



Repositorio Institucional de la Universidad Autónoma de Madrid

<https://repositorio.uam.es>

Esta es la **versión de autor** del artículo publicado en:

This is an **author produced version** of a paper published in:

INFORMATION AND SOFTWARE TECHNOLOGY 97 (2018): 163-175

DOI: <https://doi.org/10.1016/j.infsof.2018.01.010>

Copyright: © 2018 Elsevier B.V.

El acceso a la versión del editor puede requerir la suscripción del recurso

Access to the published version may require subscription

Systematic Guidance on Usability Methods in User-Centered Software Development

Luis Cayola and José A. Macías

Escuela Politécnica Superior. Universidad Autónoma de Madrid

Tomás y Valiente 11, 28049 Madrid. Spain

luis.cayola@estudiante.uam.es, j.macias@uam.es

Abstract

Context: In order to ensure usability, it is necessary to schedule activities and methods to be applied throughout different stages of the development process. There exists a substantial number of usability methods to be applied in user-centered software development. However, the application of each usability method largely depends on specific constraints that should be closely considered. Even so, these constraints are not always known beforehand, remaining unidentified or under uncertainty at early stages of the project.

Objective: This paper presents an approach to automatically recommend 43 usability methods depending on the project's stage and constraints. Our approach deals with uncertainty to recommend usability methods regardless of the completeness of the information available, which makes it suitable for enhancing initial scheduling. Besides, a supporting tool intended to schedule and guide on usability methods is presented in order to systematize the recommendation mechanism.

Method: To validate our approach, we present two application scenarios demonstrating the suitability of the mechanism, including also an expert analysis to observe the recommendation appropriateness in terms of recommendation gap. Also, a user testing was accomplished to evaluate the usability of the approach with key users.

Results: A low recommendation gap was observed ($< 2.5\%$) and, according to the results obtained in the user testing, high percentage values for usefulness (82.38%) and satisfaction (87.89%) were obtained. The user evaluation also reported high values concerning other dimensions such as ease of use (89.00%) and ease of learning (92.38%).

Conclusions: Results obtained helped answer main research questions, demonstrating that it is possible to create a mechanism to recommend usability methods according to a software project's constraints, even under uncertainty, and also affirm that it is possible to systemize the recommendations with a scheduling tool being satisfactory for key stakeholders, denoting acceptable levels of recommendation appropriateness, usefulness, and overall usability.

Keywords

Usability, User-Centered Development, Method Recommendation, User-Centered Project Management, Software Quality, CASE Tool.

1. Introduction

Nowadays, usability and user-centered design have become essential issues in order to guarantee the quality and success of a software project. A software failing to include usability aspects may lead to a decrement of productivity and low acceptance from final users. In fact, the number of studies about these concerns has recently increased, as there are real difficulties on systematically meeting the users' usability expectations [1, 2, 3].

Although the development of computer applications has evolved to tackle usability concerns, most of the existing efforts are mainly oriented to provide acceptable usability measures at the evaluation stage, focusing on improving efficiency, effectiveness and user satisfaction in a summative way, thus overlooking a formative vision. In general, usability assurance should be principally arranged and considered from the early stages of a software project, and hence it should be scheduled accordingly, proposing specific activities and methods according to the project's characteristics and constraints, which is essential to guarantee the usability in every particular software to develop [4].

In order to provide with a reference framework, there exists the standard ISO/TR 16982 [5], which provides recommendation for 12 usability methods to be applied in software projects depending on a set of constraints. This helps project managers decide whether a usability method can be applied in a certain stage of the development process. However, this standard is specifically oriented to project managers, which makes it unsuitable for real software projects today involving agile and multidisciplinary development teams, where members take on different usability tasks. In addition, the standard is restricted to a small number of usability methods, also failing to provide detailed explanation about the application of the methods in each stage of the project. Additionally, the standard requires knowing all the constraints at the very beginning of the project, which can be difficult in practice as some parameters are initially unknown or difficult to be estimated early.

1.1 Research Questions

Based on the drawbacks previously described, we define the following general questions to conduct our research:

- [RQ1] *Is it possible to create a mechanism to provide appropriate recommendation on usability methods according to specific project constraints, even under uncertainty?*
- [RQ2] *Is it possible to systematize such recommendation mechanism by means of a supporting tool to schedule and provide additional information about each usability method, being satisfactory for key stakeholders?*

1.2 Contribution

To overcome the aforementioned difficulties and give an answer to the proposed research questions, we present a mechanism improving the original ISO/TR 16982 framework. Our approach is based on providing recommendations for a much more extensive number of usability methods (a total of 43) to be applied in different stages of a software project, taking into account the suitability degree of each method according to the characteristics of the project. Additionally, our approach deals with uncertainty by providing recommendation even when a certain constraint is unknown, empowering project scheduling at early stages.

Furthermore, the proposed mechanism has been systematized through the implementation of a tool called STRUM (*Scheduling Tool for Recommending Usability Methods*). This tool enables development team to schedule software projects in terms of the usability methods to apply, and it allows team members to include comments about the methods applied for tracking usability along the project. This tool has been conceived to be easy to use and learn, so key stakeholders, and not only project managers, can use it easily in multidisciplinary project teams.

This paper is structured as follows. Section 2 presents the related work, describing current approaches and analyzing their suitability for the problem stated. Section 3 presents a description of the proposal, including the preliminary research, the developed mechanism, the supporting tool's main features, and two application scenarios to validate the approach. Section 4 includes the evaluation of the approach and the analysis of the results obtained. Finally, Section 5 reports on conclusions and future work.

2. Related Work

Currently, there is a lack of existing approaches to systematize the recommendation of usability methods in order to be applied in software project. Most of the related work is principally based on documents and information repositories rather than systematized solutions [4, 5, 6, 7, 8, 9].

To cite a few, approaches such as *Usability Body of Knowledge* [10] aim to gather information about publications, conferences and professional experience coming from usability experts. Detailed explanations about some usability methods, as well as useful definitions, can be found in this approach. In fact, one of the most relevant aspects of this approach is the number of methods described, also including brief and concise definitions. However, the information provided is somewhat heterogeneous. A similar approach is *Usability Net* [11], a European Union founded project that provides access to reference sources and application guidelines on usability methods. There are also other projects such as *Usability.gov* [12], founded by the U.S. Department of Health & Human Services. This approach provides high quality information about several usability methods and advice on application. Nevertheless, the number of described methods is low as well as their findability. Other approaches consist in explanations about usability methods based on practical experience. This is the case for *Nielsen Norman Group* [9] site, which is specialized on user experience and provides information about the eligibility of different usability methods. However, the explanation reported for each method is very limited, and it is difficult to distinguish the suitability of the different methods with respect to the constraints of a specific software project. Other works, such as the ones described in [4, 13], provide an extensive description of usability methods to be included in software engineering activities. These works can be considered as an interesting repository of usability methods that are explained and categorized for integration in different project stages. However, these approaches lack an explicit reference to specific constraints that may limit the application of the provided usability methods according to the project's characteristics.

A more related approach is *Usability Planner* [14], which comprises a web tool that provides recommendation on usability methods driven by risk and cost for specific project stages. This tool allows the user to modify the selected constraints to see how they impact on the recommendations made. However, this approach does not provide significant scheduling facilities based on dates and the current project stage. Also, the stages and constraints utilized

differ from the standard, which makes it difficult to be generalized for a broader application. Besides, the recommendation system does not deal with uncertainty based on unknown project characteristics. In addition, this approach utilizes a star rating system to display recommendation results, being complex to contextualize and providing no specific explanations of the usability methods for further usage and application during the project.

All in all, analyzed approaches have been considered as a useful reference for creating and documenting our approach, thus helping study the way usability methods can be better recommended according to the project's stages and constraints.

3. The Proposal

As explained in previous sections, the aim of this contribution is to systematize the way of obtaining recommendation about usability methods, taking into account the characteristic of a software project, even when there is uncertainty on such characteristics.

To carry out this task, we have based on the standard ISO/TR 16892 to initially determine which usability methods would be more suitable to apply in a certain stage of a project depending on certain constraints. More specifically, the standard describes the most common constraints that may arise during the development of a project and how they affect to the utilization of a specific usability method. To deal with this information, the standard utilizes qualitative values ranging from "recommended" to "not recommended" to rate each usability method. These information is codified in tables, so to know the suitability of a usability method application it is necessary to check the value associated to it according the corresponding project's stage and constraints. As in previous works [14] we have scaled these tables using numeric values, thus creating an algorithm that makes use of these values to determine which method is suitable in each case.

This way, the mechanism that we propose to automatize the methods recommendation is based on the interpretation and statistical treatment of constraint tables, also considering uncertainty values to predict recommendation when some project constraints are undefined or unknown. The idea is that key stakeholders specify the project constraints, defined in ISO/TR 16892, by filling in a questionnaire. Additionally, we have increased the number of usability methods included in the standard from 12 to 43, so we have created a recommendation algorithm based on larger information and considering uncertainty. This way, the algorithm calculates a recommendation percentage for each method depending on the project stage, also providing stakeholders with a ranking of the most important usability methods to apply in each case.

By using the supporting tool STRUM, input and output information can be easily managed by key stakeholders, enabling to change input data on demand to obtain different outputs, also obtaining further information to control the application of each usability method during the project according to temporal states and scheduling.

In the following subsections, we describe the preliminary research carried out to build our recommendation mechanism and the formal concepts behind it. Also, we present an overview of the tool the two application scenarios in order to validate the approach.

3.1 Preliminary Research

In order to systematize the recommendation mechanism, we carried out a preliminary research focused on answering the following initial research questions, which can be considered as part of RQ1:

- [RQ1.1] *What other usability methods can be also considered, apart from those already listed in ISO/TR 16892?*
- [RQ1.2] *How can these additional usability methods be integrated into the constraints framework proposed by ISO/TR 16892?*

To address RQ1.1, we carried out a Systematic Mapping Study [15] to search for other usability methods different from those listed in ISO/TR 16892. To achieve this task, we searched the following digital libraries: ACM DL, IEEEExplore, Springer Link, Scopus and Google Scholar. Also, we accomplished an additional Internet search to find out online resources, as sometimes the utilization of usability methods is better detailed from a professional perspective than from an academic point of view. We utilized the following search string in all cases:

("usability" OR "user-centered design" OR "user-centred design")

AND

("method" OR "technique")

Retrieved papers were manually checked, applying screening criteria to exclude those presenting only introductory or superfluous information, functional evaluation issues, and duplicated or redundant information. The systematic search brought to light interesting works based on previous studies on usability methods [4, 9, 13, 16], which were specifically useful for our research. Also, the systematic search helped identify principal practices and the stage of the project where each method is more suitable to be applied. After the study, we obtained a total of 31 new usability methods to be considered, which helped give an answer to RQ1.1.

Table 1 shows the final list of all usability methods featured (12+31), together with their most representative references, indicating for each method whether it was already listed in ISO/TR 16892 (those 12 methods marked with "X"). In addition, the information found helped document each usability method for our tool, in order to provide the user with useful information about each method usage (see Section 3.3).

Method	Main References	Originally Included in ISO/TR 16892
Affinity Diagrams	[17, 18]	
Automated Evaluation	[5, 19]	X
Card Sorting	[20, 21]	
Cognitive Models	[13, 16, 22]	
Collaborative Design and Evaluation	[5, 23]	X
Competitor Analysis	[16, 24]	
Contextual Inquiry	[25, 26]	
Creativity Methods	[5, 27, 28]	X
Critical Incident Analysis	[5, 29]	X
Design Guidelines	[13, 30]	
Document Based Method	[5, 31]	X

Essential Use Cases	[32, 33]	
Ethnographic Studies	[34, 35]	
Expert Evaluation	[5, 16]	X
Help Structured as Use Cases	[32, 36]	
Heuristic Evaluation	[13, 16, 24, 37]	
HTA (Hierarchical Task Analysis)	[38, 39]	
Impact Analysis	[40, 41, 42]	
Inspections	[16, 43]	
Interface Design Patterns	[44, 45]	
Interface State Chart Diagrams	[46, 37]	
Interviews	[5, 13, 16, 30]	X
JEM (Joint Essential Modeling)	[4, 32]	
Menu Trees	[48, 49]	
Model Based Methods	[5, 50, 51]	X
Navigation Map	[32, 52]	
Paper Prototyping	[16, 24, 53]	
Performance Measurement	[5, 13, 54]	X
Personas	[55, 56]	
Pluralistic Walkthrough	[16, 43]	
Post-Test Information	[32, 57]	
Questionnaires	[5, 13, 16, 30]	X
Scenarios and Storyboard	[16, 24, 58]	
Task Scenarios	[16, 30]	
Thinking Aloud	[5, 13, 16, 59, 60]	X
Usability Specification	[40, 58]	
Usability Test in Laboratory	[54, 61]	
User Feedback	[16, 48]	
User Monitoring	[24, 62]	
User Observation	[5, 16]	X
User Profiles	[30, 63]	
User Roles Map	[32, 36]	
Visual Brainstorming	[64, 65]	

Table 1. List of the 43 usability methods featured, together with their corresponding main references and indicating the 12 original methods listed in ISO/TR 16892.

To tackle RQ1.2, we carried out an expert inspection to rate each usability method according to the 17 project constraints included in the standard ISO/TR 16892. In general, and according to the ISO, the selection of each method is affected by general project constraints mainly related to issues such as the life-cycle stage, the characteristics of both users and tasks to be performed, the product or system itself, specific project conditions such as time and cost, the degree of expertise of the development or evaluation team and so on. These 17 constraints are detailed in Table 2, followed by a brief description of each one.

Project Constraints	Short Description
Time-scale	There is a very tight time-scale in the project
Cost/price control	The cost of a method is important in the project budget
Quality of the product	There is a need for high quality level of the product to be delivered as a dominant requirement
Information/feed-back/diagnosis	There is a need for early information, feedback and diagnosis
Evolving specifications	Specifications of the project are highly evolving

User involvement	User can or cannot be involved/accessed in the project
User disability	Involved users have significant disability to be considered
Task complexity	The level of tasks complexity is high
Error implication	Errors can lead to severe consequences and should be specifically considered
Task novelty	Tasks are completely new to the users
Task spectrum	The range of the task is wide and there are large variations in functionality
Major changes	There are major changes in organization/job/technical
Time and accuracy	There are high levels of time and accuracy constraints for interaction
Adaptation	There is a need for adaptation to an already existing system/product
Simple product	There is a need for limited and simple well-understood product
Customization	There is a high degree of adaptability of the product in terms of customization
Designer abilities	The designer/evaluator has access to extensive ergonomic/human-factors skills/expertise

Table 2. List of the 17 project constraints considered and listed in ISO/TR 16892.

Inspired by the rating criteria detailed in ISO/TR 16892 for the original 12 methods, we used similar punctuation values, but codified in a numerical 6-point Likert scale to facilitate the computation, where 5 indicates highly recommended, 4 indicates recommended, 3 indicates appropriate, 2 indicates neutral, 1 indicates not recommended and 0 indicates not applicable.

This rating provides advice about method usage under specific constraints. For instance, when users cannot be involved in the project, it is highly recommended to utilize methods based on indirect user involvement such as model and document based methods, or expert and automated evaluation, to cite a few. However, methods based on user observation, questionnaires, card sorting, thinking aloud, etc., are not applicable in this case.

Although 12 out of 43 methods were already rated in the ISO document, we asked usability experts to rate all the 43 methods in order to homogenize the results and reduce the criteria bias. In addition, we wanted to improve ISO recommendations including uncertainty, so we also wanted to know the recommendation of usability methods against constraints initially unknown.

This way, we asked to 10 usability experts (7 men and 3 women) to carry out an inspection. The sample included 6 university lecturers specialized in human-computer interaction and software engineering, as well as 4 professionals from IT companies specialized in user experience and usability projects, all recruited from both our institution and partner companies. Experts rated each method appearing in Table 1 according to the ISO constraints appearing in Table 2 for the three situations pursued: when the constraint fulfills, when the constraint does not fulfill, and when the constraint fulfillment is unknown. Experts carried out their evaluation individually, in order to minimize evaluation bias. The process was similar to a heuristic evaluation [66], where a template, including the methods and the constraints, was provided to each evaluator.

Once received, the results were in-depth analyzed. This way, an inter-rater reliability analysis using kappa statistic was performed to determine consistency among experts. More specifically, a Fleiss' kappa for multi-rater was applied to evaluate the agreement with the ratings obtained from the 10 experts. The kappa value comprises a real number between 0 to

1, where 1 means agreement and 0 means disagreement, and a value between 0.81-1.00 implies almost perfect agreement. A *kappa-value* = 0.85 was obtained, showing an acceptable level of agreement among experts.

Additionally, low statistical values were obtained for standard deviation, confidence interval 95% and coefficient of variation, respectively, in all cases: $SD < 0.3$, $CI (95\%) < 0.3$, $C_v < 10\%$. This way, the arithmetic mean of the experts' scores was considered as an accurate indicator to populate the knowledge base.

As for the quality of this initial research, the risk of bias was also considered. On the one hand, issues related to the systematic study affects to publication and selection biases [13]. Admittedly, the positive results problem is always inherent in every systematic study, as well as the risk of selecting a thinner spectrum of papers uncovering the initial objectives pursued by the research question. We have mitigated these risks by considering also grey literature (online professional reports). On the other hand, the solution for the recommendation mechanism has been designed as scalable as possible, allowing the possibility to include new usability methods not initially included in the considered set. Also, we have mitigated the evaluation bias by considering a convenient number of experts [66, 67] that worked individually. This allowed to maximize homogeneous, independent and unbiased evaluations from each evaluator, as it has been corroborated by statistical evidence.

All these findings helped answer research question RQ1.2, and provided the information needed to systematize the recommendation mechanism.

3.2 Recommendation Mechanism

Taking into account all the information gathered in the preliminary research, the inference engine was developed by generating first a key knowledge base consisting in three 43x17 tables to codify the 43 usability methods against the 17 project constraints, divided into the aforementioned 5 project stages. These tables contain, for each cell, application-suitability values for each usability method depending on whether a project constraint is fulfilled or not, calculated as the arithmetic mean of the experts' ratings. The information contained in each table is explained down below:

- Table *Y* (yes table): This table contains the suitability values for each usability method when a constraint fulfills the project restriction, i.e. – when the situation described by the constraint is applicable to the project.
- Table *N* (no table): This table contains the suitability values for each usability method when a constraint is not present or does not fulfill a project restriction, i.e. – the situation described by the constraint is not applicable to the project.
- Table *DK* (do not know or uncertainty table): This table contains the suitability values for each usability method when a constraint is unspecified or uncertain according to a project restriction, i.e. – when the situation described by the constraint is uncertainly applicable to the project. This table enables key stakeholders to work with unsure project constraints, obtaining recommendations anyway.

The algorithm in charge of calculating the recommendation elaborates a final ranking using the input constraints specified by stakeholders or development-team members in terms of *yes*, *no*, or *do not know* (undefined) for each constraint, and the three tables described above. This way, the input constrains are arranged in a table of 1x17 that can be defined as $K = \{k_1, k_2, \dots, k_{17}\}$, i.e. – a finite collection of input constraints, where $k_l \in \{0, 1, 2\}$, such that l

represents each of the 17 constraints for a given project. This way, if $k_l = 1$ then table Y is used to calculate the ranking, else if $k_l = 0$ then table N is applied; otherwise, if $k_l = 2$ then table DK is used.

In order to calculate the final ranking, an accumulator variable acc_i is defined to calculate the weight for a specific usability method i considering the input K :

$$acc_i = \begin{cases} acc_i + Table_{i,j}^{k_l}, & \text{if } Table_{i,j}^{k_l} \neq 0, 1; \\ 0, & \text{if } Table_{i,j}^{k_l} = 0; \\ acc_i - \alpha, & \text{if } Table_{i,j}^{k_l} = 1. \end{cases} \quad (1)$$

Where $Table_{i,j}^{k_l}$ represents the suitability value of the usability method i according to the project constraint j in table Y ($k_l = 1$), N ($k_l = 0$) or DK ($k_l = 2$), $k_l \in K$, $\forall i = 1 \dots 43$, $\forall j, l = 1 \dots 17$.

The algorithm applies penalization for lower values, subtracting the value $\alpha = \max(Table^{k_l})$ when $Table_{i,j}^{k_l} = 1$. In case of having $Table_{i,j}^{k_l} = 0$, the recommendation percentage value is 0%, as this means that the method cannot be applied when the constraint fulfills. Otherwise, the accumulated value is added with the existing weight (a value between 2 and 5) of the corresponding table (Y , N or DK):

Once acc_i is calculated $\forall i = 1 \dots 43$, we obtain the $RecommendationValue_i$ for each usability method, multiplying acc by a coefficient called $Stage_q$. This coefficient represents the weight of the usability method i for a specific stage of the project q , $\forall q = 1 \dots 5$. This operation is achieved according to the information appearing in ISO/TR 16982 and the preliminary research carried out, and it confers special importance to the current stage of the project (*initial, analysis, design and implementation, testing and maintenance*) when determining the suitability of a method:

$$RecommendationValue_i = acc_i \cdot Stage_q \quad (2)$$

After this calculation, we proceed to sort the resulting 43x1 table, $\forall i = 1 \dots 43$:

$$SortedRecommendations = \{Sort(RecommendationValue_i)\} \quad (3)$$

In addition, sorted values are transformed into percentage values by applying normalization, in order to obtain a list with sorted percentages including all usability methods. Besides, to improve classification and presentation of results, usability methods are classified into the following groups: recommended, neutral and not recommended. To carry out this task, we use terciles to classify the resulting methods, calculating corresponding percentiles 66.6% and 33.3% to categorize the presentation of results. The idea is to obtain groups with a quite similar number of methods classified, except in the case where recommendation percentage is 0; in this case the methods are directly classified as not recommended. This way, recommended usability methods are clearly presented to key stakeholders, and thus they are able to differentiate which are the most suitable usability methods to apply for a specific project and stage, and which are neutral or not directly applicable.

3.3 The tool

In order to verify and implement the proposed recommendation mechanism, we have created STRUM, which draws upon the recommendation algorithm described above to feature an easy to use usability-scheduling tool.

The main features of STRUM are the following:

- Create, set up and remove usability projects, making it possible the management of several projects simultaneously. Key stakeholders have access to a general view of all their projects, including basic information, current stage and progress percentage (see Fig.1).

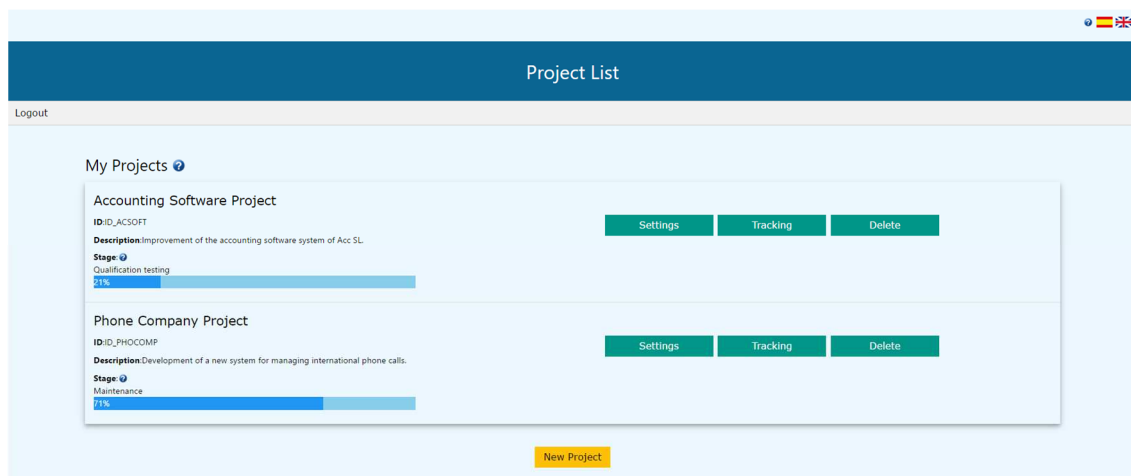


Fig. 1. Projects list and status in STRUM.

- Establish a planning for a specific project and automatically obtain recommendations for the current project stage. Key stakeholders are able to select the stage to get recommendations, allowing to go forward and backward when necessary.
- Get recommendations about which usability methods are more suitable to apply in a certain stage of a project, taking into account the constraints previously introduced. Constraints are introduced by key stakeholders through an easy-to-complete questionnaire. The tool provides recommendation even when key stakeholders are not sure about a certain constraint. In fact, constraints can be modified at any time, which means that the tool applies the recommendation mechanism and automatically updates the information accordingly. Fig.2 shows the method recommendation panel, where each usability method is classified into the three aforementioned categories (recommended, neutral and not recommended) together with a recommendation percentage. The checked methods represent those that have already been applied in the project.

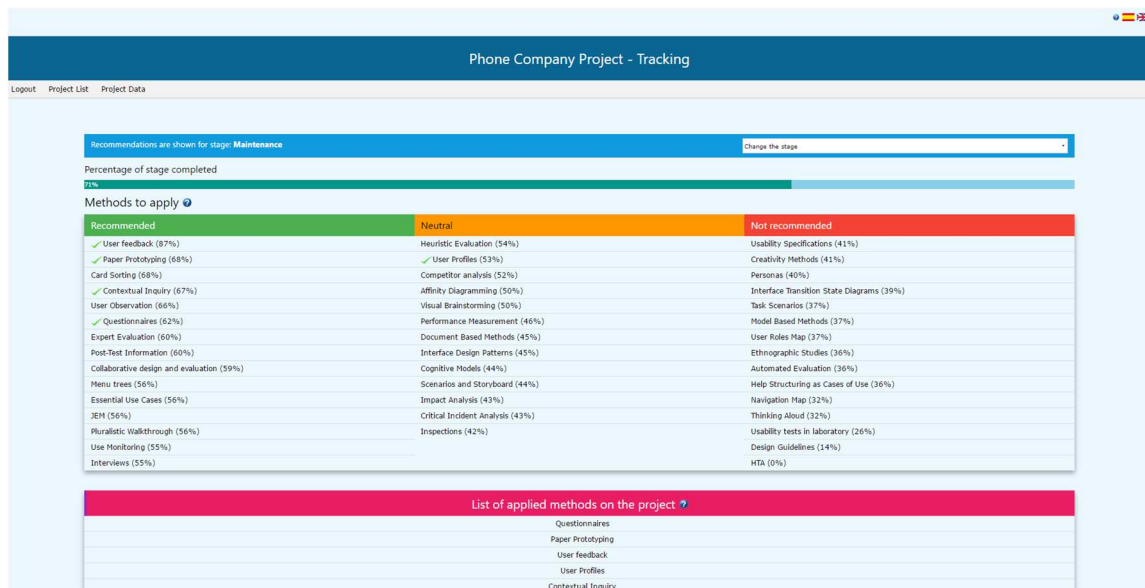


Fig. 2. Project and methods tracking in STRUM.

- Consult further information about the application of each usability method and insert comments. Different functionalities enable stakeholders to consult how to apply a specific usability method according to the current project stage. In addition, stakeholder can insert comments about already applied methods, and access this information at any time. Fig.3 shows the method panel where key stakeholders can consult information about *user feedback* method and can insert application comments. This way, when key stakeholders want to know more about a usability method, a brief description of it and an explanation about how to apply it is presented just clicking on the name of the method.

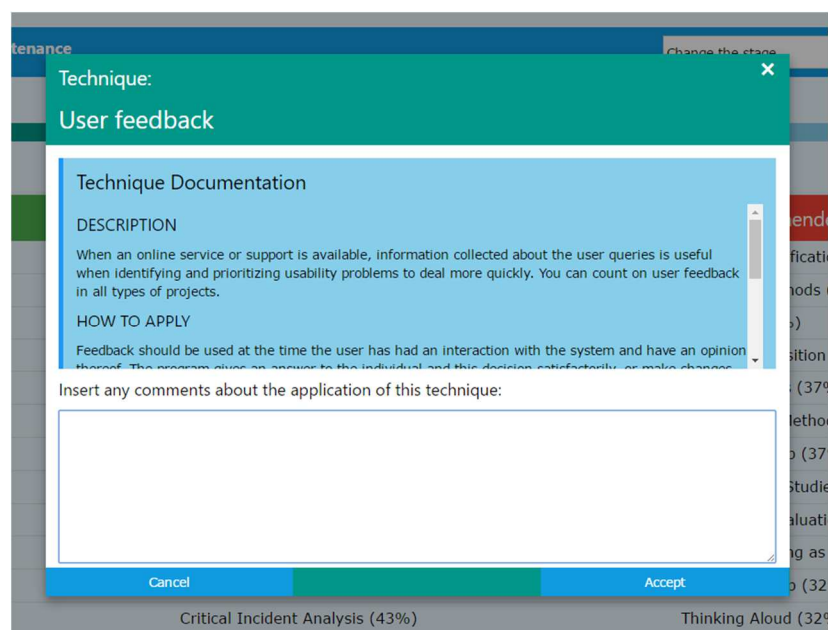


Fig. 3. Method information and comment insertion in STRUM.

Advanced users can modify the inference tables used to calculate the methods recommendation if necessary. Furthermore, STRUM has been conceived as an easy to use and learn CASE tool. Multiple help tips have been included, and the user interface has been designed in order to be appealing to key stakeholders, ensuring full control and allowing to navigate easily. Moreover, accessibility aspects have been also considered, highlighting a responsive design that successfully adapt to different platforms, which allows to use the tool in any kind of device.

3.4 Application Scenarios

We present two different application scenarios to verify the supporting tool and validate the recommendation mechanism with experts. These scenarios consist of different software projects to develop, and they are inspired by those included in the ISO /TR 16892. However, some aspects of the original scenarios have been modified in order to make them more complex, complete, and highlight specific features of the recommendation mechanism and the tool presented. In fact, we have added new constraints to each scenario, including uncertainty in some of them.

We proceeded to introduce each scenario's constraints in STRUM, and then we selected the project phase for which recommendations are required. Explanations of the different scenarios are provided down below, as well as the output obtained from STRUM that consists of the method-recommendation ranking and the corresponding ratings for each application scenario.

3.4.1 Scenario 1

This is a project where the client represents a small consulting company. The project, proposed by the company's manager, consists of developing an internal tool to be used even by employees who are not expert in computers. The company considers the usability of the software as essential, due to the complexity of the tasks to develop. Several prototypes are likely to be required because of the changing specifications of the project. Because of this, the number of tasks to develop is not clear. The interaction requirements have not been specified either. As a constraint, the company informs that they cannot afford a big budget. The prospective users of the tool include a variety of roles, such as director, director of production, and so on. Those people are not used to computers, being the tasks new for them. Moreover, they can only use the software once a month, so they are not likely to be available for testing. To supply this problem, the company has hired usability experts to collaborate and determine the usability degree of the tool. The user interface should be conceived to guarantee a high usability degree. The stage required to have recommendations for the methods to apply is Design and Implementation.

Taking into account this information, the corresponding constraints were introduced in STRUM, including uncertainty such as unclear number of tasks or interaction requirements. The output provided by the tool is shown in Fig.4. This figure depicts, for the *Design and Implementation* stage, the recommended methods, which are classified into *Recommended*, *Neutral* and *Not Recommended*. Following each method, a percentage representing its relative degree of suitability is shown.

Methods to apply 		
Recommended	Neutral	Not recommended
Menu trees (80%)	Design Guidelines (51%)	Thinking Aloud (0%)
Document Based Methods (80%)	Inspections (50%)	Performance Measurement (0%)
Interface Design Patterns (68%)	Competitor analysis (47%)	Critical Incident Analysis (0%)
Expert Evaluation (63%)	Essential Use Cases (46%)	Questionnaires (0%)
Interface Transition State Diagrams (63%)	User Roles Map (40%)	Collaborative design and evaluation (0%)
Personas (63%)		Paper Prototyping (0%)
Heuristic Evaluation (61%)		User Observation (0%)
Usability Specifications (60%)		Creativity Methods (0%)
User Profiles (60%)		Card Sorting (0%)
Help Structuring as Cases of Use (58%)		Scenarios and Storyboard (0%)
Model Based Methods (58%)		Interviews (0%)
Use Monitoring (55%)		User feedback (0%)
Navigation Map (54%)		Ethnographic Studies (0%)
Automated Evaluation (53%)		HTA (0%)
Impact Analysis (53%)		Affinity Diagramming (0%)
		Contextual Inquiry (0%)
		Visual Brainstorming (0%)
		Usability tests in laboratory (0%)
		Task Scenarios (0%)
		Cognitive Models (0%)
		Pluralistic Walkthrough (0%)
		Post-Test Information (0%)
		JEM (0%)

Fig. 4. STRUM Recommendation for scenario 1.

As shown in Fig. 4, *recommended methods* include those not requiring the final user to be available for testing, as this is one of the constraints of this scenario. This way, methods involving usability experts are recommended. *Menu Trees* (80%) is one of the methods having a higher percentage of recommendation, as it is easy to apply and is not very costly considering the constrain related to the budget. This method makes it possible to model the user interface and tasks, which are meant to be complex in this scenario. With a same recommendation percentage, *Document Based Methods* (80%) would be also a good option, as they allow to determine the degree of usability avoiding user testing. In order to apply these methods, it is necessary to count on usability experts (as required in this scenario). *Interface Design Patterns* (68%) would be also useful to increase the degree of usability of the system through graphical representations that can be tested through the different prototypes to develop. This would improve the interaction design and the final user interface. In addition, *Personas* (63%) would help obtain information about users in order to analyze their characteristics. Additionally, as *neutral methods* we can find recommendations such as *Inspections* (50%), which allow to improve the degree of usability, validating the prototypes and preventing them to have errors at reduced cost. However, the number of tasks to develop is very important in this method and its application is not clear in this case, so the recommendation percentage is not especially high. Finally, and considering the *not recommended methods*, here we can find those requiring the presence of final users, which are limited according to the scenario's constraints. Furthermore, applying these *not recommended methods* would increase the project budget. This is why the recommendation percentage for these methods is 0%. Among them, we can see *Observation of Users*, *Card Sorting* or *Interviews*, where final users have an essential and participatory role. Besides, as the tasks are new for the employees, methods like *Critical Incident Analysis* or *Cognitive Models* are not recommended, as they are based on previous knowledge about the tasks to evaluate.

3.4.2 Scenario 2

In this project, the client represents a telecom company. The project, proposed by the marketing service of the company, is aimed at designing a telephone software product for small companies that cannot afford to set up a phone standard. The users are members of such small companies that manage the communications on their own. A technical improvement is required to automatize the process, although there is uncertainty about the degree of

complexity of the tasks and the final software product. The users, which are available during the development process, should be able to test a great variety of tasks concerning this new software. It is not an urgent project, but a functional version is expected to be finished by the end of this year, as it is expected to be presented in a conference. Because of this, it would be positive to get some feedback from users as soon as possible. The company carried out a viability study and it expects to be able to afford to the costs without difficulties. However, there are still some unclear aspects such as the specification of the product, the severity of possible errors, or the participation of users with disabilities. Also, the participation of usability experts remains unclear. It is expected that users carry out an effective interaction with the software, leading to an important improvement on the development of user tasks. As the project is in an early state, it is required to have methods recommendation for this initial phase.

Considering the above information, the corresponding constraints were introduced in STRUM, including uncertainty about the specification of the product, the severity of possible errors, the participation of users with disabilities to test the product, and the participation of usability experts. The output recommendation provided by the tool is shown in Fig. 5.

Methods to apply ?

Recommended	Neutral	Not recommended
Card Sorting (85%)	Scenarios and Storyboard (56%)	Model Based Methods (40%)
User Observation (84%)	Competitor analysis (56%)	User feedback (32%)
Paper Prototyping (80%)	Menu trees (55%)	Automated Evaluation (32%)
Document Based Methods (74%)	Expert Evaluation (54%)	Post-Test Information (29%)
Pluralistic Walkthrough (72%)	Usability Specifications (54%)	Thinking Aloud (27%)
Collaborative design and evaluation (72%)	Inspections (54%)	Usability tests in laboratory (24%)
User Profiles (69%)	User Roles Map (53%)	Use Monitoring (24%)
Personas (68%)	Essential Use Cases (52%)	Help Structuring as Cases of Use (23%)
Visual Brainstorming (66%)	Heuristic Evaluation (52%)	Impact Analysis (23%)
Contextual Inquiry (66%)	Task Scenarios (48%)	Interface Design Patterns (23%)
Affinity Diagramming (65%)	Ethnographic Studies (48%)	Design Guidelines (21%)
Interviews (64%)	Navigation Map (46%)	Cognitive Models (0%)
Questionnaires (63%)	Performance Measurement (44%)	Critical Incident Analysis (0%)
JEM (57%)	Interface Transition State Diagrams (42%)	HTA (0%)
Creativity Methods (57%)		

Fig. 5. STRUM Recommendation for scenario 2.

As there is uncertainty about many aspects in this project, the methods that generally result more useful and lead to good results in most circumstances obtained the highest recommendation values, varying depending on the kind of constraint. *Card Sorting* (85%), with the highest percentage, is considered especially useful on early stages. Also, it results easy to apply when experts are not available, producing notable benefits in terms of usability when it is applied from the very beginning. Using this method, it would be possible to better analyze information and the tasks for future users, despite the technical change to accomplish. *Observation of Users* (84%) is also recommended, as there are no important time or cost constraints. Using this method, it would be possible to examine how employees work, what tasks are the most complex and therefore achieve a more efficient interaction. *Paper Prototyping* (80%) is also recommended as it helps validate context with users in early stages, thus improving specifications gradually. Among *neutral methods*, *Menu Trees* (55%) results easy to apply, and it provides good contributions in terms of usability. As there is a variety of user tasks, this method can be helpful when designing intuitive menus. However, it has been conservatively rated since other methods involving users are more suitable in this scenario. *Expert Evaluation* (54%) is also considered as neutral because, despite the fact that it adapts properly to early stages when determining user needs, it requires the presence of usability experts, which is one of the uncertain constraints of this project. Moreover, *Inspections* (54%) provide important contributions to usability, especially when there is not a time constraint. However, in this scenario the number of tasks is high, and *Inspections* require a high level of

familiarity with the system, whereas in this scenario the users are unfamiliar with the product. As for the *not recommended methods*, *User Feedback* (32%) is not considered a good choice as there is no previous version of the product and users cannot contribute with their opinion. *Thinking Aloud* (27%) is also not recommended as it requires a functional prototype of the product to be tested and, as we are in an early stage of the project, this is not applicable. *Critical Incident Analysis* (0%) is even less recommended, as this is a new product that has not been used before in the company, and therefore it is not possible to provide previous performance measures.

3.4.3 Joint Analysis

Fig. 6 depicts a graphical representation of the joint results obtained from both application scenarios, where the variability of the recommendations for the 43 usability methods is clearly visible. As shown in Fig. 6, substantial differences can be seen in each scenario among certain values for methods such as *Card Sorting*, which is rated with 0% in scenario 1 and with a higher percentage value in scenario 2. This is due to the fact that this method is principally recommended for initial stages of the project as long as users are available to participate, allowing to elicit content requirements for certain kinds of software products. In general, methods that can be applied in early stages, and where the participation of users is highly appreciate, result better rated in scenario 2. This is also the case for usability methods such as *Contextual Inquiry*, *Creativity Methods*, *Interviews*, *Paper Prototyping*, *Pluralistic Walkthrough* or *User Observation*, to cite a few. In addition, methods such as *Design Guidelines*, *Interface Design Patterns* or *Menu Trees*, are principally recommended for advanced project stages, such as design and implementation, where the participation of final users is not strictly necessary, as it occurs in scenario 2. In addition, there are methods that can be applied to both scenarios with similar rates of recommendation. This is the case for methods such as *Competitor Analysis*, *Document Based Methods*, *Essential Use Cases*, *Inspections*, *Navigation Map*, *Personas* or *User Profiles*, to cite a few. These methods show diverse ratings depending on the project's stage and constraints, or can be reduced due to uncertain conditions, such as project budget, number of tasks, the participation (or not) of final users, etc.

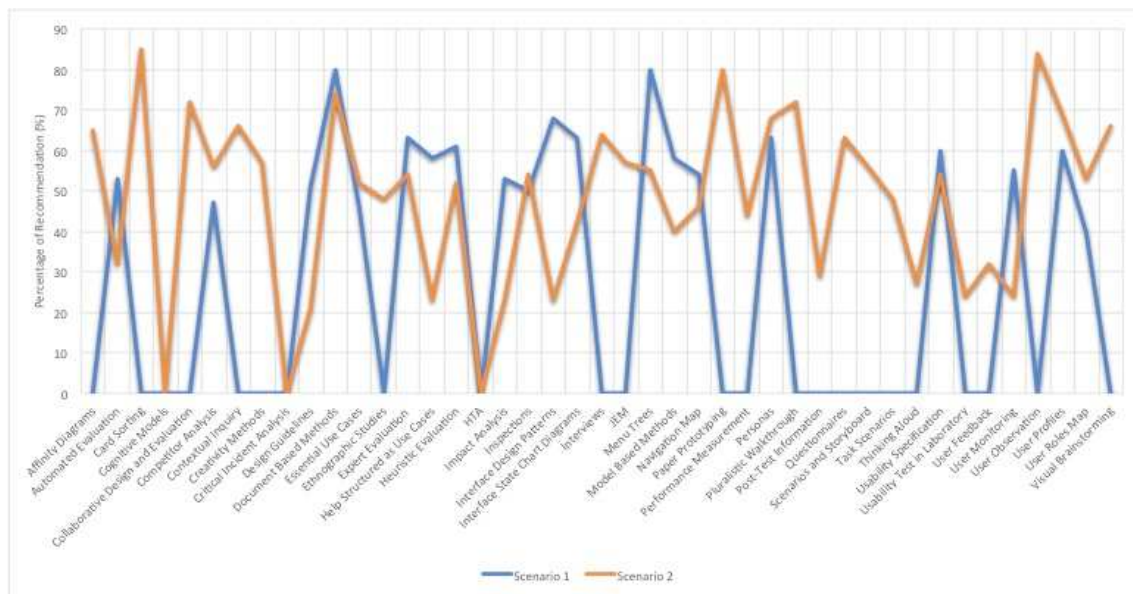


Fig. 6. Variability of the percentage value in the recommendation of usability methods for the two application scenarios.

3.4.4 Analysis of Recommendation Appropriateness

We wanted to know the recommendation appropriateness by analyzing the recommendation gap –i.e., the discrepancy between the recommendations stated by experts and those generated by the proposed mechanism, hoping to have a low difference (gap) to ensure the appropriateness of the recommendation obtained with STRUM. This way, we propose the following research question that can be considered as part of RQ1:

- [RQ1.3] *Can the recommendations generated by the proposed mechanism be considered as appropriate?*

To carry out this task, we asked 10 new experts from our institution to participate in the analysis. They were 8 men and 2 women who did not participate in previous evaluations. As for their background, they are lecturers having advanced skills in software engineering and human-computer interaction.

To carry out the analysis, we asked experts to classify each usability method into 3 different categories: recommended, neutral and not recommended, according to the specifications of each scenario described in Section 3.4.1 and 3.4.2. Once finished, we obtained the following information for each scenario:

- The inter-rater reliability (*kappa-value*).
- Expert classification sets (one for each category) as a result of assigning each method to the category with highest rating (i.e., the category that obtained the highest number of matches between experts).

Then, we compared the expert classification sets with those obtained as output from STRUM and described in sections 3.4.1 (Fig. 4) and 3.4.2 (Fig. 5) for scenario 1 and 2, respectively.

In general, results obtained provided similar recommendation and agreement among experts, obtaining *kappa-values* of 0.83 and 0.81 for scenario 1 and 2, respectively, and so indicating a high degree of agreement among expert classifications.

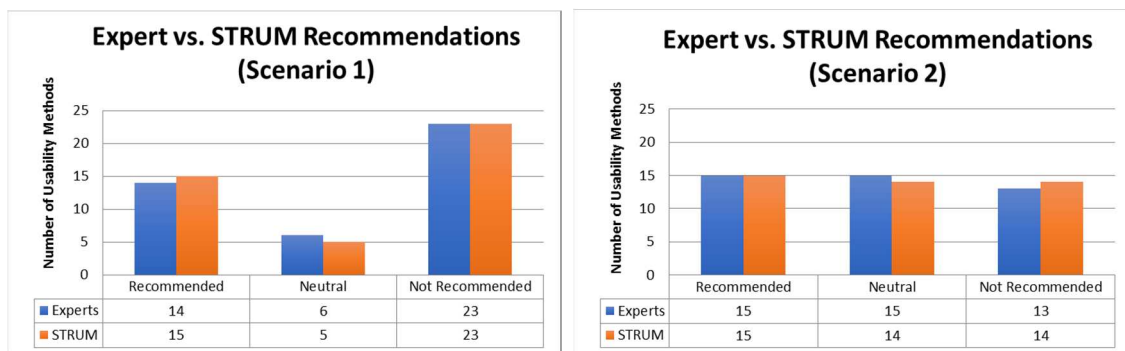


Fig. 8. Expert vs. STRUM recommendations for scenario 1 (left) and scenario 2 (right).

Comparing the expert classification sets with those obtained with STRUM, a small number of misclassified items were detected, according to the percentiles division calculated by the tool. As shown in Fig. 8, the number of classified methods by experts and the system is quite similar for each scenario. Only a 2.33% classification difference (1 out of 43 methods) was detected in each scenario. This classification gap was calculated as the difference between the amount of

correctly classified methods in each category, taking as a reference the classification accomplished by experts.

As for scenario 1, there was 1 misclassified method (*Impact Analysis*), which was considered as neutral by experts but as recommended by the system. This was due to the proximity to the percentile classification borderline that may lead to consider this method as recommended or neutral depending on the expert criteria. In fact, *Impact Analysis* method is useful for prioritizing the fixing of usability problems in a redesign or even for analyzing usability problems in advance. Besides, this method is usually independent of the final user involvement, but it can be useful when usability experts are available. This also may vary depending on the project budget, according to the scenario's specifications.

As for scenario 2, we found 1 misclassified method (*Model Based Methods*). This method was considered as not recommended by the system but as neutral by experts. Similarly, this is a method closer to the borderline between neutral and not recommended methods. Additionally, and in contrast to scenario 1, scenario 2 involved higher uncertainty about many aspects of the project. In general, *Model Based Methods* are useful to create user interface specifications for modeling user behavior and data, but also for considering formal approaches to predict user performance. Considering that this scenario contains uncertainty about the specification of the product and the complexity degree of the tasks, *Model Based Methods* could be certainly considered as neutral or even not recommended.

As analyzed, there was a low percentage difference, mostly due to the completeness of the constraints introduced in STRUM. Besides, these differences were found in the borderline, so we think that they are not significant, as most relevant methods were successfully recommended according to the principal specifications. In addition, the gap can be reduced depending on the level of completeness of the constraints. In general, the more complete the constraints are, the less percentage differences occur in the recommendation. All in all, and as previously commented, advanced STRUM users can modify the inference tables used to calculate the methods recommendation in order to tune them if necessary.

These findings provide an affirmative answer to RQ1.3, concluding that the recommendations obtained are appropriate according to the analysis carried out by experts, obtaining a low and insignificant gap to support the evidence.

In summary, the recommendation mechanism presented and the results obtained from the application scenarios helped answer affirmatively RQ1, and thus affirm that it is possible to create a mechanism to provide appropriate recommendation on usability methods according to specific project constraints, even under uncertainty.

4. Evaluation and Results

In order to have an understanding of the usability in STRUM, a user test was performed. The objective was to observe whether users consider our approach useful and satisfactory.

4.1 Evaluation method

The test consisted of a set of recruited users individually obtaining recommendations for the scenario 1 previously described in Section 3.4.1. This was a controlled evaluation, where users were briefly introduced to the objective of the test, the scenario and the tasks to perform. To

have specific measures, 7 representative tasks were evaluated using STRUM. We utilized the *thinking aloud* [16] protocol, where users were asked to express their opinions and thoughts while performing the tasks. During the interaction, we obtained different measures such as time elapsed and the degree of effectiveness and efficiency in performing each task. After performing the tasks and obtaining the resulting recommendations from the system, users were asked to fill in the USE questionnaire [67, 68], which helps measure usability through 4 variables (usefulness, ease of use, ease of learning and overall satisfaction) [69, 70]. USE questionnaire includes 30 questions grouped into 4 different categories: 8 questions for measuring usefulness, 11 questions devoted to measure ease of use, 4 questions used to measure ease of learning, and 7 questions for measuring satisfaction. Responses are assessed in a Likert scale ranging from 1 to 7, where 1 means “strongly disagree” and 7 “strongly agree”. Results were normalized in order to obtain percentage values.

4.2 Variables and Research Questions

Taking into consideration the method previously described, the following research variables were measured during the evaluation, all those corresponding to dependent variables:

- Quantitative variables:
 - o Effectiveness: average percentage of tasks successfully accomplished by users.
 - o Efficiency: average time spent (seconds) by users to complete each task.
 - o Usefulness, satisfaction, ease of use and ease of learning: normalized average percentage obtained from the USE questionnaire.
- Qualitative variable:
 - o User behavior and observations obtained from the *thinking aloud* sessions.

We also considered specific research questions, which are related to RQ2 described in Section 1.1:

- [RQ2.1] *Do the users perceive the approach as useful?*
- [RQ2.2] *Does the solution proposed feature an acceptable cognitive load?*
- [RQ2.3] *Can the overall usability of the system be considered as acceptable?*

These research questions will be validated with the results obtained from the evaluation. Specifically, to answer RQ2.1 we hope to achieve percentage values over 75% for usefulness and satisfaction variables. Additionally, to answer RQ2.2, we hope to achieve acceptable efficiency values and effectiveness percentages closer to 100%, as well as percentage values over 75% for ease of use and learning variables. In addition, to answer RQ2.3, we hope to achieve percentage values over 75% for usefulness, satisfaction, ease of use and ease of learning variables. Besides, qualitative information from the *thinking aloud* sessions was analyzed in order to obtain further information or problems found in order to enhance the knowledge obtained from direct measures and answer previous research questions in the affirmative.

As stated, we established 75% as a final acceptance benchmark for most usability values. This is a positive benchmark level, higher than others used in usability measurement [69, 71], to

indicate agreement with respect to user satisfaction when responding to the different questions in a Likert scale; 1-7 for the case of the USE questionnaire. A normalized average measure of 75% represents a number between 5 and 6 (i.e., between agree and very agree), which can be considered as an acceptable value for most usability dimensions.

4.3 Recruited Participants and Sample Size Discussion

We recruited 15 users for the test. They were 9 men and 6 women, with ages ranging between 21 and 23 ($M=22$; $SD=0.85$). Users who participated in the evaluation were software engineers –i.e., the key stakeholders of our systems, being familiar with usability methods and user-centered practices.

According to reviewed literature, most authors agree that there is not a fixed number of users to test an interactive software, as it principally depends on the objective of the evaluation [71, 72]. One key concern is to understand the nature of the users involved [72], as well as the kind of usability evaluation to carry out. Whereas formative evaluations may be accomplished with a more reduced set of users, summative evaluation demands a broader range. In addition, the comparative nature of the evaluation and the distinct groups of users involved in the software usage is also relevant to consider a higher number of users to test [73].

Besides, previous studies point out that the fewer users that a problem impacts, the larger the sample size is needed to have a good chance of finding it in an evaluation [72]. In fact, the binomial probability (or Poisson equivalent) is often used to estimate the number of users needed to detect an approximate percentage of usability problems given a certain probability that a user would encounter a problem [71, 73, 74]. Thus, the main concern is to state the problem occurrence probability in the interactive software.

Also, the combination of usability techniques applied to evaluate the interactive software is also related with the sample size. Based on predictions using observed data, a general rule for optimal sample size using *thinking aloud* protocol and other expert evaluations would be $10+2$ to reach 80% overall discovery rate. This can be applied to general or basic evaluation situations [75].

According to these arguments, the main aim of the evaluation presented in this paper is to detect important usability problems that would benefit from improvements in pursuit of a new release. More specifically, we expect to identify main problems that users have and test major functionalities related to the recommendation mechanism using different usability evaluation techniques to have a broader perspective. In addition, the user interface of our tool does not feature a complex front-end functionality, as it can be considered as a CASE tool for the specific purpose of recommending and reporting information about usability methods in a straightforward and very simple way.

According to that, and using the binomial probability, we expect to identify problems that impact 18% or more users with a 95% chance of observing them in the evaluation. This way, the number of users to test can be calculated as $\text{Log}(1-0.95) / \text{Log}(1-0.18) \approx 15$ users. It is worth noting that the discovery rate can be considered as high (95%), and an impact percentage of 18% enables to find complex problems. In fact, a problem impact percentage among 30%-60% implies problems affecting a great deal of users (i.e., coarse-grain errors), whereas reducing this figure to a more restrictive percentage (10%-20%) helps find a higher number of problems, and more specifically those being more difficult to find (i.e., less obvious

problems). This tradeoff would help find most important usability problems, so we think that a sample size of 15 is adequate for this evaluation.

4.4 Tasks Performed

Users were asked to create a project corresponding to the application scenario 1 previously described in Section 3.4.1. We provided users with the scenario and a list with the 7 tasks to perform in order to control the evaluation and have specific measures to assess the dependent variables defined. The tasks were the following:

- *Register a user in the system and log in (T₁)*: the user was asked to log in to STRUM, after creating a user account and introducing her/his corresponding personal data.
- *Create a project and insert basic data and planning (T₂)*: the user was requested to create a new project, introducing id, name and description. He was also requested to establish a planning for three different project stages (*initial, analysis and design and implementation*).
- *Define the constraints for the project (T₃)*: the user was requested to enter the settings screen and determine the constraints for the project of scenario 1 by answering the corresponding constraints questionnaire.
- *Modify project data and planning (T₄)*: the user was asked to update the description of the project and modify the planning with new dates.
- *Obtain recommendations for a specific stage (T₅)*: the user was requested to enter the tracking screen and obtain recommendations for a specific stage (*design and implementation*).
- *Obtain information of a method, insert a comment and check it (T₆)*: the user was asked to select a specific usability method (*user feedback*), analyze the available documentation for it, insert a comment and see the result.
- *Remove the project and log out (T₇)*: the user was requested to remove the project created and log out of the system.

4.5 Analysis of Usability Results

The analysis of the interaction carried out by users provided adequate results in terms of effectiveness and efficiency.

With respect to effectiveness, users performed all the tasks without requiring help or assistance. This implies 100% effectiveness reached.

	T ₁	T ₂	T ₃	T ₄	T ₅	T ₆	T ₇
Geometric Mean	56.03	86.25	103.69	55.21	20.22	62.47	11.83
Min	34.00	60.00	87.00	43.00	14.00	40.00	7.00
Max	69.00	103.00	147.00	71.00	32.00	78.00	17.00
Median	58.00	90.00	97.00	54.00	20.00	61.00	13.00
Faster Time (SL)	50.00	80.00	93.00	43.00	19.00	40.00	7.00
CI (95%)	4.35	6.48	8.59	5.12	2.15	5.05	1.53

Table 3. Statistics corresponding to efficiency on task performance in seconds.

As for efficiency, we measured the average time on task during the *thinking aloud* sessions. As recommended in [71], we used the geometric mean instead of the arithmetic mean, since time values may be skewed by a few outliers in small sample sizes (<25). In addition, the geometric standard deviation is a misleading value in this case, so we reported the confidence intervals

(95%) instead. Additionally, as task time can be meant as a relative value, we included the faster task time according to the *bootstrapped specification limit* [71] to identify the maximum acceptable time for each task.

Table 3 depicts the statistical results in seconds. As shown, users accomplished all the tasks in less than 2 minutes on average. Only tasks T_2 and T_3 have higher averages, as they are related to heavier actions such as project creation (T_2) and constraints introduction (T_3). Acceptable times for tasks T_2 and T_3 are 80 and 93 seconds, respectively. On the contrary, T_5 and T_7 required less time to be performed, as users easily identified the appropriate actions to achieve them. Acceptable times for tasks T_5 and T_7 are 19 and 7 seconds, respectively. In addition, the 95% confidence intervals can be also considered as acceptable, being smaller than 9 seconds in all cases, which demonstrates the reliability of average time values.

With respect to user behavior and observations obtained from the *thinking aloud* protocol, in general we perceived an easy interaction. However, we observed improvements to take into consideration for future releases. For instance, the completion time of T_4 could have been reduced if the date-picker plugin had not been included, due to the time spent by some users to search the right date in an interactive way; users confirmed this fact throughout the test. In addition, users missed some tips in specific user interface locations to clearly understand the meaning of the information shown. All in all, users agreed that the tool is intuitive and does not pose major difficulties when performing the tasks. Moreover, users highlighted the user-friendly interaction, the user interface design and the bilingual functionalities available.

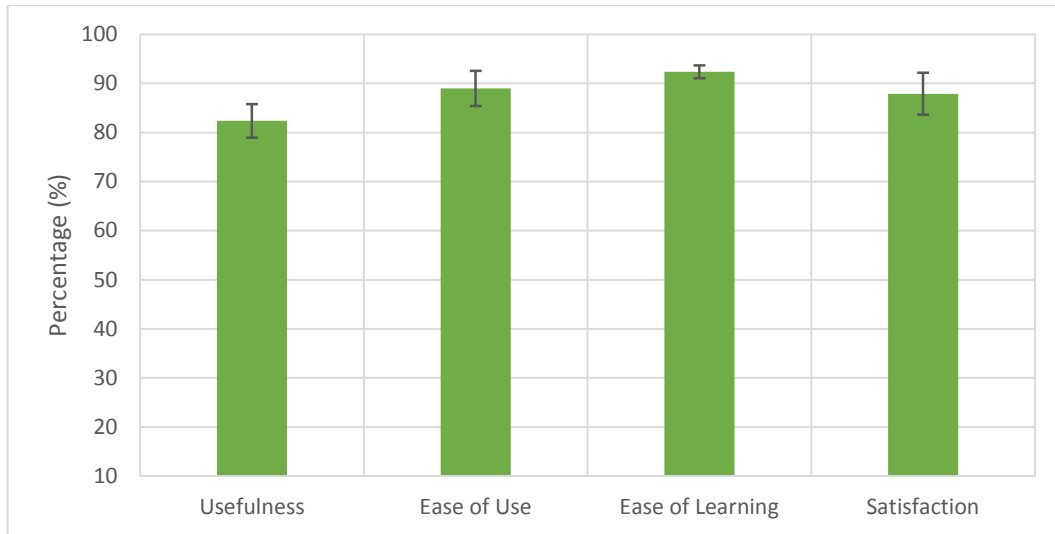


Fig. 7. Average percentage values for each USE dimension including error bars ($\pm \sigma$).

In general, an acceptable reliability value was obtained for the USE questionnaire measures ($\alpha=0.784$). Fig. 7 depicts the average values for each usability dimension, where high rates were obtained for usefulness: 82.38% ($M=5.77$ and $SD=0.24$ in 1-7 Likert scale), ease of use: 89.00% ($M=6.23$ and $SD=0.25$ in 1-7 Likert scale), ease of learning: 92.38% ($M=6.47$ and $SD=0.09$ in 1-7 Likert scale) and perceived satisfaction: 87.89% ($M=6.15$ and $SD=0.30$ in 1-7 Likert scale). The average value for the 4 dimensions is 87.86% ($M=6.15$ and $SD=0.29$ in 1-7 Likert scale), which can be considered as an acceptable overall result.

Results obtained helped answer the proposed research questions. Thus, RQ2.1 can be answered in the affirmative, as users considered useful and satisfying the approach. Average values obtained for usefulness and satisfaction were 82.38% and 87.89%, respectively, while

the minimum value to validate this claim was 75%. In addition, RQ2.2 can also be answered in the affirmative, as users perceived a low cognitive load supported by acceptable values for effectiveness and efficiency, featuring also high values for ease of use (the average value obtained in the evaluation was 89.00%) and ease of learning (the average value obtained in the evaluation was 92.38%). The minimum value to validate this claim was 75%. Finally, the average value obtained for the 4 USE dimensions represents a good estimation of the overall usability of the approach. According to that, average value resulted in 87.86%. Besides, no serious problems were found during the user interaction and, as average usability value is over 75% (minimum value to validate the claim), RQ2.3 can be also answered in the affirmative.

In general, all results obtained helped answer affirmatively RQ2, and thus affirm that it is possible to systematize the recommendation mechanism by means of a supporting tool to schedule and provide additional information about each usability method, also being satisfactory for key stakeholders.

5. Conclusions

The number of interactive software products increases every day more. Usability should be considered in all stages of a software project in order to avoid misunderstandings and increase user satisfaction [1, 3]. In this sense, it is necessary to schedule usability methods during all the stages of the project, and this should be considered and arranged at early stages. In addition, there is a high number of existing usability methods. Although there exist different standards and a considerable amount of documentation available, there is a necessity for tools that automatically help recommend the suitable usability methods to use in each project stage according to its characteristics and constraints. This should be systematized through an easy to use mechanism for key stakeholders in multidisciplinary development teams, which include heterogeneous people with different background.

To overcome such drawbacks, we have presented a recommendation mechanism inspired by ISO/TR 16892. In contrast to the 12 usability methods provided by the standard, our approach features recommendations for a total of 43 usability methods to be applied in different stages of a software project, taking into account the suitability degree of each method according to the characteristics of the project. Additionally, our approach deals with uncertainty by providing recommendation even when a certain project constraint is unknown, empowering project scheduling at early stages. The proposed mechanism has been implemented through a tool called STRUM (*Scheduling Tool for Recommending Usability Methods*). STRUM goes beyond the mere description of the suitability of the usability methods, as it happens in ISO/TR 16892, and it includes new features for project management, allowing key stakeholders to have a general view of their work and update it easily. Besides, it is possible to add a planning and automatically obtain usability methods recommendation for the current stage of the project.

The application scenarios, the expert analysis for the recommendation appropriateness and the controlled evaluation with users helped answer main research questions. Thus, RQ1 has been answered in the affirmative, as it is possible to create a mechanism to provide appropriate recommendation of usability methods according to a project's constraints, even under uncertainty. Besides, results obtained from the user testing also helped answer RQ2 in the affirmative, and thus affirm that it is possible to systemize the recommendation of

usability methods with a scheduling tool being satisfactory for key stakeholders and showing high levels of usefulness, satisfaction and overall usability.

As for future work, we expect to improve the issues reported during the evaluation sessions. In addition, new possibilities will be proposed, as dealing with specific user-centered process and intermediate products, roles, or customizable features. Also, as the recommendation mechanism and tool have been designed in a scalable way, it would be interesting to include new methods and provide exportation mechanisms in the form of reports, which would be useful for project documentation. A further extension would consist in comparing different usability attributes [76] from the application of methods recommendation.

Acknowledgements

This work was partially supported by the Spanish Government [grant number TIN2014-52129-R]; and the Madrid Research Council [grant number S2013/ICE-2715].

References

- [1] L.A. Rojas, J.A. Macías. Bridging the gap between information architecture analysis and software engineering in interactive web application development. *Science of Computer Programming*, 78, 11 (2013), 2282-2291.
- [2] A. Seffah, E. Metzker. 2004. The obstacles and myths of usability and software engineering, *Communications of the ACM*, v.47 n.12, p.71-76, December 2004.
- [3] J.A. Macías, T. Granollers, P. Latorre. *New Trends on Human-Computer Interaction: Research, Development, New Tools and Methods*. Springer, 2009. ISBN: 978-1-84882-351-8. DOI: 10.1007/978-1-84882-352-5.
- [4] X. Ferré. 2003. Integration of Usability Techniques into the Software Development Process *Proceedings of ICSE 2003 Workshop on Bridging the Gaps Between Software Engineering and Human-Computer Interaction*, May 3-4, 2003, Portland, Oregon.
- [5] ISO/TR 16982-2002. *Ergonomics of human-system interaction – Usability methods supporting human-centered design*.
- [6] A. Seffah, E. Metzker. 2004. The obstacles and myths of usability and software engineering, *Communications of the ACM*, v.47 n.12, p.71-76, December 2004.
- [7] P. Melo, L. Jorge. Quantitative support for UX methods identification: how can multiple criteria decision making help? *Universal Access in the Information Society*, v.14 n.2, p.215-229, June 2015.
- [8] Fischer, H., Strengé, B., Nebe, K. Towards a holistic tool for the selection and validation of usability method sets supporting human-centered design. *Proceedings of the Second international conference on Design, User Experience, and Usability: design philosophy, methods, and tools*, July 21-26, 2013, Las Vegas, NV.

[9] M.O. Leavitt, B. Schneiderman. Research-Based Web Design & Usability Guidelines (2007). Official U.S. Government edition, US.

[10] User Experience Professionals' Association. 2012. Usability Body of Knowledge. [ONLINE] Available at: <http://www.usabilitybok.org>. [Accessed 16 December 2017].

[11] N. Bevan. 2003. Usability Net. [ONLINE] Available at: <http://www.usabilitynet.org>. [Accessed 16 December 2017].

[12] US Department of Health and Human Services. 2015. Usability.gov. [ONLINE] Available at: <http://www.usability.gov/>. [Accessed 16 December 2017].

[13] A. Fernandez, E. Insfran, S. Abrahão, Usability evaluation methods for the web: A systematic mapping study. *Information and Software Technology*, Volume 53, Issue 8, August 2011, Pages 789-817.

[14] N. Bevan, X. Ferré, T.A. Escobar. 2012. Usability Planner. [ONLINE] Available at: <http://www.usabilityplanner.org/>. [Accessed 16 December 2017].

[15] K. Petersen, R. Feldt, S. Mujtaba, M. Mattsson. Systematic Mapping Studies in Software Engineering. In *Proceedings of the 12th international conference on Evaluation and Assessment in Software Engineering*. 2008, pp. 68–77.

[16] Nielsen Norman Group. 2008. Nielsen Norman Group. [ONLINE] Available at: <https://www.nngroup.com/>. [Accessed 16 December 2017].

[17] H. Beyer, K. Holtzblatt. (1998). *Contextual Design. Defining Customer-Centered Systems*. Morgan Kaufmann.

[18] S. Babbar, R. Behara, E. White. (2002). Mapping product usability. *International Journal of Operations & Production Management*, 22(10), 1071-1089.

[19] M.Y. Ivory, M.A. Hearst. (2001). The state of the art in automating usability evaluation of user interfaces. *ACM Computing Surveys (CSUR)*, 33(4), 470-516.

[20] J. Robertson. (2001). *Information Design Using Card Sorting*. [ONLINE] Available at: <http://www.steptwo.com.au/papers/cardsorting/>. [Accessed 16 December 2017]

[21] Card Sorting. (2017). Usability.gov. [ONLINE] Available at: <https://www.usability.gov/how-to-and-tools/methods/card-sorting.html>. [Accessed 16 December 2017]

[22] C. Lewis, C. Wharton. (1997). *Cognitive Walkthroughs. Handbook of Human-Computer Interaction*. M. Helander, T. Landauer, P. Prabhu (eds.) Elsevier, 717-732.

[23] H. Sharp, Y. Rogers, J. Preece. (2007). *Interaction design: beyond human-computer interaction*.

- [24] J. Nielsen. (1993). Usability Engineering. Academic Press.
- [25] H. Beyer, K. Holtzblatt. (1998). Contextual Design. Defining Customer-Centered Systems. Morgan Kaufmann.
- [26] S.W. Dekker, J.M. Nyce, R.R. Hoffman. (2003). From contextual inquiry to designable futures: What do we need to get there? IEEE Intelligent Systems, 18(2), 74-77.
- [27] C.R. Wilkinson, A. De Angeli. (2014). Applying user centred and participatory design approaches to commercial product development. Design Studies, 35(6), 614-631.
- [28] L. Nguyen, G. Shanks. (2009). A framework for understanding creativity in requirements engineering. Information and software technology, 51(3), 655-662.
- [29] O. Serrat. (2017). The critical incident technique. In Knowledge Solutions. Springer Singapore, 1077-1083.
- [30] D.J. Mayhew. (1999). The Usability Engineering Lifecycle. Morgan Kaufmann, San Francisco, USA.
- [31] M. Schuhmacher, S.P. Ponzetto. (2014). Knowledge-based graph document modeling. In Proceedings of the 7th ACM international conference on Web search and data mining, ACM. 543-552).
- [32] L.L. Constantine, L.A.D. Lockwood. (1999). Software for Use: A Practical Guide to the Models and Methods for Usage-Centered Design. Addison-Wesley, New York.
- [33] S. Yahya, M. Kamalrudin, S. Sidek, J. Grundy. (2014). Capturing security requirements using essential use cases (EUCs). In Requirements Engineering (pp. 16-30).
- [34] D. Wixon, J. Ramey. (1996). Field Methods Casebook for Software Design. John Wiley & Sons.
- [35] M. Hammersley, P. Atkinson. (2007) Ethnography: Principles in Practice. Routledge.
- [36] L.L. Constantine, L.A. Lockwood,. (2001). Structure and style in use cases for user interface design. Object modeling and user interface design, 245-280.
- [37] J. Nielsen, H. Loranger (2006). Prioritizing Web Usability. New Riders Press, 1 edition.
- [38] J. Annett. (2004). Hierarchical Task Analysis. Task Analysis for Human-Computer Interaction. D. Diaper and N. Stanton (eds.). Laurence Erlbaum Associates, Mahwah, USA, 67-82.
- [39] D. Diaper, N. Stanton. (2003). The handbook of task analysis for human-computer interaction. CRC Press.

- [40] D. Hix, H.R. Hartson. (1993). *Developing User Interfaces: Ensuring Usability through Product and Process*. John Wiley and Sons, New York, USA.
- [41] A. Holzinger. (2005). Usability engineering methods for software developers. *Communications of the ACM*, 48(1), 71-74.
- [42] A. Goknil, I. Kurtev, K. van den Berg, W. Spijkerman. (2014). Change impact analysis for requirements: A metamodeling approach. *Information and Software Technology*, 56(8), 950-972.
- [43] J. Nielsen, R.L. Mack. (1994). *Usability Inspection Methods*. Wiley.
- [44] J. Borchers. (2001). *A Pattern Approach to Interaction Design*. Wiley.
- [45] User Interface Design Patterns (2017). UI Patterns. [ONLINE] Available at: [http:// http://ui-patterns.com/](http://ui-patterns.com/). [Accessed 16 December 2017]
- [46] A.I. Wasserman. (1985). Extending State Transition Diagrams for the Specification of Human-Computer Interaction. *IEEE Transactions on Software Engineering*, 11(8), 699-713.
- [47] J. Horrocks. (1999). *Constructing the user interface with statecharts*. Addison-Wesley Longman Publishing Co., Inc.
- [48] B. Shneiderman. (1998). *Designing the User Interface: Strategies for Effective Human-Computer Interaction*. Addison-Wesley, Reading, USA.
- [49] T. Le, S. Chaudhuri, J. Chung, H.J. Thompson, G. Demiris. (2014). Tree testing of hierarchical menu structures for health applications. *Journal of biomedical informatics*, 49, 198-205.
- [50] D. Navarre, P. Palanque, J.F. Ladry, E. Barboni. (2009). ICOs: A model-based user interface description technique dedicated to interactive systems addressing usability, reliability and scalability. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 16(4), 18.
- [51] J.A. Macías. (2012). Enhancing Interaction Design on the Semantic Web: A Case Study. *IEEE Transactions on Systems Man and Cybernetics Part C – Applications and Reviews*. 42(6), 1365-1373.
- [52] K. Hornbæk, B.B. Bederson, C. Plaisant. (2002). Navigation patterns and usability of zoomable user interfaces with and without an overview. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 9(4), 362-389.
- [53] C. Snyder. (2003). *Paper Prototyping: the Fast and Easy Way to Design and Refine User Interfaces*. Morgan Kaufmann.
- [54] J.S. Dumas, J.C. Redish. (1999). *A Practical Guide to Usability Testing*. Intellect, Exeter, England.

- [55] A. Cooper, R. Reiman. (2003). *About Face 2.0: The Essential of Interaction Design*. Wiley Publishing, Indianapolis, USA.
- [56] J. Haikara. (2007). Usability in agile software development: extending the interaction design process with personas approach. *Agile processes in software engineering and extreme programming*, 153-156.
- [57] J. Sauro, J.S. Dumas. (2009). Comparison of three one-question, post-task usability questionnaires. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM, 1599-1608.
- [58] M.B. Rosson, J.M. Carroll. (2002). *Usability Engineering: Scenario-Based Development of HCI*. Morgan Kaufmann.
- [59] T. Boren, J. Ramey. (2000). Thinking aloud: reconciling theory and practice. *IEEE Trans. Prof. Commun.*, 43(3), 261–278.
- [60] L. Van Waes, (2000). Thinking aloud as a method for testing the usability of Websites: the influence of task variation on the evaluation of hypertext". *IEEE Transactions on Professional Communication*, 43(3), 279–291.
- [61] T. Tullis, S. Fleischman, M. McNulty, C. Cianchette, M. Bergel. (2002). An empirical comparison of lab and remote usability testing of web sites. In *Usability Professionals Association Conference*.
- [62] J.A. Macías, P. Castells. (2007). Providing End-user Facilities to Simplify Ontology-driven Web Application Authoring". *Interacting with Computers*, 19(4), 563-585.
- [63] C. LeRouge, J. Ma, S. Sneha, K. Tolle. (2013). User profiles and personas in the design and development of consumer health technologies. *International journal of medical informatics*, 82(11), e251-e268.
- [64] J. Preece, Y. Rogers, H. Sharp, D. Benyon, S. Holland, T. Carey. (1994). *Human-Computer Interaction*. Addison Wesley, England.
- [65] R. Van Der Lugt. (2002). Brainsketching and how it differs from brainstorming. *Creativity and innovation management*, 11(1), 43-54.
- [66] J. Nielsen 1995. Nielsen Norman Group. How to Conduct a Heuristic Evaluation. [ONLINE] Available at: <https://www.nngroup.com/articles/how-to-conduct-a-heuristic-evaluation/> [Accessed 16 December 2017]
- [67] A.M. Lund. (2001) Measuring Usability with the USE Questionnaire. *STC Usability SIG Newsletter*, 8-2.
- [68] A.d. Castro, J.A. Macías. SUSApp: A Mobile App for Measuring and Comparing Questionnaire-Based Usability Assessments. *Proceedings of the 17th International Conference on Human Computer Interaction (Interacción 2016)*. Salamanca, Spain, 14-16 September 2016.

Article No. 19. ICPS: ACM International Conference Proceedings Series. ACM New York, USA, 2016.

[69] T. Tullis, W. Albert. (2013). Measuring the User Experience. Morgan Kaufmann.

[70] G. Perlman. 2015. User Interface Usability Evaluation with Web-Based Questionnaires. USE Questionnaire. [ONLINE] Available at: <http://garyperlman.com/quest/quest.cgi?form=USE>. [Accessed 16 December 2017].

[71] J. Sauro. 2017. MeasuringU. [ONLINE] Available at: <https://measuringu.com>. [Accessed 16 December 2017].

[72] A. Dix, J.E. Finlay, G.D. Abowd, R. Beale (2004). Human-Computer Interaction. Prentice-Hall.

[73] J. Nielsen, T.K. Landauer. A Mathematical Model of the Finding of Usability Problems. Proceedings of ACM INTERCHI'93 Conference. Amsterdam, The Netherlands, 24-29 April 1993, pp. 206-213.

[74] J. Sauro. (2010). A Practical Guide to Measuring Usability. Sauro.

[75] W. Hwang, G. Salvendy. (2010). Number of people required for usability evaluation: the 10 ± 2 rule. Communications of the ACM, 53(5), 130-133.

[76] L.A. Hasan, K.T. Al-Sarayreh. An Integrated Measurement Model for Evaluating Usability Attributes. IPAC '15 Proceedings of the International Conference on Intelligent Information Processing, Security and Advanced Communication Article No. 94. Batna, Algeria — November 23 - 25, 2015.