Association for Information Systems
AIS Electronic Library (AISeL)

AMCIS 2009 Proceedings

Americas Conference on Information Systems (AMCIS)

2009

An Experimental, Tool-Based Evaluation of Requirements Prioritization Techniques in Distributed Settings

Sven Scheibmayr
University of Mannheim, scheibmayr@uni-mannheim.de

Tobias Hildenbrand
University of Mannheim, hildenbrand@wi.uni-mannheim.de

Michael Geisser
University of Mannheim, mgeisser@gmail.com

An Experimental, Tool-Based Evaluation of Requirements Prioritization Techniques in Distributed Settings

Sven Scheibmayr
University of Mannheim, Germany
scheibmayr@uni-mannheim.de

Tobias Hildenbrand
University of Mannheim, Germany
hildenbrand@wi.uni-mannheim.de

Michael Geisser
University of Mannheim, Germany
mgeisser@gmail.com

ABSTRACT

In this paper, we compare and analyze three common techniques for prioritizing software requirements: *Analytic Hierarchy Process* (AHP), *Cumulative Voting* (CV) and *Likert Scale Technique* (LST). These techniques are based on a ratio scale and are applied within this research as part of a *hierarchical* approach to requirements analysis on different levels of abstraction. For the systematic *evaluation* of these techniques in distributed settings, a controlled experimental setting was developed and carried out via the Internet. Therefore, a particular Web application was developed and data from 199 subjects was collected. The overall results indicate that LST is a simple, fast, and well-scaling prioritization technique, whereas slightly less precise than the other two techniques. However if accuracy is an important criterion, and a more complicated and slower technique is accepted, CV has proven to be most adequate. For the AHP, particularly when used with many requirements, a recommendation cannot be given because of poor scalability.

Keywords

Requirements Engineering, Requirements Prioritization, Analytic Hierarchy Process, Cumulative Voting, Likert Scale Technique, Experiment

INTRODUCTION

One of the most important phases within the software development process is requirements *analysis*. In this phase, the requirements to be actually implemented in the very next release are selected from the overall set (Sommerville, 2007). However, a large number of requirements is usually selected which then cannot be implemented due to limited resources (Firesmith, 2004). Moreover, globally distributed development scenarios in the software industry hinder collaboration and knowledge exchange in this critical process due to less mutual awareness and informal coordination (Regnell, Höst, Natt och Dag, Beremark and Hjelm, 2001). Nonetheless, sharing knowledge on the particular requirements' contents and customer value is essential for analysis, consolidation, and selection decisions.

In general, it is economically not reasonable to implement all the requirements, as there are different cost-value ratios among the individual requirements and in some cases, the value does not justify its cost. To face these limitations, the requirements can be prioritized (Firesmith, 2004). Reasonable and systematic prioritization represents a basis of decision-making for software release planning, allowing the most important functionality to be implemented first (Greer and Ruhe, 2004). This enables quick assessment of the software project's progress while increasing customer satisfaction (Firesmith, 2004).

Prioritization is of great benefit but it is also a complex task, as the stakeholders require profound knowledge about the application domain to estimate the importance of the requirements (Karlsson, Berander, Regnell and Wohlin, 2004). In many cases, almost all requirements are assumed utmost important (Berander, 2004; Firesmith, 2004). Furthermore, investigations should include the reliability and user-friendliness as well as the question if only the value or also the cost of requirements should be regarded as part of the prioritization metric (Berander and Andrews, 2005; Karlsson et al., 2004).

Various approaches to requirements prioritization have been described in literature (Berander and Andrews, 2005; Karlsson, Wohlin and Regnell, 1998), differing in terms of scales (ratio scale, ordinal scale) and abstraction level (Berander, Khan and Lehtola, 2006). In this research, techniques using *ratio* scale will be evaluated within a controlled experiment, since the ratio scale bears more information about the priorities than the ordinal scale and is an important prerequisite for a hierarchical

prioritization in larger projects (Karlsson, 1996; Berander and Jönsson, 2006). Therefore, we aim at evaluating the utility of different approaches in terms of efficiency, ease of use, and precision of results.

This work evaluates the fundamental techniques for requirements prioritization, which are the base for advanced methodologies like the Cost-Value Approach (Karlsson and Ryan, 1997) or Quantitative WinWin (Ruhe, Eberlein and Pfahl, 2002) (cp. (Berander et al., 2006)). Three ratio scale-based techniques are identified (AHP, CV, and LST) and explained in the next section. Section 3 then outlines the evaluation context and methodology as well as the design and actual conduction of the experiment. The quantitative results are discussed in section 4, whereas the last section provides a brief conclusion.

REQUIREMENTS PRIORITIZATION TECHNIQUES

In the following subsections, a brief introduction to the three techniques is given leading to their application to different levels of requirements abstraction and hierarchies.

Analytic Hierarchy Process (AHP)

The *Analytic Hierarchy Process* was developed by Thomas Saaty (Saaty, 1980) as a mathematical approach to decision making and was adapted for the prioritization of software requirements (Karlsson and Ryan, 1997). The AHP uses pairwise comparisons for prioritization by means of comparing two requirements relatively to each other. In total, every possible pair of requirements is compared. An individual comparison is usually based on a scale from one to nine. After the pairwise comparison of all requirements is completed, a percental value is calculated for every requirement (Karlsson and Ryan, 1997). In doing so, the AHP enables the possibility of testing the result for consistency. The disadvantage of the AHP and, thus, its main problem is represented by its poor scalability with respect to an increasing number of requirements as the number of comparisons grows quadratically with the number of requirements (Berander and Andrews, 2005). An evaluation by Lehtola and Kauppinen (2004) provides evidence for this problem.

Cumulative Voting (CV)

Cumulative Voting (Berander and Andrews, 2005), also known as *100-Dollar-Test* (Leffingwell and Widrig, 2000), is a very simple technique for requirements prioritization (Berander and Andrews, 2005). In this approach, the stakeholders distribute a constant sum, e.g. 100, of imaginary units (points, dollars, hours) to the individual requirements. The units allocated to one requirement in relation to the overall sum represent the relative priority of this requirement compared to the other requirements. CV also allows the distribution of the value zero, which is, on the other hand, not possible within the AHP (Berander and Jönsson, 2006).

Likert Scale Technique (LST)

The *Likert Scale Technique* is also a simple technique for the prioritization of requirements. It is based on a bipolar rating scale, which is used to rate the requirements regarding importance or costs. This scale is commonly used within the area of marketing research and is an established means for the elicitation of empirical data (Malhotra, 2008). The Likert Scale is a one-dimensional scale and both ends (poles) are marked with opposing statements (McIver and Carmines, 1981). Within requirements prioritization one pole would for example represent the rating "very important" while the other denotes "very unimportant". Between these poles, there are usually several discrete values. The first step of the LST is rating the requirements by choosing the respective points (Wiegers, 1999). Every point in the Likert Scale represents a numeric value. In a 7-stage Likert Scale, these would correspond to the values 1 to 7. In general, these values can be described as a_1 to a_n, *n* being the number of requirements. As these requirements are ranked relatively to each other, this technique is, therefore, based on the ratio scaling and the values are added, dividing every value by the resulting sum. In fact, the following calculation is performed: $S = \sum_{i=1}^{n} a_i$ and $b_i = a_i / S$ for all a_i. The calculated values b_1 to b_n represent the percental value figures of the *n* requirements.

Levels of Abstraction and Requirements Hierarchies

For prioritization reasons, requirements can be specified according to different levels of abstraction, since some can be kept quite general while others are very specific. Gorschek and Wohlin (2006) have developed an abstraction model for requirements, which is divided into four dimensions: *product level*, *feature level*, *function level* and *component level*, ordered from the most abstract to the most specific. Requirements of a more abstract dimension can be broken down into several requirements of a more specific level, thus, creating *hierarchies* of requirements. For instance the requirement "support of multiple languages" on the feature level can be broken down into several subordinate requirements, such as "possibility of

choosing the language while the system is running", "support functionality in the chosen language", "possibility to add languages to the system" on the function level (Gorschek and Wohlin, 2006).

These hierarchical levels of requirements abstraction can be used to facilitate prioritization (Berander and Jönsson, 2006). If the requirements are placed into a hierarchical structure, the complexity of the prioritization can be decreased significantly (Berander and Jönsson, 2006). The requirements are divided into groups, which are allocated to a requirement of a higher hierarchy level. The prioritization then only occurs to the requirements of this group. The AHP technique was successfully applied in hierarchical structures of requirements (Karlsson et al. 1998). The number of the necessary comparisons is thereby significantly reduced which leads to increased scalability. The execution of the AHP with 25 requirements would require 300 comparisons using the flat version. With the introduction of two hierarchy levels and the distribution of requirements into five groups with five requirements respectively, the required number of comparisons is decreased to 60 (50 for the lower hierarchy level and 10 for the upper hierarchy level).

This hierarchical prioritization approach was also adapted to CV techniques by Berander and Jönsson (2006) and called *Hierarchical Cumulative Voting (HCV)*. The approach also applies to all the other techniques delivering results that are based on ratio scaling, so it as well applies to AHP and LST, as the necessary calculations are only possible on the ratio scale (Berander and Jönsson, 2006). Therefore this paper applies all three of the evaluated techniques using a hierarchical approach for scalability and external validity reasons. In doing so, the hierarchical approach is evaluated using two distinct hierarchy levels: A higher level with requirements groups or sets and a lower level with requirements that are allocated to these sets.

CONTEXT OF EVALUATION AND RESEARCH METHODOLOGY

The following subsections present the hypotheses, which were evaluated within this study, related work, the conducted experiment and some limitations of the evaluation.

Hypotheses

In particular, the following hypotheses, which derive from the features of the different techniques[1] and are based on variables in the evaluation guidelines by Berander et al. (2006), are to be examined:

1. The *time needed* for the AHP is longer than that of the other techniques LST and CV, since the number of comparisons grows quadratically with the number of requirements.
2. The *perceived duration* of the AHP is longer than that of the other two techniques, LST and CV, since the number of comparisons grows quadratically with the number of requirements.
3. The AHP scales worse compared to the other techniques, i.e. the *relative increase of the time needed* for the AHP is higher than for the other two techniques, LST and CV, when the number of requirements is increased, since the number of comparisons grows quadratically with the number of requirements.
4. The LST is easier to *learn* than the other two techniques, AHP and CV, since the LST is less complex.
5. The LST is easier to *use* than the other two techniques AHP and CV, since the LST is less complex.
6. The LST is less *precise* than the other two techniques, AHP and CV, since the LST has the lowest resolution of the three techniques in terms of scale.
7. The LST is the most *attractive* of the three techniques, as it is less complex than the other techniques.

In order to test the hypotheses, the following dependent variables need to be measured: *Duration*, *perceived or estimated duration*, *learnability*, *usability*, *accuracy or precision*, and *attractiveness*. Almost all of these variables are subjective variables, i.e. they are based on the subjects' personal assessments (Wohlin, Runeson, Höst, Ohlsson, Regnell and Wesslén, 2000). Only the *duration* variable can be measured objectively. The *accuracy* variable is an exception as well, as it is determined within a blind test. Therefore, it could also be considered more objective. The evaluation guidelines by Berander et al. (2006) also describe the variables scalability and fault tolerance. Scalability is not measured explicitly here, but is determined through the duration variable by comparing different numbers of requirements in the prioritization. The fault tolerance variable, on the other hand, was removed from the experimental design after the subjects exhibited major difficulties regarding its assessment during a pre-test.

[1] These features are resolution, complexity and number of activities i.e. comparisons (cp. Berander and Andrews, 2005).

Related Work

Several studies regarding requirements prioritization have already been conducted in the past. These included the work of Karlsson (1996), Karlsson et al. (1998), Regnell et al. (2001), and Karlsson et al. (2004), which evaluate techniques for requirements prioritization like AHP, Numerical Assignment, CV, or Bubblesort. All studies mentioned chose an arbitrary approach ("Convenience Sample") to the selection of their sample in terms of the subjects. In contrary to randomized sampling, the subjects were chosen by the easiest means possible, for instance by choosing all students of a certain course (cp. Wohlin et al. (2000)). Another characteristic of these studies is the fact that they used small samples with few subjects. At most, 16 subjects were used. This smaller number was considered as a possible threat to external validity and, hence, generalizability in several studies (e.g. Karlsson et al. (1998); Karlsson et al. (2004)).

Furthermore, the prioritization approaches were seldom applied *hierarchically* within the studies mentioned. Only Karlsson et al. (1998) and Regnell et al. (2001) evaluated the hierarchical version of the AHP and CV, respectively. None of the studies, however, included a comparison of different hierarchical techniques. Therefore, and due to its practical relevance in larger projects, this shall be done within this paper. In addition the LST will be included in the evaluation, which has not been examined in previous studies.

Experimental Design

For this research, an *experiment in an artificial environment* has been chosen as evaluation method. This type of experiment can be easily conducted in an academic environment and it offers a high level of control (Zelkowitz and Wallace, 1998). This paper's goal is the evaluation of the impact of the independent variables (factors) *prioritization technique* and *number of requirements* on the before mentioned dependent variables by means of its controlled manipulation. The second factor was chosen as a means to evaluate the *scalability* of the prioritization techniques. For the first factor three factor levels (treatments) including the LST, CV, and AHP technique and for the second factor two factor levels defined by "many requirements" and "few requirements" were applied.

To measure the second factor "number of requirements" the subjects were randomly allocated to the factor levels "many requirements" and "few requirements". Within these factor levels and referring to the factor prioritization technique, the subjects are however regarded as a block. That means that *repeated measures* were conducted (Wohlin et al, 2000). In doing so, each of the subjects executes each of the three prioritization techniques. Thus, the dependent variables (learnability, usability, etc.) were measured repeatedly. In this process, the order of the techniques again is randomized to minimize sequence effects.

Furthermore, the prioritization techniques were applied hierarchically. In this process, two distinct hierarchy levels were applied: Groups of requirements, called *requirements sets* within the experiment and the lower level *requirements* themselves. Group A, that had to prioritize *many* requirements as a factor level, was given five requirements sets and a total number of 16 requirements, divided into 1×4, 1×5, 1×3 and 2×2 requirements sets. Group B, having to prioritize only *few* requirements, was given a total of nine requirements divided into three sets of 4, 3 and 2 requirements.

Subjects of group A hence had to prioritize 177,8% of the amount of requirements of the subjects of group B. Group A also had one large requirements set with 5 requirements and a total of 5 requirements sets. As the number of comparisons grows quadratically with the amount of requirements within the AHP, 10 comparisons are necessary for 5 requirements, while for 4 requirements, 6 comparisons are sufficient. Group A had to conduct two times 10 comparisons. Once for the largest requirements set and once for prioritizing the five high level requirements sets with each other. Altogether, 73 activities (comparisons in the AHP, allocation of values in CV and LST) had to be conducted through the given requirements by group A. Participants of group B, on the other hand, only had to conduct 37 activities. Subjects of group A thus had to conduct about twice as many activities compared to the subjects of group B (see table 1).

	LST	CV	AHP	Total
Number of activities, Group B (9 Req.)	12	12	13	37
Number of activities, Group A (16 Req.)	21	21	31	73
Increase, A compared to B (in percent)	75%	75%	138,5%	97%

Table 1. Overview of the Number of Activities

The software project chosen within this study whose requirements had to be prioritized was a *Web browser*. The requirements were software requirements on a relatively high level of abstraction, as they were pointed at the Web browser's users and should therefore not contain any technical details. Every requirement had a unique title and a description. In the following listing, a few examples are given:

- **Print function**: enables the printing of Web sites.
- **Full-screen browsing**: The Web site is shown on the whole screen. Desktop and window frame are no more visible.
- **Search engine input box**: an input box for a search engine next to the address bar (e.g. Google or Yahoo).

For this study, mainly students are used as subjects to evaluate the prioritization techniques since they are appropriate for this study due to a certain level of experience and high cost-efficiency. The subjects participated voluntarily in the experiment, their motivation was therefore not influenced by any academic goals, such as the achievement of a certain grade or amount of credits. The majority of the sample consisted of students from the areas of computer science, information systems, and economics.

In total, complete data sets of 199 subjects could be collected. This sample size significantly exceeded the sizes used in the related studies mentioned above, where the maximal amount of subjects was 16. The relatively large size of the sample increases the external validity of the results of the experiment (Wohlin et al., 2000).

Prior to the actual experiment, a pre-test was conducted with five students to find out about and correct possible problems and difficulties in comprehension. This showed for instance, that the question for the variable fault tolerance created difficulties with the subjects, as they were not able to assess it reasonably. Consequently, this variable was excluded from the actual experiment. It is similar to the variable accuracy (Berander et al., 2006).

Conducting the Experiment

A Web application called *IBERE* was used for the experiment of this paper and adapted accordingly. IBERE is short for "Internet-Based Empirical Requirements Evaluation" and was initially developed to facilitate distributed collaborative requirements engineering by visual cost-value-based decision support (Geisser and Hildenbrand, 2006). IBERE is an Internet-based application conceived for the geographically distributed execution of the requirements prioritization. The subjects conducted the entire data capturing for the experiment by means of the IBERE Web application, which also facilitates the validation of the data.

All three prioritization techniques were implemented within the IBERE engine. IBERE is able to manage requirements distributed into various requirements sets and supports both of the hierarchy levels used within this experiment. IBERE automatically guides the user through the process of the requirements prioritization. After the subjects had entered some personal data on the first web page a site explaining the experiment appeared. Afterwards, the three prioritization techniques were conducted in randomized order for each subject. To ensure the subjects understanding of the requirements, mouseover descriptions were shown. After the subjects had conducted the rating, the proportion of the values of the individual requirements to the total value was calculated based on the according prioritization technique.

The dependent subjective variables learnability, usability and accuracy were measured on a six-level Likert Scale. The attractiveness was measured based on a nominal scale, so that the subjects had to choose one technique. The estimated duration was given in minutes by the subjects (perceived time spent). The objective duration was saved by the IBERE application. After the prioritization phase, the blind test for the accuracy variable was conducted. The subjects were presented their own priorities determined by the techniques in order to evaluate these results according to their overall opinion.

Limitations

As the experiment was conducted with students, there was a chance for decreased external validity. Students possibly prioritize requirements differently than professional software engineers or customers in real projects. It is possible that stakeholders identify more requirements as high priority in real projects.

Another problem was the fact that relatively few requirements were used for the experiment. Therefore the question arises whether the prioritization techniques are suitable for a higher amount of requirements in real world projects. The fact that only two hierarchy levels were used for this experiment can additionally be regarded as a limitation for external validity.

The measuring of the accuracy variable may be problematic in terms of validity, since the measurement of this variable in the

blind test occurred directly after the execution of the prioritization. The subjects were possibly still influenced by it. A measurement of the variable occurring several weeks later would be better as the subjects would less recall their own prioritization. Due to the difficulty of convincing subjects to participate two or more times in an experiment, this was impossible with the resources at disposal for this study. In addition there could be a minor risk that the software tool used to conduct the experiment could have affected the measurement of the variable usability, as it may have influenced the subjects' perception on the usability of the techniques. However the participants were advised to evaluate only the generic techniques.

As three prioritization techniques were conducted subsequently, the techniques, which were conducted at first, could have influenced how the subjects executed the following techniques. This risk was reduced, by randomly choosing the sequence of the techniques and the sequence of the requirements during each technique.

As students are most probably users of web browsers they represent appropriate subjects for the evaluation. Nevertheless, limitations to external validity could emerge, as there could be more relevant factors, e.g. experience in software development[2], within an industrial scenario, than within a simplified artificial environment (Carver, Jaccheri, Morasca and Shull, 2003). Another problem that cannot be excluded and possibly impacts the results of the experiment is that stakeholders involved in a project may prioritize differently compared to the subjects within this experiment (cp. (Berander, 2004)). Nevertheless, this study can deliver valuable results with respect to hierarchical requirements prioritization in distributed settings.

OVERALL FINDINGS AND DISCUSSION

Table 2 shows the results of the measured variables. Shown are mean values with exception of Attractiveness, which was measured on a nominal scale and thus is shown as the percentage of subjects in favor for a technique. Duration and perceived duration are denoted in seconds for both groups A (16 requirements) and B (9 requirements). Analysis for the other variables was conducted for both groups together because no statistically significant differences for the variables were found between the two groups. Learnability, Usability and Accuracy were measured on a scale of 0 to 5.

	LST	**CV**	**AHP**
Duration			
Group A	117	204	289
Group B	95	140	155
Perceived duration			
Group A	232	281	306
Group B	184	211	235
Learnability	4,67	4,16	4,06
Usability	4,59	3,67	4,01
Accuracy	2,95	3,49	3,33
Attractiveness	47,24%	31,16%	21,61%

Table 2. Results of Measured Variables

Table 3 shows an evaluation of the statistical results of the experiment. Each criterion of comparison was given a certain amount of points (•), according to the rank of the technique, as demonstrated in the table. This means that the best technique received three points and the worst one point. The same amount of points was given if the mean values were so close that no statistically significant difference could be proven.

[2] Almost half of the subjects (49%) declared that they had at least a few experiences in software development.

	LST	CV	AHP
Duration	●●●	●●	●
Scalability	●●●	●●	●
Learnability	●●●	●●	●●
Usability	●●●	●	●●
Accuracy	●●	●●●	●●●
Attractiveness	●●●	●●	●

Table 3. Evaluation of Experimental Results

In terms of *duration*, the LST clearly performed as the fastest technique. The CV was the second fastest and the AHP performed worst in that respect. The differences are especially clear on the factor level "many requirements". Therefore, the first hypothesis that the duration for the AHP is longer than for the LST and CV can be confirmed by rejecting the corresponding null hypothesis regarding accidental mean differences. The analysis also shows that CV needs more time than LST. As for CV and LST the number of activities grows linearly with the amount of requirements, the individual activity, i.e. the distribution of an amount of points to a requirement, takes more time in the CV than in the LST.

The *perceived duration* estimated by the subjects turned out similar to the objective duration, whilst the differences between the techniques were weakened. This was caused by the fact that the subjects overestimated the actual duration the more, the shorter it was. Consequently no significant difference could be determined with respect to the perceived duration of the AHP and the CV, but only between AHP and LST. Thus, the second hypothesis in section "Hypotheses" could not be fully confirmed.

In terms of *scalability*, the LST again took the first place. Comparing group A and group B, the relative increase of duration was the lowest with 23,2%. The AHP, on the other hand, exhibited the worst scalability. The third hypothesis, denoting that the AHP provides the worst scalability, could thus be confirmed. With an increase of 45.7%, the CV scales worse than the LST, even though the number of activities grows linear with the amount of requirements in both techniques. This indicates that in the CV, the individual allocation of points to a requirement becomes more difficult with an increasing amount of requirements. The AHP should not, however, be applied for a higher amount of requirements as it quickly reaches its limits due to its quadratical growth of activities.

The scalability problems can be reduced by a hierarchical approach to prioritization. However, the additional complexity that such hierarchical structures can reach should be put into question. With requirements of a very high level of abstraction, the stakeholders possibly have difficulties in rating them. Stakeholders can interpret requirements differently due to the high level of abstraction. For these reasons, users of the techniques should always be informed about which requirements in the hierarchy level below belong to the requirement to be rated. That such a complex hierarchical structure can possibly not be devised at all, represents another problem as there are many requirements on a level of abstraction that can hardly be associated to a requirement that is more abstract. The scalability is, therefore, of great importance to the prioritization process.

Hypothesis four, stating that the LST is *easiest to learn,* can be confirmed. Both the CV and the AHP were rated worse than the LST and did not differ much from each other. Altogether, all of the three techniques were apparently learned very well. The mean values were all above 4 in a scale of 0 to 5. Regarding a quick and easy learnability, no problems can therefore be expected, even with respect to hierarchical prioritization approaches.

The *usability* was well rated regarding all of the three techniques. All the values were located around 4. Here, the LST performed best again, followed by the AHP and the CV. Therefore, the fifth Hypothesis, that the usability is improved by the LST with respect to the AHP and CV was confirmed as well. The reason for this could be that the subjects generally perceive an assignment of numbers between 1 to 100 as more difficult than the relative comparisons conducted by the AHP (cp. also Regnell et al. (2001)).

While the LST performed best so far, it turned out to be less appropriate, compared to the other techniques regarding its *accuracy*. Therefore, the sixth hypothesis could also be confirmed. Although the difference of LST and the other two techniques was significant in statistical terms, it was only a slight difference of 0.38 compared to the AHP and 0.54 to the CV. In addition to the lower resolution of its rating scale, another reason for the worse performance of the LST could be that

some users did not consider rating requirements relatively in the LST. In general, the accuracy of all techniques was rated worse than the two other variables learnability and usability. With a rating of three points it was however still located in the upper half of the scale.

Regarding the *attractiveness*, the LST was in turn the clear winner. 47.2% of the participants found this technique to be the most attractive. The CV was placed second and the AHP third. This result also confirmed the eighth and last hypothesis. As the reason for the worst rating the AHP received, it can be assumed that it was considered tiresome due to many pairwise comparisons. Several subjects had already indicated this fact in the pre-tests.

CONCLUSION

To conclude, the LST was rated best in all evaluation criteria but accuracy. If a smaller amount of accuracy can be tolerated, the technique represents an alternative that is simple, quick, easy to learn, and easy to handle in practice. The CV represents the best alternative if high accuracy is to be achieved in requirements prioritization. Especially with a larger number of requirements, which are common in practice, the AHP can only be recommended for particular occasions, since its scalability is strongly limited.

However, the overarching question remains whether an extra effort for learning and using the CV and the larger amount of time needed for doing so justifies the improved accuracy. Especially considering the fact that the LST does not lie much behind regarding its accuracy.

When choosing a particular requirements prioritization technique, this trade-off must be evaluated according to the individual situation and context of the development project. It should be considered that the LST was regarded as the most attractive technique and that user *acceptance* plays a major role for the successful application of a technique.

Considering *distributed* requirements analysis processes, user acceptance and correct tool usage are even more critical. Therefore, choosing an appropriate technique implemented by adequate tool support represents a necessary initial step.

Taken together, the results presented in this paper provide valuable decision support for various practical prioritization contexts, not only in distributed settings.

REFERENCES

1. Berander, P. (2004) Using Students as Subjects in Requirements Prioritization, *Proceedings of the 3rd International Symposium on Empirical Software Engineering (ISESE'04)*, IEEE Computer Society, 167–176.
2. Berander P. and Andrews A. (2005) Requirements Prioritization, in A. Aurum and C. Wohlin (Eds.), *Engineering and Managing Software Requirements*, Springer, Berlin, Germany, 69–94.
3. Berander P. and Jönsson P. (2006) Hierarchical Cumulative Voting (HCV) - Prioritization of Requirements in Hierarchies, *International Journal of Software Engineering and Knowledge Engineering*, 16, 6, 819–849.
4. Berander, P., Khan K. A. and Lehtola L. (2006) Towards a Research Framework on Requirements Prioritization, *Proceedings of the 6th Conference on Software Engineering Research and Practice in Sweden (SERPS'06)*, 39–48.
5. Carver J., Jaccheri L., Morasca S. and Shull F. (2003) Issues in Using Students in Empirical Studies in Software Engineering Education, *Proceedings of the 9th International Software Metrics Symposium (METRICS'03)*, IEEE Computer Society, 239–251.
6. Firesmith D. (2004) Prioritizing Requirements, *Journal of Object Technology*, 3, 8, 35–47.
7. Geisser M. and Hildenbrand T. (2006) A Method for Collaborative Requirements Elicitation and Decision-supported Requirements Analysis, in S. F. Ochoa and G.-C. Roman (Eds.), *Advanced Software Engineering: Expanding the Frontiers of Software Technology*, Springer, Boston, USA, 108–122.
8. Gorschek T. and Wohlin C. (2006) Requirements Abstraction Model, *Requirements Engineering*, 11, 1,79–101.
9. Greer D. and Ruhe G. (2004) Software Release Planning: An Evolutionary and Iterative Approach, *Information and Software Technology*, 46, 4, 243–253.
10. Karlsson J. (1996) Software Requirements Prioritizing, *Proceedings of the Second International Conference on Requirements Engineering (ICRE'96)*, IEEE Computer Society, 110–116.

11. Karlsson J. and Ryan K. (1997) A Cost-Value Approach for Prioritizing Requirements, *IEEE Software*, 14, 5, 67–74.
12. Karlsson J., Wohlin C. and Regnell B. (1998) An Evaluation of Methods for Prioritizing Software Requirements, *Information and Software Technology*, 39, 14-15, 939–947.
13. Karlsson L., Berander P., Regnell B. and 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'04)*, IEEE Computer Society, 145–154.
14. Leffingwell D. and Widrig D. (2000) *Managing Software Requirements: A Unified Approach,* Addison-Wesley, Massachusetts, USA.
15. Lehtola L. and Kauppinen M. (2004) Empirical Evaluation of Two Requirements Prioritization Methods in Product Development Projects, *Proceedings of the 11th Conference on European Software Process Improvement (EuroSPI'04)*, Springer, 161–170.
16. Malhotra, N. K. (2008) *Marketing Research: An Applied Orientation*, Prentice Hall, Upper Saddle River, USA.
17. McIver J. P. and Carmines E. G. (1981) *Unidimensional Scaling*. Sage University Paper Series on Quantitative Applications in the Social Sciences, Sage, Newbury Park, USA.
18. Regnell B., Höst M., Natt och Dag J., Beremark P. and Hjelm T. (2001) An Industrial Case Study on Distributed Prioritisation in Market-driven Requirements Engineering for Packaged Software, *Requirements Engineering*, 6, 1, 51–62.
19. Ruhe G, Eberlein A. and 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 (SEKE'02)*, ACM Press, 159–166.
20. Saaty T. L. (1980) *The Analytic Hierarchy Process,* McGraw-Hill, New York, USA.
21. Sommerville I. (2007) *Software Engineering,* Addison-Wesley, Boston, USA.
22. Wiegers K. E. (1999) First Things First: Prioritizing Requirements, *Software Development*, 7, 9, 48–53.
23. Wohlin C., Runeson P., Höst M., Ohlsson M. C., Regnell B. and Wesslén A. (2000) *Experimentation in Software Engineering: An Introduction*. Kluwer, Boston, USA.
24. Zelkowitz M. V. and Wallace D. R. (1998) Experimental Models for Validating Technology, *IEEE Computer*, 31, 5, 23–31.