Tool-supported requirements prioritization: Comparing the AHP and CBRank methods

Anna Perini a,*, Filippo Ricca b, Angelo Susi a

a Fondazione Bruno Kessler IRST, Via Sommarive, 18 I-38100 Trento, Italy
b Unità CINI at DISI Laboratorio Iniziativa Software FINMECCANICA/ELSAG, spa CINI Viale Dodecaneso, 35 I-16146 Genova, Italy

A R T I C L E   I N F O

Keywords:
Controlled experiment
Requirements prioritization
AHP
Case-based ranking

A B S T R A C T

Requirements prioritization aims at identifying the most important requirements for a software system, a crucial step when planning for system releases and deciding which requirements to implement in each release. Several prioritization methods and supporting tools have been proposed so far. How to evaluate their properties, with the aim of supporting the selection of the most appropriate method for a specific project, is considered a relevant question.

In this paper, we present an empirical study aiming at evaluating two state-of-the-art tool-supported requirements prioritization methods, AHP and CBRank. We focus on three measures: the ease of use, the time-consumption and the accuracy. The experiment has been conducted with 23 experienced subjects on a set of 20 requirements from a real project. Results indicate that for the first two characteristics CBRank performs better than AHP, while for the accuracy AHP performs better than CBRank, even if the resulting ranks from the two methods are very similar. The majority of the users found CBRank the "overall best" method.

© 2008 Elsevier B.V. All rights reserved.

1. Introduction

Requirements prioritization aims at identifying the most important requirements for a software system [32], a key knowledge for an effective planning of the system releases. Given a set of candidate requirements, the challenge is to identify the subset that maximizes the fulfillment of technical constraints (such as limited development time and resources availability), business aspects (such as financial benefit or market strategies) and crucial stakeholders preferences.

Several studies, e.g. [13,23,26], analyze the role of requirements prioritization in the software development process. They aim at defining the features for an effective prioritization methodology. In parallel, a large number of methods and techniques for requirements prioritization have been proposed. They have been classified in the review proposed by Berander et al. [5] in two main categories.

The first category includes methods which assume that values can be assigned, by an expert, to different aspects of requirements (e.g. importance; penalty; cost; financial benefit; strategic benefit). These methods offer a way to elicit the expert’s evaluation on a requirement aspect and exploit a specific technique to compute the ranking of the set of requirements with respect to this aspect, playing the role of ranking criterion. Examples are the Analytic Hierarchy Process method (AHP) [16,31], the Case-Based Ranking method (CBRank) [4], Cumulative Voting [22] and Numerical Assignment [7]. Some approaches combine two or more single-criterion requirements rankings. Examples are: the cost-value approach [17] in which the value-based ordering is plotted against the cost-based ordering in a two dimensions diagram (both orderings are computed using AHP); Planning Game [19], which can be considered an extreme programming version of the cost-value approach; Wieger’s method [33], in which each requirement is evaluated on a scale from 1 to 9, along the following aspects: value for the customer, the penalty if it is not implemented, implementation costs, and risks.

The second category includes negotiation approaches in which requirements priorities result from an agreement among subjective evaluations by different stakeholders. An example is given by the Win–Win approach proposed in [29].

The ranking techniques used by these methods may differ in the way requirements priorities are computed, in the scale of values used to represent the resulting ordering and in the accuracy of the resulting ranking.

This variety of prioritization methods makes it hard to select the most appropriate one, for a given project. To address this problem, comparisons of prioritization methods have been performed through empirical studies. These works focus on specific features, like the accuracy of the prioritization results and the time spent during the whole requirements prioritization process, and provide
some useful measurements for the selection of the most suitable method, that is the method that achieves the best trade-off between the accuracy of requirements rankings and the effort of the decision makers to prioritize.

Main limits of the empirical studies conducted so far are the number and type of subjects involved (e.g. only three in [15]), or the number of requirements to be prioritized (e.g. only 13 requirements in [1]), or the heterogeneity of the methods compared (tool-supported versus pure manual methods as in [18]), which makes their results hardly comparable and generalizable to an industrial setting.

A deeper analysis of main reasons that make results of these empirical studies difficult to be compared (or in contradiction) is presented in [6] that reports an analysis of these works, based on a systematic review [20], and propose a framework for designing empirical studies on requirements prioritization methods and techniques.

In this paper, we present an empirical study to compare two state-of-the-art methods, namely AHP and CBRank.

AHP [31] is a multiple criteria decision-making method that has been adapted for prioritization of software requirements [16]. AHP allows to order a set of n candidate requirements on the basis of pairs evaluation. The evaluation effort goes quadratically with n. AHP is playing a pivotal role in empirical evaluations of prioritization methods.

The CBRank method [3,4], also exploits pair-wise elicitation. To limit the number of pairs that need to be elicited CBRank uses a machine learning algorithm that computes an approximation of the requirements ordering. This makes it applicable also to large sets of requirements. The properties of CBRank have been presented and compared with those of AHP in previous papers [3,4], but so far not yet analyzed in the context of a comparative empirical study.

We use tool-supported versions of the two methods, namely JAHPI for AHP and SCORE for CBRank. The tools are strict implementations of the methods’ prioritization processes and have homogeneous graphical user interfaces. We focus on three measures: the ease of use, the time-consumption and the accuracy, taking into account definitions proposed in [18] and general guidelines given in [6]. The experiment has been conducted with 23 experienced subjects on a set of 20 requirements from a real project.

The paper is structured as follows: In Section 2, related work are presented. In Section 3, the two prioritization methods, AHP and CBRank, are illustrated. Section 4 discusses relevant measures for comparing the two methods while the experiment definition and setting is given in Section 5. The experiment results are presented in Section 6 and conclusions are given in Section 7.

2. Related work

Comparative evaluations of requirements prioritization techniques and methods have been performed by means of experimental studies designed, mainly, with the purpose of measuring usability and performance properties.

Most of these works compare methods or techniques in pairs, i.e. the current state-of-the-art technique with a new one, proposed by the study’s authors.

A first extensive study on six techniques has been performed by [15]. The compared techniques included AHP, hierarchy AHP, priority groups, minimal spanning tree, bubblesort and binary search [2]. They have been evaluated against technical properties, such as ability of a techniques to indicate consistency in the decision maker’s judgment, technique’s ranking scale, time-consumption, and against subjective measures, namely, ease of use, reliability of result and fault tolerance with respect to judgmental errors.

For this evaluation, the authors exploited a 16 quality requirements project for a mobile phone system. Three evaluators were asked to prioritize requirements according to their importance, using the six techniques. The AHP-based techniques resulted as the most promising, according to this study, although presenting limits in scalability.

An analogous experiment has been performed more recently [1], on five techniques, including AHP, binary search, Planning Game, “100 Points Method” and a new method which combines Planning Game with AHP. The following properties were measured: time-consumption, ease of use, accuracy and scalability. Fourteen evaluators, Master and Ph.D students, were asked to prioritize a 13 requirements set. The binary search resulted to be the best technique for prioritizing requirements. According to the authors, both experiments are limited with respect to the number of requirements to be prioritized and the evaluators sample. An extensive analysis of empirical studies applied to prioritization methods and techniques has been performed in [20], which motivates the proposal of a research framework to conduct studies in the area of requirements prioritization [6].

In most of these studies AHP plays a pivotal role since it has been largely applied in decision-making tasks aiming at ranking a set of items, its properties have been deeply studied and supporting tools are available. A recent experiment [18], compares Planning Game to a tool-supported version of AHP, with the aim of characterizing the two approaches with respect to time-consumption, ease of use, and accuracy, as perceived by the evaluator. The empirical study has been conducted with 16 subjects on 8 and 16 requirements sets. Results state the superiority of the tool-supported AHP with respect to Planning Game for the ease of use and time-consumption while cannot state clearly which of the two methods is more accurate. In our work we refer to this study revisiting its design and attempting at overcoming some limitations pointed out by the authors with reference to the type and number of subjects of the experiments, the size of the case study and the definition and measurements of dependent variables.

Concerning the two prioritization methods considered in our work, they have been previously compared on simulated data [4]. In particular, CBRank has been compared both to AHP and to AHP with “stopping rules” [14], an heuristic for reducing the elicitation effort in AHP, which allows to limit the scalability problem, at the price of a reduced accuracy of the resulting requirements ranking. The results shown that CBRank outperforms AHP with respect to the trade-off between decision makers elicitation effort and result accuracy.

As the design and planning of the empirical study that will be discussed in this paper has been previously introduced in [28], together with a discussion on the different categories of the involved subjects (e.g., Ph.D students vs. researchers) and a summary of early results on measuring rank accuracy with 17 subjects (working session 1). Among the most relevant extensions described in this paper: additional measurements collected with new subjects (working session 2); analysis on ease of use and time-consumption; analysis of the effect of co-factors (e.g., method order), and of the answers provided by subjects to survey questionnaires.

3. Prioritization methods

In this section we describe the two methods that have been compared in this study and their supporting tools.

3.1. The AHP prioritization method

The Analytic Hierarchy Process (AHP) [31] is a multiple criteria decision-making technique based on a pair-wise comparison ap-
approach. It has been largely applied in software engineering, for instance in software package and component selection [21,25], in COTS evaluation [27] and in requirements prioritization [16].

Fig. 1A, sketches the prioritization process followed by a decision maker (user) when using the AHP method. Given a set of requirements specified by the user, the method allows to define a matrix whose rows and columns represent the candidate requirements. Each element of this matrix is defined by a pair of requirements and it will be assigned a value representing the user preference on the corresponding pair. Given a prioritization criterion, the user performs a set of activities aiming at achieving the pairwise evaluation of the whole set of requirements pairs (see activities “select a pair” and “give a preference” in Fig. 1A). Each pair is assigned an integer belonging to the scale [1,9] which represents a qualitative measure of the preference relation between the corresponding requirements (e.g. if the requirement A is “equally important” than requirement B respect to the given criterion, the value 1 is given, if the requirement A is “essentially more important” than requirement B the value 9 is given). When all the pairs have been evaluated, an ordering is computed via the AHP algorithm which rests on the computation of the eigenvalues of the requirements matrix. The resulting ordering is represented as a vector of weights that specifies the rank of each requirement.

In case of multiple criteria, the whole process is repeated for each criterion and a further step is required, that is the synthesis of a global rank based on a weighted composition of the different criteria orderings is computed. The weights are derived using an analogous preference elicitation process performed on a matrix where rows and columns represent the different criteria.

Among the main issues of AHP is the quadratic growth of the number of comparisons needed as the number of candidate requirements increases.

In Saaty [31] a technique is proposed to handle this scalability problem by introducing the so called “dominance hierarchy” whose top levels elements refer to criteria and the lowest level correspond to the requirements. Intuitively, following the dominance hierarchy, the general prioritization problem is decomposed into sub-problems allowing for a reduction of the elicitation effort, but, at the same time introducing a strong bias. In fact the dominance hierarchy reflects the a priori knowledge on the relative importance of the criteria, independently from the current candidate requirements.

Another approach to handle the AHP scalability problem in prioritizing requirements has been proposed in [16]. This approach rests on the exploitation of the local stopping rule technique proposed in [11]. This technique allows to determine when new pair-wise comparisons are no longer needed. In our experiment we used the basic AHP process, without considering the use of the stopping rules.

The AHP supporting tool we exploited in our experiment is JAHP2 (Java Analytic Hierarchy Process), a Java based implementation of the AHP algorithm that uses a scale of [1,3,5,7,9], a granularity that is considered sufficient to represent the possible values for decision makers preferences. JAHP allows to automate the steps of the AHP process shown in the left side of Fig. 1A.

To enable a distributed use of the tool, we integrated the basic JAHP computational component into a client–server application, adding a client-side graphical user interface which was designed in such a way to achieve homogeneity between the two GUIs of the tools under consideration. Fig. 2, shows a snapshot of the pair-wise evaluation web page where the user can choose the value of the relative importance between two requirements.

The tool supports the user in the whole elicitation process. In particular, after the authentication, the system presents the user the agenda of \( \frac{n(n-1)}{2} \) pair-wise comparisons. The user can

\footnote{More precisely, half of the matrix, \( \frac{n(n-1)}{2} \) values, is elicited from the user while the other half is computed by symmetry.}

\footnote{http://www.di.unipi.it/~morge/software/JAHP.html.}
analyze the description of the requirements for each pair, specify
the value of her preference in terms of the relative importance of
one requirement with respect to the other, by selecting one of
the radio buttons shown in Fig. 2. Once the user completed all
the proposed evaluations the system is able to compute the result-
ring rank of the requirements using the AHP algorithm.

3.2. The CBRank prioritization method

The Case-Based Ranking (CBRank) method [4] exploits machine
learning techniques to guide the user preferences elicitation in the
prioritization process. The framework rests on an iterative process
that can handle single and multiple decision makers (stakeholders)
and different criteria (both business goals and technical
parameters).

Fig. 1B sketches the basic steps of the prioritization process,
where manual elicitation interleaves with machine supported
steps.

The main input to the process is the collection of requirements
that have to be ranked. The final output of the process is an approx-
imation of the target ranking.

The pair sampling activity is an automatic procedure which se-
lects a pair (or a sample of pairs) of requirements on the basis of a
predefined selection policy which may take into account informa-
tion on the currently available rankings.

The user performs the evaluation of the requirements pairs, by
iterating the following steps till all the pairs in the sample have
been evaluated: select a pair from the sample; evaluate the relative
importance of the requirements in the pair. That is, given a pair of
requirements A and B, the user is asked to specify what is the
“most important” requirement among A and B with respect to
the given criterion. Differently from AHP, here there is no range
of values, the preference is strict. The output of this step is a set
of ordered pairs.

The ranking learning activity takes in input the stakeholder pref-
erences acquired in the previous step, and computes an approxima-
tion of the ranking function. The learning procedure is based on the
boosting approach described in [10] and may eventually exploit
also available knowledge on the requirements rankings induced
by other prioritization criteria (e.g. the cost for the realization of
the requirements, the estimated utility) defined on the initial set
of requirements in order to best approximate the final ranking.

If the ranking produced by the ranking learning activity can be
considered a good approximation (e.g. the error measures
exploited in the method are minimized) it is given in output, other-
wise it may become the input to a further iteration of the process.

The CBRank method is supported by a web-based tool named
SCORE (Supporting Case-based Oriented Rank Elicitation) [3]
which allows for a distributed use of the framework, to support
the pair-wise priority elicitation by distributed stakeholders.
Fig. 3 shows a snapshot of the SCORE graphical user interface.

The system supports the whole evaluation process. In particu-
lar, SCORE presents the user an agenda of comparisons. The user
can analyze each one of the pairs specifying the preferred require-
ment in the pairs, by indicating which one of the requirement is
“the most important” (see the system user interface showed in
Fig. 3). Finally, once all the evaluations have been performed, the
system computes the rank and, in the case of a further iteration,
it presents to the user the set of new pairs of requirements to be
evaluated.
4. Measuring method properties

To perform an effective comparison of the two requirements prioritization methods a key issue is to decide which method’s properties to measure. We discuss it briefly in this section focusing on two aspects: the prioritization process that a user undertakes using one of the two methods, and the quality of the result produced as output by this process, namely the ranking of the candidate set of requirements. The latter is usually evaluated in terms of accuracy of the resulting ranking. But different definitions of accuracy—and different ways of measuring it—can be found in state-of-the-art evaluations, as discussed in the following. Concerning the evaluation of the prioritization process, we take the point of view of the user of a method and consider time-consumption and perceived ease of use, as two relevant properties.

4.1. Process properties

The time-consumption of a prioritization process is defined as the time interval between the time we get the final ranking (end time of the prioritization task) and the time the user starts prioritizing (start time of the prioritization task). This time measure results from the sum of the times spent in each step of the prioritization process sketched in Fig. 1. In practice the most time-consuming step is the give preference step, which is repeatedly performed by a method’s user. Time-consumption can be measured by asking the subjects to annotate the start and end times of prioritization tasks (we call it declared time). In the case of tool-supported methods, the start and end times can be recorded by the tool and the difference computed automatically (actual time).

The ease of use of a prioritization process rests on a subjective evaluation, which is performed by means of direct questions to a subject. Several factors can influence this subjective measure. First of all the cognitive effort in charge of the user when providing the requested input, a pair preference in our case, second the number of times this activity needs to be iterated to conclude the prioritization task.

4.2. Result quality

Rank accuracy can be defined as the measure of how much the ranking computed while using a given prioritization approach is close to the ideal target ranking. In the case of requirements, the ideal target ranking, is usually not known a priori. We may define it as (definition 1) the ranking the decision maker has in mind based upon some implicit considerations, or as (definition 2) the result of a negotiation process among several decision makers.

5. The experiment

Here, we describe in detail the definition, design and settings of the proposed experiment, following the guidelines by Wohlin et al. [34] on how to document and report empirical studies in software engineering. Table 1 summarizes the main elements of the experiment. For replication purposes, the experimental package is available on the web4 and we are currently working towards making the tool available to the research community.

The goal of the study is to analyze two different tool-supported prioritization methods, CBRank and AHP, with the purpose of evaluating their time-consumption, ease of use and accuracy. Moreover, we are also interested to understand which method is perceived as “overall the best” and whether the ranks produced by the two methods are similar. Tools are necessary to execute this kind of experiment; without them it will be impossible to conduct the comparative study. Moreover, SCORE and JAHP are strict implementations of the prioritization process of the corresponding methods and offer to the user graphical user interfaces with homogeneous layouts.

In detail the research questions of our controlled experiment are the following:

**RQ1**: Which method between CBRank and AHP is less time-consuming in performing the whole prioritization task?

**RQ2**: Is it easier to use CBRank or AHP?

**RQ3**: Which method produces the more accurate ranking?

**RQ4**: Which method is overall the best?

**RQ5**: Ranks produced by CBRank and AHP are similar?

The experiment is run with 23 participants, more precisely with 15 Ph.D students and 8 junior researchers with experience in industrial projects. The object is unique and consists of a selected set of 20 textual requirements of the Compilation Compiler Advisor (CoCoA) project4, a real web-based application supporting the creation of musical compilations. The perspective is that of the decision maker who evaluates the possibility of adopting CBRank or AHP in her/his organization.

5.1. Hypotheses

In the following, we call prioritization task the prioritization process performed by a subject using a specific tool-supported method on the selected set of requirements.

The goal of the experiment is to investigate the following null hypotheses:

- **H01** The average time to conclude a prioritization task5 is equal for CBRank and AHP.
- **H02** The ease of use is equal for CBRank and AHP.
- **H03** The accuracy is equal for CBRank and AHP.
- **H04** The “overall best” method does not exist.

When the null hypothesis can be rejected with relatively high confidence, it is possible to formulate an alternative hypothesis:

- **H11** The average time to conclude a prioritization task is not equal for CBRank and AHP.
- **H12** The ease of use is not equal for CBRank and AHP.
- **H13** The accuracy is not equal for CBRank and AHP.
- **H14** The “overall best” method exists.

5.2. Variables

The only one independent variable is the method used with two treatments for the main factor: CBRank and AHP. As dependent variables, similarly to [18], we consider: time-consumption, ease of use, accuracy and a measure that in some sense summarizes the firsts three as perceived by the participants, that is “overall the best”.

The average time to conclude a prioritization task is captured by subjects noting down their start and stop time for each prioritization task (declared time). Moreover, for a sample of them (17/23) the time is computed in automatic way, directly by the prioritization tools (actual time). It is worth noticing that the computation

---

3 See http://sra.itc.it/people/susi/material_req_prior_exp.zip.

4 http://cocoa.itc.it:8080/cocoakaradar/.

5 That is to perform processes as described in Fig. 1.
Table 1 Overview of the design of the experiment.

<table>
<thead>
<tr>
<th>Goal</th>
<th>Comparing two tool-supported requirements prioritization methods, CBRank and AHP, for the purpose of gaining increased understanding with respect to their time-consumption, ease of use and accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Perspective</td>
<td>From the point of view of the decision maker</td>
</tr>
<tr>
<td>Context</td>
<td>The experiment is run using 15 Ph.D students and 8 junior researchers as subjects. The object is unique and consist of 20 textual requirements of the CoCoA project</td>
</tr>
<tr>
<td>Null hypotheses</td>
<td>The time-consumption, ease of use and accuracy are equals for CBRank and AHP. The “overall best” method does not exist</td>
</tr>
<tr>
<td>Main factor</td>
<td>Prioritization methods used: CBRank and AHP</td>
</tr>
<tr>
<td>Other factors</td>
<td>Position (Ph.D students vs. Researchers), Industrial experience as Analyst, Ability with Requirements, Session and Method order</td>
</tr>
<tr>
<td>Dependent variables</td>
<td>Time-consumption, ease of use, accuracy, “overall the best”, r_acc, r_easy and r_best</td>
</tr>
</tbody>
</table>

To minimize the effect of order, in which the subjects apply the treatments, the order is assigned randomly to each subject. We choose this design to have the possibility to measure not only the expected accuracy (post-test 1) but also the perceived accuracy (post-test 2).

To study the effect of the other factors (Position, Experience, Ability, Session and Tool order) on time-consumption, ease of use and accuracy we add three questions to the post-test 1:

- which relation between CBRank and AHP (among \( \geq \), >, =, <, \(<\)), would you choose, with respect to “easier to use”?  
- which relation between CBRank and AHP (among \( \geq \), >, =, <, \(<\)), would you choose, with respect to “accuracy”?  
- which relation between CBRank and AHP (among \( \geq \), >, =, <, \(<\)), would you choose, with respect to “better to use”?  

For each of them, subjects could choose among: CBRank \( \equiv \) AHP, CBRank > AHP, CBRank = AHP, CBRank < AHP and CBRank \( < \) AHP with the obvious meaning. We coded the answers with integers ranging from 1 to 5 (five points Likert scale [24]), a common choice in this kind of studies. In this way we added to the experiment three dependent variables \((r_{acc}, r_{easy} \text{ and } r_{best})\) that measure numerically the relations “easier to use”, “accuracy” and “better to use” between the two methods.

5.3. Experiment design

The design we use is the paired comparison design (shown in Table 2), a particular kind of one factor with two treatments [34]. The same design is called “randomized paired comparison design: two alternatives on one experimental unit” in [12]. In this design, each team uses both treatments \((i.e., \text{CBRank and AHP})\) on the same object, i.e., the set of requirements taken from the CoCoA project.

Table 2 Design used.

<table>
<thead>
<tr>
<th>Group</th>
<th>Task 1</th>
<th>Task 2</th>
</tr>
</thead>
<tbody>
<tr>
<td>G1</td>
<td>CBRank</td>
<td>AHP</td>
</tr>
<tr>
<td>G2</td>
<td>AHP</td>
<td>CBRank</td>
</tr>
</tbody>
</table>

The 23 subjects participating in the study are distributed in the following way: 15 are Ph.D students of the University of Trento (Italy) and 8 are Junior Researchers of FBK-IRST with experience in industrial projects. The “Fondazione Bruno Kessler”6 (FBK-IRST) is a research center funded by the Autonomous Province of Trento that conducts Information Technology research in different areas, such as Software Engineering, Service Oriented Applications, Human Language Technologies.

The Junior Researchers involved in the experiment have a Ph.D and currently work as post-doc at FBK-IRST. They collaborate, as Ph.D students, with Senior Researchers on several (industrial and research) projects. All the participants have similar background even if Junior Researchers have more experience that Ph.D students. Researchers and Ph.D students previously attended several programming and software engineering university courses. They have a good knowledge about requirements, programming languages and software engineering in general. We think that the subjects of our study can be considered not far from industrial developers/analysts.

For practical reasons (not all the 23 participants were available at the same date), we run the experiment in two separate dates: 17 subjects participate to the first run (working session 1), 6 people to the second one (working session 2). In each session, subjects are required to perform two prioritization tasks, one using JAHP and the other using SCORE.

5.5. Object

The object of the study is a subset of the user requirements of the CoCoA project. CoCoA is a real Web application that delivers a personalized service for audio compilation. In particular, CoCoA maintains a songs repository that can be used to specify personal compilations. The user can create and modify a compilation (also starting from existing compilations created by other users with
similar music preferences) and download songs and compilations from the CoCoA repository.

From the CoCoA documentation we extracted a subset of 20 high level and rather independent requirements that have been selected taking into account the fact that they have to be clear enough for subjects that are not experts in designing recommendation systems like CoCoA.

Requirements are represented as simple textual descriptions, such as:

- **Save a user defined compilation.**
- **Search a song in the songs repository, by title.**
- **Visualize the recording frequency of a song.**

The prioritization task we propose to the experiment’s participants does not require to take into account dependencies among requirements. Participants are asked to prioritize requirements using the value for the user criterion. That is, the subject, playing the role of CoCoA’s user, has to judge how important and valuable he/she consider a given requirement. Other factors such as: risk, time, and cost have not been considered (the same is true in [18]).

### 5.6. Experiment execution

Participants, before the experiment, attended a short presentation aimed at giving an introduction on: the requirements prioritization problem and on available methods, CBRank and AHP and the CoCoA project. Examples of CoCoA requirements were explained in order to clarify them. A short training on the tools SCORE and JAHP, used on a small set of requirements, was performed. After that participants were asked to compile a pre-questionnaire.

The pre-questionnaire is composed of two parts. The first is used to obtain general information about participants (position, nationalities, working experience, etc.) and to assess the ability level and experience of each involved subject. In particular, this part is useful to divide participants in classes (Researchers vs. Ph.D students, experienced vs. no experienced, etc.). The second part of the pre-questionnaire (see Table 3, left) is instead used: (i) to capture the knowledge of participants about CoCoA requirements, and (ii) to capture the subjects’ knowledge about requirements in general and about specific prioritization methods (CBRank and AHP).

The experiment took place in a laboratory room equipped with computers. A computer, with a browser to access SCORE and JAHP, has been provided to each participant. Participants executed the two prioritization tasks for the 20 CoCoA requirements, applying the two methods sequentially (the order was assigned randomly to each subject as specified in the design). The experiment took approximately 3 h, including introductory presentation and training on the tools, as well as the short break of 5 min between the two prioritization tasks. After the experiment, participants compiled the first post-test (post-test 1). This post-test was useful for three reasons:

- To compare the ease of use of CBRank and AHP.
- To compare the expected accuracy of the two different methods.
- To understand which is the tool preferred by participants (i.e., the overall best tool: easiest to use, most accurate, most scalable, most fast, etc.).

Post-test 1 is composed of questions with multiple-choice answers and free answers in which the subjects had the possibility to discuss the previous set of questions. In addition, an hour later, a second post-test (post-test 2) was given to measure the perceived accuracy. The same was repeated one week later and confirmed the results.

### 5.7. Pilot experiment

A pilot experiment was performed before the main study: (i) to evaluate the design, and (ii) to compute the time necessary to execute the prioritization processes. Two colleagues, trained with an half-hour seminar, participated and executed the tasks using SCORE and JAHP. Approximately, in one hour they completed the prioritization. At the end, they filled the questionnaires. The gathered data were used only for tuning the experimental material. After this pilot experiment, it was concluded that the experiment was well suited for Ph.D students and junior researchers, the level of difficulty was medium and the total time of 3 h sufficient. Following the suggestions of our colleagues, minor changes at the questionnaires and in the way to define and measure the dependent variables used in the study were made.

### 6. Experimental results

This section summarizes the main results obtained from the experimentation we conducted. Results of statistical tests are intended as usual to be significant for a significance level of 95%. The analysis was performed with Microsoft Excel and R.7

#### 6.1. Pre-questionnaire

From the analysis of the pre-questionnaire we obtained the following results: 52% of the subjects had a full-time industrial experience (programmer or analyst), 30% had a previous part-time experience and only 17% had no working experience. In particular, 17% worked full-time as analyst in the industry, 13% worked part-time and 70% had no working experience as Analyst. At the question *What is your experience in the field of Requirements engineering?*, 4% answer very low, 26% low, 39% medium, 30% high and 0% very high. These data confirm that subjects have had, on average, software development experiences and that they can be considered not far from industrial developers/analysts.

In the second part of the pre-questionnaire (the part related to subjects’ knowledge about CoCoA, CBRank and AHP) we used for the answers a five points Likert scale coded with integers (levels of agreement) ranging from 1 (strongly agree) to 5 (strongly disagree). As Table 3 (right) shows, subjects affirm to have understood the CoCoA requirements (median = 2), to know well the “audio compilation” domain (median = 2), to have understood how to use the tools SCORE and JAHP (median = 2) and to have understood the tasks required (median = 1). These data confirm that subjects believe to be experienced with software projects activities in general and requirements engineering in particular. Moreover,

---

7. [http://www.r-project.org/](http://www.r-project.org/)
the subjects are convinced to have understood what they have to do to perform the required tasks.

6.2. RQ1: Which method between CBRank and AHP is less time-consuming in performing the whole prioritization task?

Some insights can be obtained by looking at the descriptive statistics in Table 4 and at the boxplots in Fig. 4, that compare the declared time to conclude the prioritization task using AHP and CBRank, resulting from start/end times recorded by the subjects. It is clear that the time to conclude the task is larger with AHP than with CBRank. As Table 4 shows, the difference in time (sample mean) between the two methods is 27.87 min. This can also be seen in Fig. 4, where the median values are higher for AHP than for CBRank. Additionally, the box-plot indicates that the measures of the time-consumption of AHP, obtained from subjects annotations, are more dispersed (SD_{AHP} = 9.78 vs. SD_{CBRank} = 4.4) than for CBRank.

Due to the nature of the variables and the limited number of data points we decided to apply a non-parametric statistical test. In particular we selected the Wilcoxon test [34] that is very robust and sensitive. The difference in time is significant as the paired two-tailed Wilcoxon test results in a \( p \)-value < 0.01. Thus, the first null hypothesis is rejected.

The statistical significance alone does not tell anything about the practical impact of the treatment: it is important to measure the effect size of the main factor over the dependent variables, i.e., the strength of the relationship between the main factor treatment and the dependent variables. We used the Cohen standardized difference between two groups [8], defined as the difference between the means (\( M_1 \) and \( M_2 \)), divided by the pooled standard deviation (\( \sigma \)) of both groups \( d = (M_1 - M_2)/\sigma \). In literature, the effect size is considered small for \( d = 0.2 \), medium for \( d = 0.5 \) and large for \( d = 0.8 \). We observed a positive (i.e., AHP is more time-consuming than CBRank) and very large effect size (\( d = 3.67 \)).

Considering the actual time, computed by prioritization tools, the results does not change significantly. The difference in time between the two methods is still significant with a Wilcoxon test (\( p \)-value < 0.01). The difference between mean and median values is only slightly higher than before (now: mean_{AHP} = 39.57, mean_{CBRank} = 29, median_{AHP} - median_{CBRank} = 30, see Table 4 for a comparison).

6.3. RQ2: Is it easier to use CBRank or AHP?

Among the 23 subjects, 22 found CBRank easier to use than AHP, nobody found them equally easy to use and 1 stated that AHP was easier to use. Hence, 95.65% of the participants found CBRank easier to use. This was tested with a two-sided prop-test by comparing the number of answers in favor of CBRank to the total number of answers. It turned out that there is a statistically significant difference, as \( p \)-value < 0.01. We obtained the same results using the dependent variable \( r_{easy} \): median = 2 (i.e., CBRank > AHP). Thus, the second null hypothesis is rejected.

6.4. RQ3: Which method produces the more accurate ranking?

After each working session, the participants performed the post-test 1 that captured which method the subjects were expecting to be the most accurate. Among the 23 subjects, 7 (30.43%) declared CBRank more accurate than AHP, nobody declared them equally accurate and 16 (69.57%) stated that AHP was more accurate. This is, however, not statistically significant given that the \( p \)-value computed with the prop-test is 0.09. A similar result was obtained using \( r_{accurate} \): median = 4 (i.e., CBRank < AHP).

In a second post-test, aimed at measuring the perceived accuracy, subjects received two lists of ordered requirements without information on which method had produced them. We asked them to mark the list that better fitted with their preferences. The result was totally in favor of AHP. All the subjects chose the list corresponding to AHP, i.e., 100% of the subjects found the result of AHP most accurate than CBRank. This is, clearly, statistically significant with the prop-test (\( p \)-value < 0.01).

Overall, we can not reject the third null hypothesis due to the perceived accuracy but clearly the direction is in favor of AHP.

6.5. RQ4: Which method is overall the best?

As a final question in the post-test 1 we asked the subjects to specify the method they consider the “overall best”. 16 (69.57%) of them indicated CBRank, 4 (17.39%) AHP and 3 (13.04%) considered the two methods similar. This result is statistically significant using a two-sided prop-test (\( p \)-value = 0.01) and comparing the number of answers in favor of CBRank to the number of answers in favor of AHP. A similar result was obtained using \( r_{best} \): median = 2 (i.e., CBRank > AHP). Thus, \( H_0 \) can be rejected.

6.6. RQ5: Ranks produced by CBRank and by AHP are similar?

In order to have a measure of the difference between the ranking obtained by CBRank and the ranking produced by AHP for each subject, we use the agreement measure that has been introduced in Section 4. Fig. 5 shows the agreement of one of the subjects in our experiment: the x-axis represents the positions in the ranking and the y-axis indicates the value of agreement of the rankings computed by the two methods. The result indicates that there is a very

---

Table 4

<table>
<thead>
<tr>
<th>Method</th>
<th>Median</th>
<th>Mean</th>
<th>Standard deviation</th>
</tr>
</thead>
<tbody>
<tr>
<td>AHP</td>
<td>36</td>
<td>38.65</td>
<td>9.78</td>
</tr>
<tr>
<td>CBRank</td>
<td>10</td>
<td>10.78</td>
<td>4.4</td>
</tr>
<tr>
<td>Difference (AHP−CBRank)</td>
<td>26</td>
<td>27.87</td>
<td>5.38</td>
</tr>
</tbody>
</table>

---

Fig. 4. Boxplot of the time-consumption for each prioritization task, derived from times recording by the subjects.

\footnote{R command used: prop.test (22, 23, alternative = “two.sided”, \( p \) = 0.5).}
good agreement between the two ranks: in the diagram the two methods put the same requirement in the first position, resulting in a value 1 for the agreement, and have an agreement of 0.8 in the sixth position. This is similar for all the subjects. In Fig. 6 is shown the “mean agreement” measure that is the agreement, resulting from considering the mean of the agreement measures for all the subjects in the experiment. For example, at the 7th position of the ranks we have a very high mean agreement, close to 0.7, that confirms the results from the analysis of the different ranks. So, notwithstanding the good results for the accuracy of AHP,
CBRank demonstrated to be very close to the performances of the other method.

6.7. Effect of other factors

This section analyzes the effect of the dependent variables (i.e., time-consumption, $r_{\text{acc}}$ and $r_{\text{easy}}$, as defined in Section 5.2), of other factors, namely:

- Subjects’ Position, i.e., Ph.D students vs. researchers.
- Subjects’ Experience, i.e., industrial experience as analyst vs. no experience.
- Subjects’ Ability, i.e., high vs. low requirements engineering ability.
- Working session, i.e., first vs. second (to understand whether there are differences between sessions).
- Method order, i.e., CBRank first group vs. AHP first group (it represents the order of use of the two methods).

6.7.1. Effect of other factors on time-consumption

The analysis was performed by using a two-way analysis of variance (ANOVA). On the overall data set, we found no significant effect of the Position on the time ($p$-value = 0.2) nor any interaction with the main factor ($p$-value = 0.8). Also the Experience and Ability factors did not produce any significant effect on the time (respectively, $p$-values 0.50 and 0.28) nor any interaction with the main factor ($p$-values 0.93 and 0.93).

The fourth factor we considered was the Session. In this case, as shown in Table 5, the effect is significant ($p$-value = 0.03). Analyzing the data we discovered that the group belonging to the second session used in total more time than the first group. Precisely 29.1 min (second session group)—the average of the times of the group—vs. 23.5 (first session group). We found no interaction effect with the main factor.

Also for the fifth factor, i.e. the Method order, the effect is significant ($p$-value = 0.01). On the average, subjects starting with CBRank used in total more time (27.37 min) than subjects starting with AHP (21.8 min). A possible explanation could have reference to the “fatigue effect”. The task with AHP is more demanding than that one with CBRank and when executed as second became even more time-consuming. We found no interaction effect of the Method order with the main factor ($p$-value = 0.88).

6.7.2. Effect of other factors on accuracy ($r_{\text{acc}}$)

Since Position, Experience, Ability, Session and Method order may have an influence on how the subjects chosen the most accurate method (expected accuracy), we analyzed their effects. For this purpose we used the non-parametric Mann–Whitney test to verify whether the median values of $r_{\text{acc}}$ were different between the following groups: (1) Ph.D students and researchers, (2) analysts and no experienced subjects, (3) high and low ability subjects, (4) session 1 and session 2 groups, and (5) CBRank first and AHP first groups. The results are presented in Table 6 (first row). We can observe that none factors has a significant effect on $r_{\text{acc}}$ (i.e., accuracy) and that the median is always 4 for each group. It is important to highlight that the tests could be non-significant, because of the limited number of subjects.

6.7.3. Effect of other factors on ease of use ($r_{\text{easy}}$)

As in the above case we analyzed the effects of Position, Experience, Ability, Session and Method order on how the subjects chosen the simplest method. We can observe (see row 2 of Table 6) that only the Method order has a significant impact on $r_{\text{easy}}$. In particular we can observe that subjects using AHP as first tool judged CBRank more easy to use ($r_{\text{easy}} = 1$) than subjects using CBRank as first tool ($r_{\text{easy}} = 2$). It is important to highlight that the tests could be non-significant, because of the limited number of subjects.

6.8. Threats to validity

This Section discusses the threats to validity that can affect our results: internal, construct, conclusion and external validity threats [34].

Internal validity threats concern external factors that may affect the outcome of our study. They can be due to the fatigue effect and to learning effect experienced by subjects between the two tasks. The subjects could become fatigued during the experiment, which may affect the concentration. To limit this threat we keep the number of requirements low in order to conclude the experiment in approximately 3 h. Moreover, we scheduled a mandatory break between the two tasks. However, a possible influence of fatigue is identifiable, given that the effect of Method order on time-consumption is significant ($p$-value = 0.01) as revealed by the ANOVA test. The “learning effect threat” can take place when subjects get practice during the experiment and unconsciously get an opinion on the context using the first method, which will affect the result for the second method. To understand if the results of our experiment have been influenced or not from the order in which the two methods have been used we investigated the method order effect on $r_{\text{acc}}$ and $r_{\text{easy}}$. A set of significance test was conducted; only $r_{\text{easy}}$ showed a significant difference but without changing the opinion of the participants (in fact CBRank is constantly judged as the easiest, $r_{\text{easy}} = 2$ for the CBRank first group and $r_{\text{easy}} = 1$ for the AHP first group). We think that this finding, together with the fact that the time exploited is increased and not decreased during the second prioritization task, validates the fact that the experiment has not suffered from any order effects. To avoid social threats due to evaluation apprehension, Ph.D students and junior researchers were not evaluated in any way on their performance in the experiment. Finally, subject were not aware of the experimental hypothesis even if it was clear to them our intention to compare AHP with CBRank.

Table 5
Two-way ANOVA of method and session.

|                    | Df | Sum Sq | F-value | Pr(>|F|) |
|--------------------|----|--------|---------|---------|
| Method             | 1  | 8932.2 | 172.61  | 2.220e-16 |
| Session            | 1  | 245.4  | 4.74    | 0.03    |
| Method:session     | 1  | 1123.3 | 2.17    | 0.27    |
| Residuals          | 42 | 2173.4 |         |         |

Table 6
Effect of other factors on accuracy and ease-of-use measurements.

<table>
<thead>
<tr>
<th></th>
<th>Ph.D</th>
<th>Researchers</th>
<th>p</th>
<th>Yes</th>
<th>No</th>
<th>p</th>
<th>High</th>
<th>Low</th>
<th>p</th>
<th>1</th>
<th>2</th>
<th>p</th>
<th>CBRank</th>
<th>AHP</th>
<th>p</th>
</tr>
</thead>
<tbody>
<tr>
<td>$r_{\text{acc}}$</td>
<td>4</td>
<td>4</td>
<td>0.85</td>
<td>4</td>
<td>4</td>
<td>1</td>
<td>4</td>
<td>4</td>
<td>0.25</td>
<td>4</td>
<td>4</td>
<td>0.59</td>
<td>4</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>$r_{\text{easy}}$</td>
<td>2</td>
<td>1</td>
<td>0.5</td>
<td>1</td>
<td>2</td>
<td>0.23</td>
<td>2</td>
<td>2</td>
<td>0.59</td>
<td>2</td>
<td>2</td>
<td>0.89</td>
<td>2</td>
<td>1</td>
<td>0.02</td>
</tr>
</tbody>
</table>
Construct validity threats concern the relationship between theory and observation. Time was measured by means of proper time sheets and validated qualitatively by researchers who were present during the experiment (experiment conductors checked that the forms were correctly filled). Although this may not be very accurate, this is a widely adopted way of measuring performance. Moreover, for a sample of the participants the time was computed in automatic way (actual time), directly by the prioritization tools, and all the analysis repeated on this sample obtained similar results. Furthermore, we compared declared time and actual time for each subject obtaining a small difference between them. The ideal target ranking is not known a priori, in general, and depend-ent variables such as ease of use, accuracy and overall the best subjective variables, that is their measurement rests upon how they are perceived by the subjects and may be further biased by the subjects previous experience and knowledge about a particular problem domain. These aspects makes hard to measure the ranking accuracy, as discussed also in previous work in which definitions and measures for the accuracy are given [6,18]. For us an accurate prioritization approach is one that produces an ordering that reflects the decision maker opinion, as in [18]. Moreover, we measured accuracy in two ways, following [6]. To gather the opinions of participants about CBRank and AHP we used a questionnaire designed using standard ways and scales. Values of dependent variables were derived from the analysis of these answers.

Conclusion validity concerns the relationship between the treatment and the outcome. Proper tests were performed to statistically reject the null hypothesis. In particular the chosen experiment design permitted the use of paired tests for the time-consumption variable. In case of presence of differences, although not significant, this was explicitly mentioned. Non-parametric tests were used in place of parametric tests where the conditions necessary to use parametric tests do not hold. Questionnaires were designed using commonly accepted structure and value scales for the answers. However, a threat that may affect statistical power is due to the limited sample of subjects of the experiment (only 23 subjects were available). This is particularly true when we have measured the effect of other factors on accuracy (r_acc) and on ease of use (r_easy), because in this case we have compared subsets of subjects (e.g., Ph.D students and researchers) further decreasing the samples.

External validity threats concern the generalization of the findings. They have to be taken into account when experimenting with students and junior researchers. Since the selected subjects represent a population: (i) with good knowledge on requirements, (ii) industrial experience, and (iii) trained on prioritization methods we expect similar results for industrial developers/analysts [30]. Threats to external validity are also related to the requirements we used as experimental object. While the system itself represents a real world application, requirements and related questions are forcefully simple, thus they may be deemed as realistic but not real. On the other hand, their size and complexity were designed to be proportional to the time available for the experiment. This makes the context quite realistic, despite only further studies with different types of domains, different requirements (other software applications or other level of abstraction of the requirements) and subjects can confirm the obtained results. The small number of requirements (20) used in the experiment to limit the “fatigue effect”, decreases the possibility to generalize to situations where a larger set of requirements (e.g., >50) are prioritized. We do not think that the results could change by using different implementations of CBRank and AHP. All the existing implementations of CBRank and AHP are very similar to SCORE and JAHP. Moreover, “the look and feel” of the GUIs of JAHP and SCORE has been uniformed as much as possible. This allowed us to focus on the methods differences rather than on the tools differences. Furthermore, the elaboration of the time of the algorithms and GUIs usage is negligible with respect to the time required by the human decision-making activities in the prioritization process. So the computed time-consumption refers essentially to the subjects’ decision-making time.

6.9. Discussion

The main results obtained in this experiment are summarized in Table 7.

On the basis of the data from post-test 1 the main results of the experiment are that for the variables “time-consumption” and “ease of use” CBRank overcomes AHP and nothing can be determined for the variable “accuracy” (p-value = 0.09 for expected accuracy). The results of the post-test 2, where the subjects compared the ranks produced by the two methods, clearly state that the rank produced by AHP better fits user preferences (p-value < 0.01 for perceived accuracy) and thus we can say that the highest accuracy is achieved with AHP.

Considering the time-consumption, the difference between the two methods appears to depend mainly on the number of comparisons requested to the subjects, 190 for AHP, that performs an exhaustive process, and 55 pairs for CBRank, that exploits the machine learning techniques to reduce the number of comparisons. Moreover, also the different range of values for expressing a preference exploited by the two methods could have contributed to this difference in time-consumption. From one side, the fine grained evaluation that subjects have to perform in AHP (i.e., choosing an integer belonging to the scale [1,9]) could result in an increasing cognitive effort, to the other, a strict (binary) choice, as in CBRank, could result in a more difficult selection.

In the same line is the results for the variable “ease of use”. Both the number of comparisons and the need to specify a value in a range influenced the subjects perception on the methods ease of use.

These observations have been confirmed by a set of free answer questions we have included in the post-test 1. In particular the subjects recognized the need for a major effort in using the values of the range for AHP in a coherent way, but, at the same time, some of them also appreciated the possibility to be more precise in the evaluation, even if this requires a greater attention.

Concerning the expected accuracy of the methods, even if the result is not statistically significant, the majority of the subjects chose AHP as the most accurate method. This has been discussed in the free answers in post-test 1 with the observations related to the exhaustive nature of the AHP process. However, the post-test 2 gives a clear result in favor of the accuracy of AHP.

Regarding a possible bias due to the fact that the analysed methods exploit two different scales (ordinal in CBRank vs. rational in AHP), findings obtained in previous studies shall be considered. The difference between using an ordinal scale and a rational scale has been investigated in an off-line experiment, showing that for sets of about 20 requirements and more, the effect of using one scale or the other becomes indifferent [9]. Concerning the influence on the results by CBRank with machine learning support for selecting pairs to be prioritized, we shall mention that experiments have

<table>
<thead>
<tr>
<th>Hypothesis</th>
<th>Rejected?</th>
<th>Direction</th>
<th>p-Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>H0:1: time-consumption is equal</td>
<td>Yes</td>
<td>CBRank</td>
<td>&lt;0.01</td>
</tr>
<tr>
<td>H0:2: the ease to use is equal</td>
<td>Yes</td>
<td>CBRank</td>
<td>&lt;0.01</td>
</tr>
<tr>
<td>H0:3: the accuracy is equal</td>
<td>No</td>
<td>AHP</td>
<td>0.05</td>
</tr>
<tr>
<td>expected accuracy is equal</td>
<td>No</td>
<td>AHP</td>
<td>&lt;0.01</td>
</tr>
<tr>
<td>perceived accuracy is equal</td>
<td>Yes</td>
<td>AHP</td>
<td>&lt;0.01</td>
</tr>
<tr>
<td>H0:4: The overall best tool does not exist</td>
<td>Yes</td>
<td>CBRank</td>
<td>0.01</td>
</tr>
</tbody>
</table>

Table 7

Summary (p-values under the threshold are in bold).

A. Perini et al. / Information and Software Technology 51 (2009) 1021–1032
been made to compare the behavior of this method, exploiting the machine learning step for the choice of the pairs to be evaluated, against a version of the same method with a random choice of the pairs. The performance of CBRank with the learning process was better than that of the method using random selection.

The analysis of the specific results for the ease of use, time-consumption, accuracy, “overall the best” and level of agreement gives useful information to guide the selection of the more appropriate method, for a given project.

AHP seems to be more suitable when the accuracy is one of the main issues and the number of requirements to be prioritized is limited. This could be the case of long prioritization processes where a first classification of the requirements have been performed a priori. On the other side, CBRank could be exploited when it is crucial to find a trade-off between the accuracy of the result and the time spent in the prioritization process. For example, in an agile software development process or in the case the number of requirements to be prioritized is large.

7. Conclusions

This paper described an empirical study aimed at comparing two tool-supported requirements prioritization methods, AHP and CBRank.

We focused mainly on three measures: the ease of use, the time-consumption and the accuracy. The experiment has been conducted with 23 experienced subjects on a set of 20 requirements from a real project. Results showed that for the first two characteristics CBRank overcomes AHP, while for the accuracy AHP performs better than CBRank, even if the resulting ranks from the two methods are very similar. The majority of the users found CBRank the “overall best” method.

As a final observation, the results indicated that AHP should be preferred to CBRank in prioritization problems for which the ordering accuracy is a main issue and the number of requirements is around 10, while CBRank should be used when it is crucial to find a trade-off between the accuracy of the result and the time spent in the prioritization process, as in the case of large set of requirements.

We are currently exploring the possibility to evaluate the methods for multi-criteria prioritization, e.g. cost, value.

More generally, as future work it would be of interest to perform a controlled experiment in a real industrial setting where different stakeholder views play a crucial role in practice, motivating the need to integrate negotiation mechanisms into the requirements prioritization process.

Acknowledgements

We would like to thank the students and researchers that participated to the experiment, and Cinzia Bazzanella for her support. Thanks also to the anonymous reviewers for their valuable suggestions. This work has been partially funded by the STAMPS project.

References


