating priorities in requirements hierarchies - An experiment on hierarchical cumulative voting. *Journal of Systems and Software, In Press, Corrected Proof.*

Bergman, B., & Klefsjö, B. (2003). *Quality: From customer needs to customer satisfaction*. Lund, Sweden: Studentlitteratur.

Boehm, B. W. (1991). Software risk management principles and practices. *IEEE Software, 8*(1), 32-41.

Boehm, B. W., Grünbacher, P., & Briggs, R. O. (2001). Developing groupware for requirements negotiation: Lessons learned. *IEEE Software, 18*(3), 46-55.

Bradner, S. (1997). Key words for use in RFCs to indicate requirement levels. *RFC 2119*.

Brereton, P., Kitchenham, B. A., Budgen, D., Turner, M., & Khalil, M. (2007). Lessons from applying the systematic literature review process within the software engineering domain. *Journal of Systems and Software, 80*(4), 571-583.

Carlshamre, P. (2001). *A usability perspective on requirements engineering – From methodology to product development*. Unpublished Ph.D. thesis, Linking Institute of Technology, Sweden.

Cleland-Huang, J., & Mobasher, B. (2008). *Using data mining and recommender systems to scale up the requirements process*. Proceedings of the 2nd International Workshop on Ultra-large-scale Software-intensive Systems.

Cormen, T. H., Leiserson, C. E., Rivest, R., L, & Stein, C. (2001). B-Trees. In *Introduction to Algorithms* (2nd ed., pp. 434-454): MIT Press and McGraw-Hill.

Davis, A. (1993). *Software requirements: Objects, functions, and states*. Englewood Cliffs, New Jersey: Prentice-Hall International.

Doerr, J., Hartkopf, S., Kerkow, D., Landmann, D., & Amthor, P. (2007). *Built-in user satisfaction - Feature appraisal and prioritization with AMUSE*. 15th IEEE International Requirements Engineering Conference, 101-110.

DSDM Consortium. (2009). *DSDM public version 4.2*. Retrieved 6 Jan, 2009, from www.dsdm.org

Firesmith, D. (2004). Prioritizing requirements. *Journal of Object Technology, 3*(8), 35-47.

Gilb, T., & Maier, M. W. (2005). *Managing priorities: A key to systematic decision-making*. Proceedings of INCOSE Conference. Rochester NY USA. Retrieved 10 Aug 2008, from www.gilb.com/tiki-download_file.php?fileId=60

Goguen, J., & Jirotka, M. (1994). *Requirements engineering: Social and technical issues*. London: Academic Press.

Goldstein, H. (2005). Who killed the virtual case file? *IEEE Spectrum, 42*(9), 24-35.

Gotel, O., & Finkelstein, A. (1994). *An analysis of the requirements traceability problem*. 1st International Conference on Requirements Engineering (ICRE'94), 94-101.

Greer, D., & Ruhe, G. (2004). Software release planning: An evolutionary and iterative approach. *Information and Software Technology, 46*(4), 243-253.

Hatton, S. (2007). Early prioritisation of goals. In *Advances in conceptual modeling – Foundations and applications* (pp. 235-244).

Hatton, S. (2008). *Choosing the right prioritisation method*. 19th Australian Conference on Software Engineering, 517 - 526.

Henry, J., & Henry, S. (1993). *Quantitative assessment of the software maintenance process and requirements volatility*. In Proceedings of the ACM Conference on Computer Science, 346-351.

IEEE-STD 610.12-1990. (1990). Standard glossary of software engineering terminology. *Institute of Electrical and Electronics Engineers*.

IEEE-STD 830-1998. (1998). IEEE recommended practice for software requirements specifications. *IEEE Computer Society*.

Karlsson, J., & Ryan, K. (1997). A Cost-Value approach for prioritizing requirements. *IEEE Software, 14*(5), 67-74.

Karlsson, J., Wohlin, C., & Regnell, B. (1998). An evaluation of methods for prioritizing software requirements. *Information and Software Technology, 39*(14-15), 939-947.

Karlsson, L., Berander, P., Regnell, B., & Wohlin, C. (2004). *Requirements prioritisation: An experiment on exhaustive pair-wise comparisons versus planning game partitioning*. Proceedings of the 8th International Conference on Empirical Assessment in Software Engineering (EASE 2004), 145-154.

Karlsson, L., Host, M., & Regnell, B. (2006). *Evaluating the practical use of different measurement scales in requirements prioritisation*. Proceedings of the 2006 ACM/IEEE International Symposium on Empirical Software Engineering (ISESE'06), 326-335.

Karlsson, L., Thelin, T., Regnell, B., Berander, P., & Wohlin, C. (2007). Pair-wise comparisons versus planning game partitioning-experiments on requirements prioritisation techniques. *Empirical Software Engineering, 12*(1), 3 - 33.

Khan, K. A. (2006). *A systematic review of software requirements prioritization*. Unpublished master's thesis, Blekinge Institute of Technology, Ronneby, Sweden.

Kitchenham, B. (2004). *Procedures for performing systematic reviews*. Joint Technical Report, Computer Science Department, Keele University (TR/SE-0401) and National ICT Australia Ltd (0400011T.1).

Laurent, P., Cleland-Huang, J., & Duan, C. (2007). *Towards automated requirements triage*. 15th IEEE International Requirements Engineering Conference, 131 - 140.

Leffingwell, D., & Widrig, D. (2000). *Managing software requirements - A unified approach*. Upper Saddle River: Addison-Wesley.

Lehtola, L., & Kauppinen, M. (2004). *Empirical evaluation of two requirements prioritization methods in product development projects*. In Proceedings of the European Software Process Improvement Conference (EuroSPI 2004), 161-170.

Lehtola, L., & Kauppinen, M. (2006). Suitability of requirements prioritization methods for market-driven software product development. *Software Process Improvement and Practice, 11*(1), 7 - 19.

Lehtola, L., Kauppinen, M., & Kujala, S. (2004). Requirements prioritization challenges in practice. In *Product focused software process improvement* (pp. 497-508).

Lötter, H. P. P. (2000). How to judge scientific research articles. *SA Journal for Language Teaching*.

MacDonell, S. G., & Shepperd, M. J. (2007). *Comparing local and global software effort estimation models - Reflections on a systematic review*. First International Symposium on Empirical Software Engineering and Measurement, 401-409.

Machado, R. J., Ramos, I., & Fernandes, J. M. (2005). Specification of requirements models. In A. Aurum & C. Wohlin (Eds.), *Engineering and managing software requirements* (pp. 47-68): Springer Berlin Heidelberg.

Macintosh, G., & Gentry, J. W. (1999). Decision making in personal selling: Testing the "K.I.S.S. Principle". *Psychology and Marketing, 16*(5), 393-408.

Maiden, N. (2008). User requirements and system requirements. *IEEE Software*, *25*(2), 90 - 91.

Miller, G. A. (1956). The magical number seven, plus or minus two: Some limits on our capacity for processing information. *The Psychological Review*, *63*(2), 81-97.

Mustafa, M. A., & Al-Bahar, J. F. (1991). Project Risk Assessment Using the Analytic Hierarchy Process. *IEEE transactions on engineering management, 38*(1), 46-52.

New dawn technologies. (2009). *Beat the odds, making your IT projects a success*. Retrieved 02 Aug 2009, from http://www.newdawntech.com/webinar/beattheoddspdf.pdf

Ngo-The, A., & Ruhe, G. (2005). Decision support in requirements engineering. In A. Aurum & C. Wohlin (Eds.), *Engineering and managing software requirements* (pp. 267-286): Springer Berlin Heidelberg.

Nuseibeh, B., & Easterbrook, S. (2000). *Requirements engineering: A roadmap*. Proceedings of the Conference on the Future of Software Engineering, 35 – 46.

Perini, A., Susi, A., Ricca, F., & Bazzanella, C. (2007). An empirical study to compare the accuracy of AHP and CBRanking techniques for requirements prioritization. *Fifth International Workshop on Comparative Evaluation in Requirements Engineering*, 23-35.

Pettersson, F., Ivarsson, M., Gorschek, T., & Öhman, P. (2008). A practitioner's guide to light weight software process assessment and improvement planning. *Journal of Systems and Software, 81*(6), 972 - 995.

Pressman, R. S. (2001). *Software engineering: A practitioners approach.* (Fifth ed.): MacGraw-Hill.

Regnell, B., & Brinkkemper, S. (2005). Market-driven requirements engineering for software products. In A. Aurum & C. Wohlin (Eds.), *Engineering and managing software requirements* (pp. 287-308): Springer Berlin Heidelberg.

Regnell, B., Höst, M., Dag, J. N. O, Beremark, P., & Hjelm, T. (2001). An industrial case study on distributed prioritization in market-driven requirements engineering for packaged software. *Requirements Engineering, 6*(1), 51- 62.

Reifer, D. J. (2003). Is the software engineering state of the practice getting closer to the state of the art? *IEEE Software, 20*(6), 78-83.

Ruhe, G., Eberlein, A., & Pfahl, D. (2002). *Quantitative WinWin-A new method for decision support in requirements negotiation*. Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering, 159-166.

Ruhe, G., Eberlein, A., & Pfahl, D. (2003). Trade-off Analysis for Requirements Selection. *International Journal of Software Engineering and Knowledge Engineering, 13*(4), 345-366.

Saaty, T. L. (1980). *The analytic hierarchy process*. McGraw-Hill, New York.

Sidak, Z., Sen, P. K., & Hajek, J. (1999). *Theory of rank tests (probability and mathematical statistics)* (2nd ed.). San Diego, USA: Academic Press.

Somé, S. S. (2006). Supporting use case based requirements engineering. *Information and Software Technology, 48*(1), 43-58.

Sommerville, I. (1996). *Software engineering* (5th ed.). Wokingham, England: Addison-Wesley.

Sommerville, I., & Sawyer, P. (1997). *Requirements engineering - A good practice guide*. Chichester: John Wiley and Sons.

Staples, M., & Niazi, M. (2007). Experiences using systematic review guidelines. *Journal of Systems and Software, 80*(9), 1425-1437.

Thayer, R., & Dorfman, M. (1997). *Software requirements engineering* (2nd ed.): IEEE Computer Society Press.

Tudor, D., & Walter, G. A. (2006). Using an agile approach in a large, traditional organisation. *Proceedings of AGILE 2006 Conference (AGILE'06)*, 367-373.

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

Zave, P. (1997). Classification of research efforts in requirements engineering. *ACM Computing Surveys, 29*(4), 315-321.

Zowghi, D., & Coulin, C. (2005). Requirements elicitation: A survey of techniques, approaches, and tools. In A. Aurum & C. Wohlin (Eds.), *Engineering and managing software requirements* (pp. 19-46): Springer Berlin Heidelberg.

Appendix A: Why not Introduce Priority Groups Method

The priority groups method is not introduced in section 2.2 (Techniques of Requirements Prioritization). One reason is that the priority groups method here is different from general grouping requirements. General grouping requirements methods only group requirements into different priority groups once. The idea of priority groups here is that it assigns each requirement into one of the three groups: high priority, medium priority and low priority. If any group contains more than one requirement, three new subgroups will be created and the requirements (those within that group) are assigned into these subgroups. The process will be repeated until there is only one requirement in each subgroup. Figure 5 shows the idea of priority groups. The final result of the priority groups is like a list of ranked requirements. Karlsson et al. (1998) put it into the numerical assignment category when doing the evaluation. But the author of this paper thinks it belongs more to the simple ranking category. This conflict is one reason for excluding this method from the Techniques of Requirements Prioritization section.

Another reason is that the author of this paper also does not think this method is worth being introduced. The reason is that the results of this method are like simple ranking, but the presentation of the results seems more complicated than simple ranking. Just like Figure 5, many inner groups reside in outer groups making the presentation more complicated than just a simple ordered list of requirements. The evaluation results provided by Karlsson et al. (1998) (refer to Tables 4 and 5) also show that this method is very hard to use and it provides low result reliability and low fault tolerance.

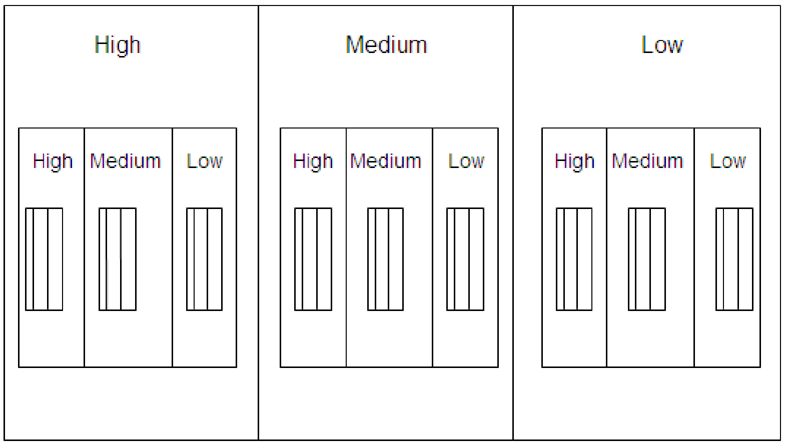


Figure 5: The idea of priority groups

Appendix B: Quality Assessment Form

The quality evaluation form (Table 15) is used to assess the quality of each primary study. For the answer part of the quality assessment form, "N" indicates no such information provided at all, "1" indicates poor quality and "5" indicates good quality. Data are extracted according to the data extraction form (Table 16).

Table 15: The quality assessment form

(a)	Title: Author(s): Publication year: Publication journal/conference/conference proceedings: Date of quality assessment:	
	Quality assessment questions	**Answers**
(b)	**Research problem**	**Non Poor Good**
	Does the study clearly present the research problem?	N 1 2 3 4 5
	Is the research problem well justified?	N 1 2 3 4 5
	Does the study clearly define the aim of the research?	N 1 2 3 4 5
(c)	**Literature review**	
	Does the study refer to the previous work on the same or similar research area?	No Yes
(d)	**Research method**	
	Is there any research method presented for the evaluation?	No Yes
	Is the choice of the research method well justified?	N 1 2 3 4 5
	What is the research methodology?	________________
	Is the choice of the research method appropriate for the evaluation?	No Yes
	What is the study setting?	________________
	What is the occupation of the participant?	N 1 2 3 4 5
	Is the evaluation adequately interpreted?	N 1 2 3 4 5

(e) Results Is there sufficient evidence derived from the evaluation? Is the evidence derived from the research adequately interpreted to derive the results?	 N 1 2 3 4 5 N 1 2 3 4 5
(f) Overall judgements Does the study contain high internal validity? Does the study contain high external validity?	 N 1 2 3 4 5 N 1 2 3 4 5

Table 16: The data extraction form

(a) Title: Author(s): Publication year: Publication journal/conference/conference proceedings: Date of data extraction:
(b) General information Does the study design a new technique? If yes, state the name of the new technique. Name of the prioritization technique What technique is introduced in order to manage requirements? How does the prioritization perform? Types of requirements claimed to be suited to Number of requirements claimed to be suited to Number of stakeholders claimed to be suited to
(c) Evaluation Is there any evaluation presented? If yes, is the technique evaluated in an industrial setting or academic setting? What is the methodology of the evaluation? What is the occupation of the participant? Number of requirements used for prioritizing requirements Types of requirements used for prioritizing requirements Number of participants used for prioritizing requirements Type of market the evaluation resides in The purpose of the prioritization When does the prioritization process perform?
(d) Outcome measurements and results What is the measurement of effectiveness? Effectiveness of the technique Strength of evidence of effectiveness

Appendix C: Customised Search String for each Pre-selected Search Engine

This section displays the customised search string details for each pre-selected search engine.

Search engine: ACM Digital library

Search string:
(((Title:requirements or Title:requirement) and (Title:prioritization or Title:prioritisation or Title:prioritize or Title:prioritise or Title:prioritizing or Title:prioritising or Title:prioritizes or Title:prioritises or Title:prioritizes or Title:prioritises)) or ((Abstract:requirements or Abstract:requirement) and (Abstract:prioritization or Abstract:prioritisation or Abstract:prioritize or Abstract:prioritise or Abstract:prioritizing or Abstract:prioritising or Abstract:prioritizes or Abstract:prioritises or Abstract:prioritized or Abstract:prioritised))) and (software)

Search engine: Google Scholar

Search string:
Google Scholar does not provide the service that enables key words to be searched for in the abstract field of the article. Therefore the key words are only searched for in the title field.

Allintitle: (requirements OR requirement) + (prioritization OR prioritisation OR prioritize OR prioritise OR prioritizing OR prioritising OR prioritizes OR prioritises OR prioritized OR prioritised)

Search engine: IEEE

Search string:
(((((requirements)<in>ti) <or> ((requirement)<in>ti)) <and> (((prioritization)<in>ti) <or> ((prioritisation)<in>ti) <or> ((prioritizing)<in>ti) <or> ((prioritising)<in>ti) <or> ((prioritize)<in>ti) <or> ((prioritise)<in>ti) <or> ((prioritizes)<in>ti) <or> ((prioritises)<in>ti) <or> ((prioritized)<in>ti) <or> ((prioritised)<in>ti))) <or> ((((requirements)<in>ab) <or> ((requirement)<in>ab)) <and> (((prioritization)<in>ab) <or> ((prioritisation)<in>ab) <or> ((prioritizing)<in>ab) <or> ((prioritising)<in>ab) <or> ((prioritize)<in>ab) <or> ((prioritise)<in>ab) <or> ((prioritizes)<in>ab) <or> ((prioritises)<in>ab) <or> ((prioritized)<in>ab) <or> ((prioritised)<in>ab)))) <and> (software)

Search engine: ProQuest

Search string:
This search engine limits the number of search terms. Therefore two split search strings are displayed.

((TITLE(requirements) OR TITLE(requirement)) AND (TITLE(prioritization) OR TITLE(prioritisation) OR TITLE(prioritize) OR TITLE(prioritise) OR TITLE(prioritizing) OR TITLE(prioritising) OR TITLE(prioritizes) OR TITLE(prioritises) OR TITLE(prioritized) OR TITLE(prioritised))) AND TEXT(software)

((ABS(requirements) OR ABS(requirement)) AND (ABS(prioritization) OR ABS(prioritisation) OR ABS(prioritize) OR ABS(prioritise) OR ABS(prioritizing) OR ABS(prioritising) OR ABS(prioritizes) OR ABS(prioritises) OR ABS(prioritized) OR ABS(prioritised))) AND TEXT(software)

Search engine: ScienceDirect

Search string:
((TITLE-ABSTR-KEY(requirements) or TITLE-ABSTR-KEY(requirement)) and (TITLE-ABSTR-KEY(prioritization) or TITLE-ABSTR-KEY(prioritisation) or TITLE-ABSTR-KEY(prioritizing) or TITLE-ABSTR-KEY(prioritising) or TITLE-ABSTR-KEY(prioritize) or TITLE-ABSTR-KEY(prioritise) or TITLE-ABSTR-KEY(prioritizes) or TITLE-ABSTR-KEY(prioritises) or TITLE-ABSTR-KEY(prioritized) or TITLE-ABSTR-KEY(prioritised))) and ALL(software)

Search engine: Springer

Search string:
Searches are limited to ten search terms in Springer. Therefore the four split search strings are displayed.

((title:requirements or title:requirement) and (title:prioritization or title:prioritisation or title:prioritizes or title:prioritises or title:prioritized or title:prioritised)) and (software)

((title:requirements or title:requirement) and (title:prioritize or title:prioritise or title:prioritizing or title:prioritising)) and (software)

((abstract:requirements or abstract:requirement) and (abstract:prioritization or abstract:prioritisation or abstract:prioritizes or abstract:prioritises or abstract:prioritized or abstract:prioritised)) and (software)

((abstract:requirements or abstract:requirement) and (abstract:prioritize or abstract:prioritise or abstract:prioritizing or abstract:prioritising)) and (software)

Appendix D: Study Quality Assessment

This section presents the data on the quality assessment for each identified primary study.

Questions / Paper No.	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
Research problem									
Does the study clearly present the research problem?	4	5	5	5	5	5	5	4	5
Is the research problem well justified?	N	1	5	4	1	4	2	2	5
Does the study clearly define the aim of the research?	5	4	5	5	5	5	5	5	5
Literature review									
Does the study refer to the previous work on the same or similar research area?	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes	Yes
Research method									
Is there any research method presented for the evaluation?	Yes	Yes	Yes	No	Yes	Yes	Yes	Yes	Yes
Is the choice of the research method well justified?	N	N	N	N/A	N	N	N	N	N
What is the research methodology?	Simulation	N	Experiment	N	Experiment	Case study	Case study	Experiment	Case study
Is the choice of the research method appropriate for the evaluation?	Yes	Yes	Yes	N/A	Yes	Yes	Yes	Yes	Yes
What is the study setting?	Academia	Academia	Academia	N	Academia	Industry	Industry	Academia	Industry
What is the occupation of the participant?	N	N	Masters students	N	Ph.D. Students and professor	N	Users, project manager, requirements engineer	Ph.D students and Junior Researchers	Domain expert, product area manager, project manager, group manager, quality manager, advanced engineer, tester, and developer
Is the evaluation adequately interpreted?	3	N	5	N	5	4	4	5	3
Results									
Is there sufficient evidence derived from the evaluation?	5	1	5	N	5	5	5	5	5
Is the evidence derived from the research adequately interpreted to derive the results?	5	N	5	N	5	5	5	5	3

Appendix E: Data Extraction

This section presents the data that are extracted from each identified primary study.

Paper [1]:

Title: *Facing scalability issues in requirements prioritization with machine learning techniques* Author(s): Avesani, P., Bazzanella, C., Perini, A., and Susi, A. Publication year: 2005 Publication journal/conference/conference proceedings: Proceedings of 13th IEEE International Conference on Requirements Engineering Date of data extraction: 22/05/2009
General information
Does the study design a new technique? If yes, state the name of the new technique. Yes, Case-based ranking
Name of the prioritization technique Ranking
What special technique is introduced in order to manage requirements? Machine learning technique
How does the prioritization perform? The system tries to let the stakeholders do a limited elicitation effort to generate an approximation of the final ranking. The system first selects a pair of requirements, then the stakeholders make the decision on which requirement they prefer, and finally the system uses machine learning techniques to generate an approximation of the final ranking.
Types of requirements claimed to be suited to Not specified
Number of requirements claimed to be suited to The paper states: “large”
Number of stakeholders claimed to be suited to The paper states: “single and multiple”
Evaluation
Is there any evaluation presented? Yes
If yes, is the technique evaluated in an industrial setting or academic setting? Academia
What is the methodology of the evaluation? Simulation
What is the occupation of the participant? Not specified
Number of requirements used for prioritizing requirements 25, 50, and 100 requirements
Types of requirements used for prioritizing requirements? Not specified
Number of people used for prioritizing requirements Not specified
Type of market the evaluation resides in Not specified
The purpose of the prioritization Not specified
When does the prioritization process perform? Not specified
Outcome measurements and results
What is the measurement of effectiveness?

User effort and accuracy
Effectiveness of the technique The user's effort is reduced. The simulation results show that on average it outperforms AHP with both human elicitation effort and the accuracy of the result.
Strength of evidence of effectiveness Weak

Paper [2]:

Title: An Approach for Requirement Prioritization Using B-Tree Author(s): Beg, R., Abbas, Q., and Verma, R.P. Publication year: 2008 Publication journal/conference/conference proceedings: First International Conference on Emerging Trends in Engineering and Technology. Date of data extraction: 24/05/2009
General information
Does the study design a new technique? If yes, state the name of the new technique. Yes, B-Tree method
Name of the prioritization technique Balanced search tree
What special technique is introduced in order to manage requirements? No other technique
How does the prioritization perform? B-Tree method has internal nodes that can contain variable numbers of child nodes. This method is coded to help users to prioritize requirements.
Types of requirements claimed to be suited to Not specified
Number of requirements claimed to be suited to The paper states: “large”
Number of stakeholders claimed to be suited to Not specified
Evaluation
Is there any evaluation presented? Yes
If yes, is the technique evaluated in an industrial setting or academic setting? Academia
What is the methodology of the evaluation? Not specified
What is the occupation of the participant? Not specified
Number of requirements used for prioritizing requirements 121 requirements
Types of requirements used for prioritizing requirements? Not specified
Number of people used for prioritizing requirements Not specified
Type of market the evaluation resides Not specified
The purpose of the prioritization Not specified
When does the prioritization process perform? Not specified
Outcome measurements and results
What is the measurement of effectiveness?

Number of comparisons
Effectiveness of the technique The number of comparisons is dramatically reduced. But the paper does not state the accuracy of the result.
Strength of evidence of effectiveness Weak

Paper [3]:

Title: Evaluating two ways of calculating priorities in requirements hierarchies – An experiment on hierarchical cumulative voting Author(s): Berander, P., and Svahnberg, M. Publication year: 2008 Publication journal/conference/conference proceedings: Journal of Systems and Software Date of data extraction: 27/05/2009
General information
Does the study design a new technique? If yes, state the name of the new technique. Yes, but not named.
Name of the prioritization technique Hierarchical cumulative voting (HCV)
What special technique is introduced in order to manage requirements? Hierarchical technique
How does the prioritization perform? Cumulative voting is also known as Hundred dollar method (for details please see chapter 2). HCV uses the cumulative voting technique to prioritize requirements that are structured hierarchically. A compensation factor is introduced when using HCV to prioritize requirements in order to deal with unbalanced hierarchy.
Types of requirements claimed to be suited to Hierarchical requirements
Number of requirements claimed to be suited to Not Specified
Number of stakeholders claimed to be suited to Not specified
Evaluation
Is there any evaluation presented? Yes
If yes, is the technique evaluated in an industrial setting or academic setting? Academia
What is the methodology of the evaluation? Experiment
What is the occupation of the participant? Master students
Number of requirements used for prioritizing requirements 27 requirements
Types of requirements used for prioritizing requirements? High-level and lower-level, functional requirements
Number of people used for prioritizing requirements 18
Type of market the evaluation resides in Not specified
The purpose of the prioritization Find important requirements
When does the prioritization process perform? Not specified
Outcome measurements and results
What is the measurement of effectiveness?

Ease of use, scalability, and accuracy
Effectiveness of the technique HCV with a compensation factor provides more accurate results than without a compensation factor. Besides that, HCV is reasonably easy to use, and can provide reliable results. HCV is scalable if the number of low-level requirements does not grow considerably.
Strength of evidence of effectiveness Medium

Paper [4]:

Title: Using data mining and recommender systems to scale up the requirements process Author(s): Cleland-Huang, J., and Mobasher, B. Publication year: 2008 Publication journal/conference/conference proceedings: Proceedings of the 2nd international workshop on Ultra-large-scale software-intensive systems Date of data extraction: 28/05/2009
General information
Does the study design a new technique? If yes, state the name of the new technique. Yes, but not named.
Name of the prioritization technique Numerical assignment
What special technique is introduced in order to manage requirements? Data-mining technique
How does the prioritization perform? The system dynamically assigns stakeholders to appropriate forums, and then the stakeholders work collaboratively to prioritize requirements.
Types of requirements claimed to be suited to The paper states: “unstructured or semi-structured data”
Number of requirements claimed to be suited to The paper states: “massive”
Number of stakeholders claimed to be suited to The paper states: “broad stakeholders”
Evaluation
Is there any evaluation presented? No
If yes, is the technique evaluated in an industrial setting or academic setting? N/A
What is the methodology of the evaluation? N/A
What is the occupation of the participant? N/A
Number of requirements used for prioritizing requirements N/A
Types of requirements used for prioritizing requirements? N/A
Number of people used for prioritizing requirements N/A
Type of market the evaluation resides in N/A
The purpose of the prioritization N/A
When does the prioritization process perform? N/A
Outcome measurements and results
What is the measurement of effectiveness?

N/A
Effectiveness of the technique N/A
Strength of evidence of effectiveness Weak

Paper [5]:

Title: Pair-wise comparisons versus planning game partitioning—experiments on requirements prioritisation techniques Author(s): Karlsson, L., Thelin, T., Regnell, B., Berander, P., and Wohlin, C. Publication year: 2007 Publication journal/conference/conference proceedings: Empirical Software Engineering Date of data extraction: 30/05/2009
General information
Does the study design a new technique? If yes, state the name of the new technique. No
Name of the prioritization technique Pair-wise comparisons (PWC), Planning Game, and Tool-supported pair-wise comparisons (TPWC)
What special technique is introduced in order to manage requirements? Computer tool (TPWC)
How does the prioritization perform? See chapter 2
Types of requirements claimed to be suited to Not specified
Number of requirements claimed to be suited to Not specified
Number of stakeholders claimed to be suited to Not specified
Evaluation
Is there any evaluation presented? Yes
If yes, is the technique evaluated in an industrial setting or academic setting? Academia
What is the methodology of the evaluation? Experiment
What is the occupation of the participant? Ph.D. Students and professor
Number of requirements used for prioritizing requirements 16
Types of requirements used for prioritizing requirements? High-level and independent requirements
Number of people used for prioritizing requirements First experiment: 8 (total 16 participants, but 8 participants prioritize more than 14 requirements), second experiment: 30
Type of market the evaluation resides in Market-driven
The purpose of the prioritization Find high value and low cost requirements
When does the prioritization process perform? Not specified
Outcome measurements and results
What is the measurement of effectiveness? User effort and accuracy
Effectiveness of the technique The results show that PWC is time consuming and the least easy to use compared with the other two. TPWC is the fastest

technique and it is as easy to use as planning game. There is no significantly difference regarding accuracy among the three techniques.
Strength of evidence of effectiveness Medium

Paper [6]:

Title: Towards Automated Requirements Triage Author(s): Laurent, P., Cleland-Huang, J., and Duan, C. Publication year: 2007 Publication journal/conference/conference proceedings: 15th IEEE International Requirements Engineering Conference Date of data extraction: 06/06/2009
General information
Does the study design a new technique? If yes, state the name of the new technique. Yes, Pirogov
Name of the prioritization technique Any prioritization technique
What special technique is introduced in order to manage requirements Clustering techniques
How does the prioritization perform? Requirements are first automatically clustered according to different goals such as feature sets, business goals or high level use cases. Stakeholders then manually prioritize the clusters by using any of the traditional prioritization techniques. The system then automatically generates a list of ranked requirements.
Types of requirements claimed to be suited to Not specified
Number of requirements claimed to be suited to The paper states: “large”
Number of stakeholders claimed to be suited to The paper only states: “a large set of stakeholders”
Evaluation
Is there any evaluation presented? Yes
If yes, is the technique evaluated in an industrial setting or academic setting? Industry
What is the methodology of the evaluation? Case study
What is the occupation of the participant? Not specified
Number of requirements used for prioritizing requirements 202 requirements
Types of requirements used for prioritizing requirements? Functional and non-functional requirements
Number of people used for prioritizing requirements Not specified
Type of market the evaluation resides in Bespoke
The purpose of the prioritization Find important requirements
When does the prioritization process perform? Not specified
Outcome measurements and results
What is the measurement of effectiveness? User effort and accuracy
Effectiveness of the technique The user's effort is reduced. But it does not return perfect results.

Strength of evidence of effectiveness Medium

Paper [7]:

Title: Suitability of Requirements Prioritization Methods for Market-driven Software Product Development Author(s): Lehtola, L., and Kauppinen, M. Publication year: 2006 Publication journal/conference/conference proceedings: Software Process Improvement and Practice Date of data extraction: 04/06/2009
General information
Does the study design a new technique? If yes, state the name of the new technique. No.
Name of the prioritization technique Pair-wise comparison technique
What special technique is introduced in order to manage requirements? No other technique is introduced
How does the prioritization perform? See chapter 2.
Types of requirements claimed to be suited to Not specified
Number of requirements claimed to be suited to Not specified
Number of stakeholders claimed to be suited to Not specified
Evaluation
Is there any evaluation presented? Yes
If yes, is the technique evaluated in an industrial setting or academic setting? Industry
What is the methodology of the evaluation? Case study
What is the occupation of the participant? Users of system
Number of requirements used for prioritizing requirements The paper states “ten categories of 20 or less requirements”
Types of requirements used for prioritizing requirements? User requirements
Number of people used for prioritizing requirements 4
Type of market the evaluation resides in Market-driven
The purpose of the prioritization Find important requirements
When does the prioritization process perform? Not specified
Outcome measurements and results
What is the measurement of effectiveness? User effort
Effectiveness of the technique • Participants feel that it is difficult to estimate how much one requirement is of more/less value than another. • It is difficult to prioritize more than 20 requirements. • It is difficult to prioritize requirements at different levels of abstraction. • Some users feel that it would be easier for them just to select the requirements which are most important for them.

Strength of evidence of effectiveness Medium

Paper [8]:

Title: An Empirical Study to Compare the Accuracy of AHP and CBRanking Techniques for Requirements Prioritization Author(s): Perini, A., Susi, A., Ricca, F., and Bazzanella, C. Publication year: 2007 Publication journal/conference/conference proceedings: Fifth International Workshop on Comparative Evaluation in Requirements Engineering Date of data extraction: 07/06/2009
General information
Does the study design a new technique? If yes, state the name of the new technique. No.
Name of the prioritization technique CBRanking (Case-based ranking) and AHP
What special technique is introduced in order to manage requirements? Two web based tools: SCORE (Supporting Case-based Oriented Rank Elicitation) and JAHP (Java Analytic Hierarchy Process)
How does the prioritization technique perform? This paper uses a web based tool named SCORE (Supporting Case-based Oriented Rank Elicitation) to support CBRanking method and a Java based implementation of AHP named JAHP to support AHP method.
Types of requirements claimed to be suited to Not specified
Number of requirements claimed to be suited to Not specified
Number of stakeholders claimed to be suited to Not specified
Evaluation
Is there any evaluation presented? Yes
If yes, is the technique evaluated in an industrial setting or academic setting? Academia
What is the methodology of the evaluation? Experimental
What is the occupation of the participant? Ph.D students and junior researchers
Number of requirements used for prioritizing requirements 20 requirements
Types of requirements used for prioritizing requirements? User, high-level, and independent requirements
Number of people used for prioritizing requirements 18
Type of market the evaluation resides in Not specified
The purpose of the prioritization Find important requirements
When does the prioritization process perform? Not specified
Outcome measurements and results
What is the measurement of the effectiveness? Accuracy
Effectiveness of the technique

The results show that AHP produces more accurate results than CBRanking.
Strength of evidence of effectiveness Medium

Paper [9]:

Title: A practitioner's guide to light weight software process assessment and improvement planning Author(s): Pettersson, F., Ivarsson, M., Gorschek, T., and Öhman, P. Publication year: 2008 Publication journal/conference/conference proceedings: Journal of Systems and Software Date of data extraction: 08/06/2009
General information
Does the study design a new technique? If yes, state the name of the new technique. No.
Name of the prioritization technique Cumulative voting (Hundred dollar method)
What special technique is introduced in order to manage requirements? No other technique is introduced
How does the prioritization perform? See chapter 2
Types of requirements claimed to be suited to Not specified
Number of requirements claimed to be suited to Not specified
Number of stakeholders claimed to be suited to Not specified
Evaluation
Is there any evaluation presented? Yes
If yes, is the technique evaluated in an industrial setting or academic setting? Industry
What is the methodology of the evaluation? Case study
What is the occupation of the participant? Domain expert, product area manager, project manager, group manager, quality manager, advanced engineer, tester, developer
Number of requirements used for prioritizing requirements 15
Types of requirements used for prioritizing requirements? Not specified
Number of people used for prioritizing requirements 27
Type of market the evaluation resides in Bespoke
The purpose of the prioritization Find important requirements
When does the prioritization process perform? The early stage
Outcome measurements and results
What is the measurement of the effectiveness? Accuracy
Effectiveness of the technique The paper states that participants disagree between each other. But this disagreement does not influence the planning of the initial effort. Individual participants vary in their satisfaction with the overall order of departments.
Strength of evidence of effectiveness

Weak

Appendix F: Project Timetable

The project timetable presents different milestones and the dates for starting and finishing each milestone.

Milestones	**Dates**
Research question and methodology justification	10/03/2009 – 24/03/2009
Define review protocol	25/03/2009 – 27/03/2009
Validate review protocol	27/03/2009 – 05/04/2009
Data selection	06/04/2009 – 02/05/2009
Study quality assessment	03/05/2009 – 15/05/2009
Study quality assessment results review	16/05/2009 – 21/05/2009
Data extraction	22/05/2009 – 08/06/2009
Data extraction results review	09/06/2009 – 25/06/2009
Data synthesis	26/06/2009 – 29/06/2009
Rewrite report	30/06/2009 – 21/08/2009
Refine report	22/06/2009 – 18/09/2009