Available online at www.sciencedirect.com
ScienceDirect
Information and Software Technology xxx (2007) xxx–xxx
INFORMATION AND SOFTWARE TECHNOLOGY
www.elsevier.com/locate/infsof

Enhancing usability testing through datamining techniques: A novel approach to detecting usability problem patterns for a context of use

María Paula González [a,b,c,*], Jesús Lorés [a,1], Antoni Granollers [c]

[a] National Council of Scientific and Technical Research (CONICET), Argentina
[b] Visualization and Graphics Research Laboratory, VyGLab, Department of Computer Science and Engineering – Universidad Nacional del Sur, Av. Alem 1253–8000 Bahía Blanca, Argentina
[c] Group GRIHO – Universitat de Lleida – clJaume II, 69–25001Lérida, Spain

Received 22 September 2006; received in revised form 13 May 2007; accepted 5 June 2007

Abstract

Usability is a software attribute usually associated with the "ease of use and to learn" of a given interactive system. Nowadays usability evaluation is becoming an important part of software development, providing results based on quantitative and qualitative estimations. In this context, qualitative results are usually obtained through a *Qualitative Usability Testing* process which includes a number of different methods focused on analyzing the interface of a particular interactive system. These methods become complex when a large number of interactive systems belonging to the same context of use have to be jointly considered to provide a general diagnosis, as a considerable amount of information must be visualized and treated simultaneously. However, diagnosing the most general usability problems of a context of use as a whole from a qualitative viewpoint is a challenge for *UE* nowadays. Identifying such problems can help to evaluate a new interface belonging to this context, and to prevent usability errors when a novel interactive system is being developed. From a quantitative viewpoint, condensing results in singles scores, metrics or statistical functions is an acceptable solution for processing huge amounts of usability related information. Nevertheless, *QUT* processes need to keep their richness by prioritizing the "what" over the "how much/how many" questions related to the detection of usability problems.

To cope with the above situation, this paper presents a new approach in which two datamining techniques (association rules and decision trees) are used to extend the existing Qualitative Usability Testing process in order to provide a general usability diagnosis of a given context of use from a qualitative viewpoint. In order to validate our proposal, usability problems patterns belonging to academic webpages in Spanish-speaking countries are assessed by processing 3450 records which store qualitative information collected by means of a Heuristic Evaluation.

Keywords: Usability engineering; Qualitative usability testing; Context of use; Usability problem patterns; Datamining; Association rules; Decision trees

1. Introduction

Usability is a software attribute usually associated with the "ease of use and to learn" of a given interactive system [19], and largely recognized in the literature as "the extent to which a product can be used by specified users to achieve

* Corresponding author. Address: Visualization and Graphics Research Laboratory, VyGLab, Department of Computer Science and Engineering – Universidad Nacional del Sur, Av. Alem 1253–8000 Bahía Blanca, Argentina.
E-mail addresses: mpg@cs.uns.edu.ar, mpg@diei.udl.cat (M.P. González), tonig@cs.uns.edu.ar (A. Granollers).
[1] In memoriam.

0950-5849/$ - see front matter © 2007 Published by Elsevier B.V.
doi:10.1016/j.infsof.2007.06.001

specified goals with effectiveness, efficiency and satisfaction in a specified context of use'' [1].[2] In this setting, the notion of *context of use* stands for a description of the actual conditions under which the interactive system is being assessed, or will be used in a normal working situation. Developing a highly usable interactive system[3] is a complex task. Development teams have at their disposal methodologies from the area of *Usability Engineering* (*UE*) [49], which is defined as ''the systematic approach to improving the usability of user interfaces by applying a set of proven methods throughout the system development lifecycle'' [51].

The key principles of *UE* models are based on *incremental prototyping*, where usability goals are prioritized [44]. In other words, developing a software product under a *UE* model involves a number of intermediate stages that are performed cyclically on the basis of an active user participation. In particular, an *Evaluation Stage* is usually included, where usability estimation plays a major role. This Evaluation Stage is carried out by an *evaluation team* in charge of assessing the results obtained from applying *usability testing processes* on the existing system or prototype. In this respect, the Evaluation Stage allows to introduce critical changes in software development, enabling designers and developers to incorporate user feedback until an acceptable level of usability is reached.

In the above setting, two kinds of complementary usability results are obtained. On the one hand, a *quantitative usability estimation* is typically associated with the calculation of metrics that assess some factors or dimensions of software quality [19,37,62]. On the other hand, a *qualitative usability estimation* is also normally included, since no quantitative measure can be expressive enough to represent something so complex as the overall usability of a software problem or a user desire [14,19,49,37,18]. Indeed, this qualitative usability estimation often generates better insight, and is typically carried out through a process that we will call *Qualitative Usability Testing* (*QUT*). To be effective, the *QUT* process cannot return just a number or ''yes/no'' as an answer (e.g. ''the interface is not usable'' or ''the system achieved a 75% of usability''). Instead, detailed information must be provided about why the design of a given system *S* did not work as anticipated and which will be the main usability problems that real users will find when interacting with *S*. It is also expected that the *QUT* process be able to prioritize the problems identified. Thus, the main goal of the *QUT* process is to produce a *usability report* including a prioritized list of usability problems for *S*. The *QUT* process will be based on the application of different *usability testing methods* (such as Heuristic Evaluation or Cognitive Walkthrough [49]).

Most *QUT* methods are focused on analyzing the interface of a particular interactive system (see e.g. [11,19,37,49]). Even when such methods can be successfully applied for evaluating individual cases, they become complex when a large number of systems belonging to the same context of use have to be jointly considered to provide a general diagnosis. The main reason for this is due to the large amount of qualitative information that must be processed and visualized simultaneously. However, diagnosing the most general usability problems of a context of use as a whole from a qualitative viewpoint is a challenge for *UE* nowadays. Identifying such problems can help to evaluate a new interface belonging to this context, and to prevent usability errors when a novel interactive system is being developed. From a quantitative viewpoint, condensing results in singles scores, metrics or statistical functions is an acceptable solution for processing huge amounts of usability related information. Nevertheless, *QUT* processes need to keep their richness by prioritizing the ''what'' over the ''how much/how many'' questions related to the detection of usability problems.

This paper presents a novel approach called QUT_C, which empowers the traditional *QUT* process by extending it through data processing and data mining techniques coming from *Knowledge Discovery in Databases* (*KDD*) [32,46]. Starting with the data collected by applying a *QUT* process to a sample group of interactive systems (all of them belonging to a given context of use *C*), our goal is to obtain a general usability diagnosis of *C*. This usability diagnosis will be based on the detection of usability problem patterns of *C*. Each of the patterns obtained will describe a relevant usability feature from a qualitative viewpoint. By using this extended approach, queries posed by the evaluation team will be associated with detecting hidden relationships in the data collected from a traditional *QUT* process. Answers from queries will be computed and ranked through *KDD* techniques. In our approach we have focused on the application of two particular *KDD* techniques: *association rules* (as a descriptive technique that searches for interesting relationships among items in a given data set [32]) and *decision trees* (as a model for predictive usability problems). The final output will be a usability report of the context of use under consideration. This report will include a list of prioritized usability problems, each of them expressed in a standard format. In order to validate our proposal, we have carried out an experimentation in which a general usability diagnosis for the context of use of academic web sites in Spanish-speaking countries was assessed by applying the QUT_C process. The experimentation involved the processing of 3450 records which stored qualitative information coming from the application of a Heuristic Evaluation (*HE*).

The rest of this paper is structured as follows. First, in Section 2 we summarize the most relevant aspects of the traditional *QUT* process under *UE*. Then, Section 3 pre-

[2] This definition of usability is also used in the Common Industry Format (CIF) for usability testing [59].

[3] For the sake of simplicity in what follows we will assume that whenever we refer to a *system S* we are referring to either a prototype of *S* or a fully developed version of *S*, although we are aware of the differences between both notions. Note that any actual interactive software *S* can be always considered as an advanced prototype of the next, improved version *S'* of the system *S*.

Please cite this article in press as: M.P. González et al., Enhancing usability testing through datamining techniques: ..., Inform. Softw. Technol. (2007), doi:10.1016/j.infsof.2007.06.001

sents an overview of the main concepts related with *KDD* and the two particular datamining techniques mentioned before (association rules and decision trees). Section 4 presents our proposal for extending the traditional *QUT* process by incorporating *KDD* techniques to identify typical usability problems patterns of a particular context of use. Section 5 describes some experimental results which demonstrate the applicability of our integrated approach. Section 6 discusses related work, and Section 7 concludes by summarizing the main results that have been obtained and discussing future work. We want to remark that this article is based on some preliminary research work from the authors on modeling usability evaluation of early prototypes under User-Centered Design through association rules [26,27].

2. Qualitative usability testing process under usability engineering

During the last decade the *QUT* process has proven to be a crucial part of the evaluation stage in *UE* [19,37]. Under this perspective, different kinds of methods are normally used, each of them having its particular features. Moreover, these methods are usually classified in *inspection*, *testing* and *inquiry* methods according to the kind of activities involved [19,37,49]. Usability inspection [7,50,51,71] methods are based on the verification of usability principles or "guidelines". In such methods, usability experts examine and work with the interface of a given interactive system to detect principle violations in their design [61]. On the contrary, usability testing methods require the active presence of the final users, who provide usability related data to be analyzed by the evaluation team [19,49]. Indeed, the testing approach usually involves the evaluation of the system through the collection of data generated for representative users when working on general tasks. Sometimes users are asked to complement the collection of measurable data with their opinion. Including the participation of final users has proven to be successful to enhance inspection methods efficacy [28,29,47], thus bringing closer the inspection and testing activities. In usability inquiry methods [19,49,33,61] usability evaluators obtain information about users' preferences, needs and understanding of the system by talking to them, observing them as they use the system in real work, or letting them answer questions verbally or in written form.

The listings shown in Figs. 1 and 2 (based on [37]) depict the most common testing, inspection and inquiring methods used to perform *QUT* processes under *UE*. Note that this listing is not exhaustive, summarizing just those *QUT* methodologies which are most widely used. Indeed, some methods given in the original listing presented in [37] were omitted (as they are related to quantitative usability testing processes) and some other methods not mentioned in [37] were added, such as *Participatory Walkthrough, Retrospective Think-Aloud, Scenarios-Based Walkthrough, Cooperative Usability Testing, Cognitive Walkthrough for the Web*, etc. Additionally, every method in Figs. 1 and 2 has been linked to related bibliographical references. Note that some of these methods are not defined explicitly to perform *QUT* processes (e.g. *Focus Group* or *Thinking-Aloud* techniques). However, they are considered useful tools for carrying out such processes only if they are oriented towards testing usability [18]. In the same way, methods such as *Performance Measurement* or *Log File Analysis* could be more close to quantitative usability testing than to *QUT*. Nevertheless, as we will see in Section 4, their results can be also interpreted qualitatively by performing the approach proposed in this paper. In spite of the differences present among the methods in Figs. 1 and 2 [4] there exist a number of common elements that can be identified in any *QUT* process (see Fig. 3), namely:

- an *interactive system S* which is being assessed using the *QUT* process
- an *generic evaluation method M* for qualitative usability testing (selected among the possible methods listed in Figs. 1 and 2
- a *particularization* (or instance) M' of the method M, defined for applying M to the particular usability evaluation of the system S
- a dataset representing the *results* obtained from the application of M' when assessing the qualitative usability of the system S
- a number of different *visual representations* (visualization) of the above results, used by the evaluation team for facilitating their processing and analysis
- a final *usability report* containing the general results coming from the whole qualitative usability evaluation process

Next we will analyze the different steps related to applying a *QUT* process to a generic system *S*. We are aware that in actual software development projects some parts of these steps might overlap, as *QUT* is not always characterized in a uniform and linear manner within the existing models for software development in *UE*.

(1) *Planning the QUT process*. As a first step, a particular method *M* is selected by the evaluation team out of the listings of possible options shown in Figs. 1 and 2. This choice will depend on several aspects (the number of times this technique was already used, the cost associated with it, its adequacy to the system *S* at issue, etc.).[5] According to the particular features of the selected method *M*, sometimes a redefinition of the team should be done. This redefinition could involve the addition of some experts or users in the evaluation team. Besides, the conditions to perform

[4] For the purposes of this paper, we will restrict ourselves to the *QUT* methods given in Figs. 1 and 2 although we are aware that this listing is not exhaustive.

[5] In that respect we assume that the evaluation team keeps track of the techniques that have been used in previous Evaluation Stages during the lifecycle of the system or prototype at issue.

Method Name	Description	Citation
Testing Methods		
(*) Thinking-Aloud Protocol	user talks during test	[19,52]
RetrospectiveThinking-Aloud	user talk during test. Then this information is contrasted against other data coming from the test (e.g. eye movement)	[31]
Cooperative Usability Testing (CUT)	usability test is complemented with used-supported interpretation	[24]
(*) Shadowing Method	expert explains user actions to tester	[19]
(*) Coaching Method	user can ask an expert questions	[49]
(*) Teaching Method	expert user teaches novice user	[70]
(*) Codiscovery Learning	two users collaborate	[19,49]
(*) Log File Analysis	tester analyzes usage data	[2,49,68]
Self–Reporting Logs	paper–and–pencil journals in which users are requested to log their actions and observations while interacting with a product	[49]
(*) Retrospective Testing	tester reviews videotape with user	[49,61]
Scenario Building	users confront system in expected real-life usage to generate usability requirements. Focused on early stages of system evaluations	[5,61]
(*) Remote Testing	tester and user are not collocated during test	[63]
Journaled Sessions	kind of remote testing based on task performing. The software to be tested is available in a CD or similar, and it himself capture information about users actions.	[49]
Inspection Methods		
Feature Inspection	expert assessment products feature set within the context of a task. Emphasis on sequence of tasks, accessibility, potential for confusion, and documentation.	[50]
(*) Cognitive Walkthrough	expert simulates users problem solving	[71]
(*) Pluralistic Walkthrough	multiple experts conduct Cognitive Walkthrough	[7]
Participatory Walkthrough	Cognitive Walkthrough performed directly by users	[29]
Scenarios-Based Walkthrough	evaluator systematically describes user scenarios to attain the objectives of a Website and evaluates the user actions required to complete each scenario	[64]
Groupware Walkthrough	specialized cognitive wallktrough to assess groupware systems	[56]
(*) Heuristic Evaluation	expert identifies violations of heuristics	[50,53]
Participatory Heuristic Evaluation	expert and users identifies violations of heuristics	[47]
(*) Perspective-Based Inspection	expert conducts narrowly focused heuristic evaluation	[73]
(*) Formal Usability Inspection	expert conducts formal heuristic evaluation	[44,50]
Diagnostic evaluation	users run representative tasks through a variety of scenarios using the system in question. Output: list of usability problems, categorized by importance.	[19]
(*) Consistency Inspection	expert checks consistency across products	[50]
Metaphors of Human Thinking (MOT)	experts inspect the interface solving representative tasks to establish when it supports or violates the aspects of human thinking that the metaphors and key questions aim to capture.	[35]

Fig. 1. Most common testing and inspection methods to perform the *QUT* process. Those methods marked with (∗) are based on [37].

Please cite this article in press as: M.P. González et al., Enhancing usability testing through datamining techniques: ..., Inform. Softw. Technol. (2007), doi:10.1016/j.infsof.2007.06.001

Method Name	Description	Citation
Inquiry Methods		
Contextual Inquiry	interviewer questions users in their environment	[33,6]
Field Observation	interviewer observes system use in users environment	[49,57]
Focus Groups	multiple users participate in a discussion session	[14,30]
Interviews	one user participates in a discussion session	[57]
Surveys	interviewer asks user specific questions	[3]
Questionnaires	user provides answers to specific questions	ASQ [38] CSUQ [40] NAU [49] NHE [49] PHUE [55] PSSUQ [39] PUEU [16] PUTQ [41] QUIS [12] SUMI [69] SUS [9]
Self-Reporting Logs	user records operations when using the system	[49]
Screen Snapshots	user captures GUI screens	[49]
User Feedback	user submits comments from solving a task	[19,44]

Fig. 2. Most common inquiring methods to perform the *QUT* process. All methods are based on [37].

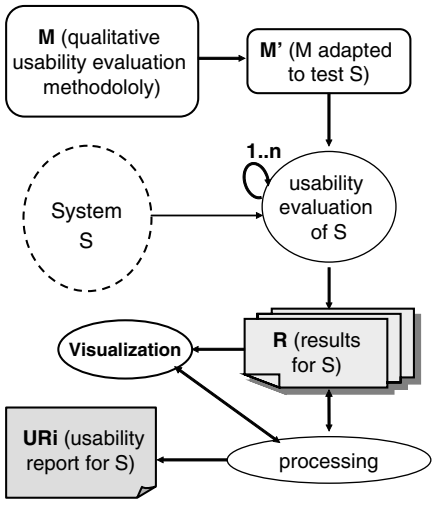

Fig. 3. Schema of the Qualitative Usability Evaluation (*QUT*) process for an interactive system *S*.

M (timetable, physical environment, equipment, etc.) should be clearly stated.

Note that most *QUT* methods are defined in a general way. Consequently, many times a specialization of *M* should be defined in order to adequate it to the particular *QUT* process in progress. As a result, an instance M' of the original method *M* is defined, including the selection of appropriate ideal values, and a more specific definition of the main features of *M* (e.g. the definition of task in the case of the Cognitive Walkthrough, or the re-writing of questions in the case of Questionnaires). Additionally, the use of specialized software may be required (as in the case of the Remote Testing). In such cases, part of the planning should include the selection of a particular software platform. Mostly this selection will be depend on the associated costs and the available resources.

(2) *Carrying out the QUT process.* This step materializes the real performance of the method M' on the basis of the decisions made during the previous Planning step. If the method M' requires the use of a specialized software, it will be necessary to run it with the adequate parameters. In any case, the members of the evaluation team will apply the method M' to test the system *S* under evaluation, carrying out actions such as task completion, time capture, collecting usability data, etc. Depending on the features of M', the process will be carried out different numbers of times. It must be stressed that due to the nature of the *QUT* methods, qualitative information (often alphabetic data) is collected and complemented with natural language explanation of its significance.

(3) *Processing and analyzing QUT results.* A crucial part of the *QUT* process is the processing and the analysis of the results obtained in the previous step. In the worst case, the final usability problem set is created just by compiling the data obtained from applying the method M' without any other consideration. In an ideal situation, data coming from the application of M' are processed and analyzed to achieve a general diagnosis of the usability problems in *S*. To do this, the members of the evaluation team should compare those data in order to detect agreements and disagreements between the different applications of M' during the previous *QUT* step. Any usability problem should be carefully examined to decide if its relevance is strong enough. Frequently, this examination implies not only

the processing of qualitative information but also the consideration of natural language explanations that complement the alphanumeric data available. The use of different kinds of graphical representations (such as diagrams, concept maps, etc.) can enhance the visualization and processing of the obtained results.

(4) *Reporting QUT conclusions.* As an output, the *QUT* process returns a set of justified usability problems detected in the system *S* under evaluation. To report this set there are different standards and reporting sheets. For example, Fig. 4 shows a common format to report a usability problem that can be embedded in any standard usability report, such as the *Common Industry Format for Usability test report* (CIF) [59].

Interestingly, the above *QUT* process steps can be applied to evaluate not only the usability of a particular interactive system *S*, but also the usability of a *group* of interactive systems as a whole, all of them belonging to the same context of use. However, as explained in Section 1, processing and analyzing *QUT* results becomes complex when a large group of systems has to be jointly considered to provide a general diagnosis of this current context of use, since a considerable amount of qualitative information must be visualized and treated simultaneously. In this second situation, it would be desirable that the evaluation team be able to consider all such information with a reasonable cost and without losing the richness of the qualitative perspective. As we will see in the next sections, extending the characterization of the *QUT* process presented in this Section by integrating the traditional approach with some *KDD* techniques can help to cope with the above problem.

3. Knowledge discovering in databases: A brief overview

As stated in Section 1, *KDD* is a discipline which involves the nontrivial extraction of implicit, previously unknown, and potentially useful information from data [32]. *KDD* can be formalized as a process in which data coming from (possibly) heterogenous sources are normalized and combined into a transactional database, in which features or *attributes* are identified. The extraction can be modelled as a process in which different steps [32,58] can be identified:

- *Data preprocessing:* This first step condenses a number of intermediate steps required to prepare data for the mining task, such as data cleaning, integration, selection and transformation. Data cleaning routines attempt to fill in missing values, smooth out noise, and correct inconsistencies in the data. If required, multiple data sources may be combined in a single database, and additional selection and transformation operations may be performed to unify and consolidate data formats.
- *Datamining (DM):* Once data has been preprocessed and condensed in a database, datamining techniques can be applied in order to extract relevant patterns from it. There exist several datamining techniques, such as association rules, decision trees, Bayesian data analysis, neural network models, etc. (for a detailed discussion see [32,46]).
- *Pattern evaluation and knowledge representation:* Different criteria (e.g. interestingness measures) are applied to evaluate the patterns obtained as an output of the datamining algorithms. Finally, visualization and representation techniques are used to present the mined knowledge to the user.

Note that datamining is only one step in the entire process, albeit an essential one since it uncovers hidden patterns for evaluation. In order to detect these hidden patterns of information, datamining can be *descriptive* – identifying general properties of the data being analyzed – or *predictive* – building models upon the data available. Based on the previous steps, a typical software platform for *KDD* would involve the following major components (see Fig. 5): (a) a database or information repository, consisting of one or a set of databases, spreadsheets, or other kinds of information repositories where data is stored; (b) a *KDD* engine, which consists of a set of functional modules for accomplishing the datamining tasks described before. The *KDD* engine provides a number of datamining algorithms, as well as modules for pattern evaluation and visualization (by means of a graphical interface); (c) a front-end module which allows the final user to interact with the *KDD* engine,

Problem	Brief statement of the problem (Problem are numbered for easy reference)
Scope	Global or local
Severity Level	The appropriate number and what it means
Frequency	How often the problem occurred and how many participants had the problem
Explanation	A paragraph or two of information about the problem, including examples of it, further supporting data, perhaps users' comments.
Recommendations	A bulleted or numbered list of ways to solve the problem

Fig. 4. A sample organization and page layout for reporting usability problems [19].

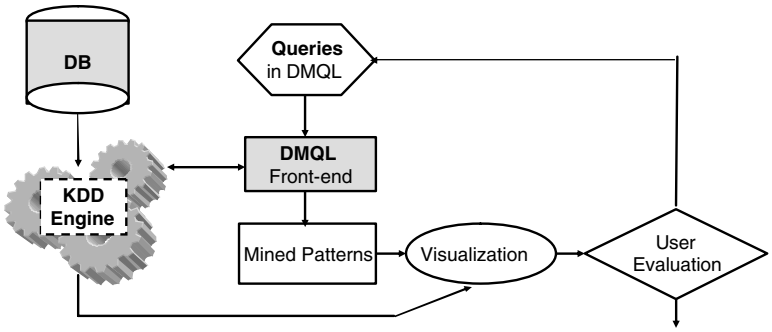

Fig. 5. Major components in a *KDD* software platform.

typically by means of a specialized *datamining query language* (DMQL), which covers a wide spectrum of tasks, such as data characterization for mining association rules, data classification, computation of decision trees, etc. [32]. Different general-purpose platforms like WEKA [72], Orange [17] or DBMiner [32] have been developed to carry out the *KDD* process, each of them with their specific DMQL language. Such query-oriented languages range from very basic command-line interpreters (as the Command-line Interface provided by the WEKA platform [72]) up to more sophisticated tools such as specialized scripting languages (as the one provided by the module `orngMySQL` in the Orange platform [17]), MSQL [36], OLE DB for Datamining [48], or more recently KDDML [60], among others).

As already stated in the introduction, in this paper we present a new, extended approach to the traditional *QUT* process by incorporating two particular datamining techniques (association rules and decision trees). In order to make this article self-contained, we will summarize next some of their main features.

3.1. Association rules

Association rule mining finds interesting relationships among items in a given data set [32]. The discovery of such relationships among transaction records is typically used in many business decision making processes, such as catalog design and cross-marketing. A typical example of association rule mining is the so-called *market basket analysis*, where a company analyzes customer buying habits by finding associations between the different items that customers place in their "shopping baskets". The discovery of such associations can help retailers to develop marketing strategies in which they identify which items are frequently purchased together by customers.

Fig. 6 shows a very simple example of a transactional database D onto which association rule mining can be applied. In such databases we identify a set I of *items* involved (for this example, I={milk, cheese, beer, fries, hot-dogs}). An *association rule* is an implication of the form $A \Rightarrow B$, where $A \subset I$, $B \subset I$, and $A \cap B = \emptyset$. Two basic concepts are used to measure the interestingness of an association rule, namely *support* and *confidence*. The rule $A \Rightarrow B$ is said to hold in D with *support* s if s is the percentage of transactions in D that contain $A \cup B$, i.e. the probability $P(A \cup B)$. The rule $A \Rightarrow B$ has *confidence* c if c is the percentage of transactions in D containing A that also contain B, i.e. $P(B|A)$. Thus, confidence c for a rule $A \Rightarrow B$ can be calculated as $Support(A \cup B)/Support(A)$. Fig. 6 shows two possible association rules that can be obtained from the database D. From this example we can see how association rules allow us to identify patterns in datasets which suggest hypotheses (e.g. 100% of customers buying milk are also buying cheese). Clearly, real-world problems demand hundreds or thousands of records with different items. The main difficulties in the generation of association rules from large datasets involve the combinatorial explosion in the number of rules to be assessed. The solution for this problem is twofold: on the one hand, association rules are constrained by means of thresholds to be considered useful as mined knowledge (e.g. confidence should be greater than 95%). On the other hand, very efficient algorithms have been developing for mining association rules from large databases, notably APRIORI and FP-GROWTH [32]. Such algorithms reduce the computational complexity by relying on specified thresholds for confidence and support values. Additionally, powerful query primitives have been developed to post-process the generated rulebase, as well as for performing selective, query based generation as explained before.

Basket	Items
1	milk, cheese, beer, fries
2	milk, cheese, beer
3	milk, beer, fries, cheese
4	beer, fries
5	hot-dogs, fries

$beer \Rightarrow fries$ (Support=60%, Conf=75%)
$milk \Rightarrow cheese$ (Support=60%, Conf=100%)

Fig. 6. A sample Market basket represented as a transactional database (left) and some possible association rules (right).

Please cite this article in press as: M.P. González et al., Enhancing usability testing through datamining techniques: ..., Inform. Softw. Technol. (2007), doi:10.1016/j.infsof.2007.06.001

3.2. Decision trees

Decision trees are flow-chart-like tree structures, where each internal node denotes a test on an attribute value, each branch represents an outcome of the test, and tree leaves represent classes or class distribution [46,72]. Decision trees classify instances by sorting them down the tree from the root to some leaf node, which provides the classification of the instance. Each node in the tree specifies a test of some attribute of the instance, and each branch descending from that node corresponds to one of the possible values for this attribute. In order to classify an unknown sample, the attribute values of the sample are tested against the decision tree. A path is traced from the root to a leaf node that holds the class prediction for that sample. Fig. 7 (left) illustrates a typical learned decision tree. This decision tree classifies Saturday mornings according to whether they are suitable for playing tennis. Thus, the instance (*Outlook* = *sunny*, *Temperature* = *hot*, *Humidity* = *high*, *Wind* = *Strong*) would be sorted down the leftmost branch of the tree and would therefore be classified as a negative instance (i.e. the tree would be predicting *PlayTennis* = *no*). Decision trees can also be depicted alternatively in text format as shown in Fig. 7 (right).

Decision trees can be automatically generated from databases in which different attributes are identified, as well as some particular "target attribute" (which corresponds to the class to be predicted by the tree). Typical algorithms used for generating decision trees are ID3 and C4.5 [46]. Such algorithms learn decision trees by constructing them top-down, identifying which is the best attribute to be selected for classifying the instances of the dataset in different classes (according to the target attribute chosen). Fig. 8 shows a sample database associated with the behavior of people playing tennis Saturday mornings [46]. From this database, by applying the ID3 algorithm, the tree shown in Fig. 7 can be automatically obtained.

When building decision trees, two disjoint subsets from the dataset can be distinguished: *training data* (used to build the decision tree) and *test data* (used to validate the tree that has been obtained). Validation allows to determine the proportion of instances from the test set that have been successfully predicted. Usually 2/3 of the original dataset are used for training, whereas the remaining 1/3 is used for testing [46,72]. Most *KDD* platforms (such as WEKA) integrate the training and validation processes within the same interface, giving the user facilities to choose among several possible options. It must be remarked that decision trees can be obtained even from relatively small databases. In fact, excessive amount of data used for learning decision trees leads to the problem known as "overfitting" [46,58]. This occurs with noise and correlations in the training set that are not reflected in the data as a whole.

Day	Outlook	Temp.	Humid	Wind	PlayTennis?
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Fig. 8. Sample database with examples characterizing when to play tennis in Saturday mornings (target attribute: *PlayTennis*).

4. Incorporating KDD techniques into the *QUT* Process

In this Section, we propose a novel characterization of the *QUT* process based on the integration of the traditional approach described in Section 2 and *KDD* techniques. In our proposal, *KDD* is used to gather common usability features of a large group of systems $S_C = \{S_1 \ldots S_n\}$ (each S_i belonging to the same context of use C) in order to achieve a general usability diagnosis of C as a whole. In what follows we will call this new approach QUT_C. By performing the proposed QUT_C process, the members of the evaluation team will be able to formulate different *queries* to an underlying *KDD* engine, getting as an output additional, qualitative information that will condense usability problem patterns

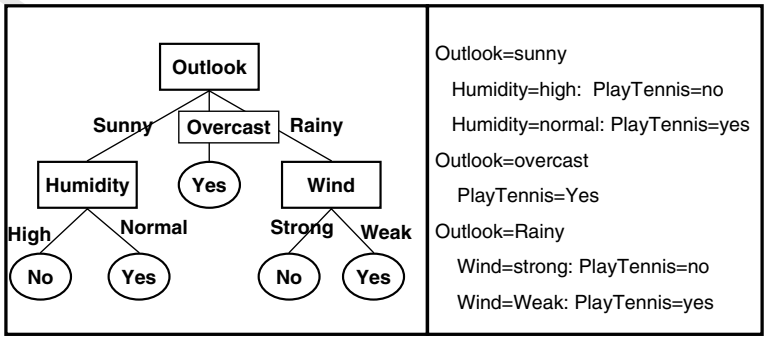

Fig. 7. A decision tree obtained by ID3 from the database shown in Fig. 8 (left) and the alternative representation of the same tree in textual form (right).

in C. The result of every query performed by the evaluation team will be expressed in terms of a *decision tree* or a ranked list of *association rules* where every association rule will express a (possibly hidden) relationship among usability features of S_C. Decision trees, on the other hand, will represent a predictive pattern derived from those usability problems of S_C. These trees will provide a model that can help to understand the usability features of C and to better assess the QUT evaluation of a new system S_{n+1} belonging to C that has to be tested under a UE perspective.

Fig. 9 shows a schematic view of the framework for performing the novel approach QUT_C. Following the UE principles, our proposal relies on different data collected during the existing QUT process and automatically compiled as a transactional database onto which KDD techniques can be applied. In fact, our proposal is based on extending the traditional QUT approach by adding the following elements:

- a *transactional database DB* which stores all the results obtained from applying the traditional QUT process to a group of systems S_C.
- a *KDD engine* (such as those present in the platforms described in Section 3) capable of automatically computing association rules and decision trees.
- a *DMQL Front-end* for the *KDD* engine, which should provide the interface for the evaluation team in order to pose queries (using the DMQL associated with the *KDD* platform) to detect usability problem patterns which will be given as an output of the *KDD* engine. In our case, for every query made by the evaluation team these patterns will be represented as a decision tree or as a ranked list of association rules.
- a *visualization module* (in most cases provided by the *KDD* engine itself) which provides different alternative graphical representations of the patterns found (e.g. tree-like charts for decision trees as those provided by WEKA [72] or 3-D representations for association rules as provided by DBMiner [32]).
- a *collection of usability problems patterns* related with the context of use C. Every pattern in this collection will be either a decision tree or a ranked list of association rules.

As before, we will analyze next the steps related to applying a QUT_C process to a given context of use C:

(1) *Planning the QUT_C process*. As in the QUT process described in Section 2, the QUT_C process will start with the selection of a particular method M by the evaluation team. A specialized method M' to focus M on C may be required as already indicated in the Planning Step of the traditional QUT process (see Section 2). This may required the possible redefinition of the evaluation team on the basis of M'. Besides, if M' involves the use of specialized software, the selection of a particular software platform should be also included.

Clearly, the ideal QUT_C process for the context of use C should involve an exhaustive usability evaluation of all systems in C to ensure a complete vision of such context. However, with the exception of some particular situations, this evaluation is not practically feasible for two main reasons. First, the cost associated with this exhaustive usability evaluation becomes prohibitive if the context of use includes several systems (e.g. it is practically impossible to consider all the systems belonging to the context of use of electronic banking). Second, in most cases the definition of a context of use is not static but rather dynamic, as

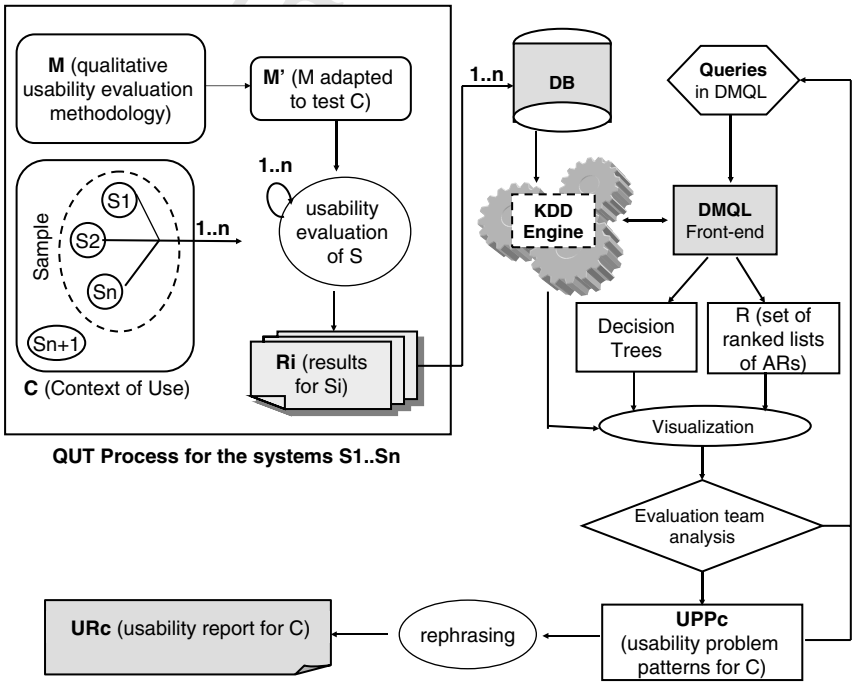

Fig. 9. Schema of the proposed Qualitative Usability Evaluation (QUT_C) Process for a context of use C of interactive systems.

the context changes whenever new systems are developed or existing ones are modified. Consequently, a sample group of systems $S_C = \{S_1 \ldots S_n\}$ must be selected to represent the full context C. Choosing this sample group is not a trivial task, and the use of a random-based criterion or another selection criteria will depend on the nature of C. Respecting how many systems must be included in S_C to ensure that they provide a representative sample of C, several possible guidelines can be adopted, as those given in [58].

Planning a QUT_C process will also include the selection of a particular software platform and the associated DMQL to carry out the *KDD* process. As before, also the conditions to perform M' (timetable, physical environment, equipment, etc.) should be clearly stated. Additionally, the conditions to perform the *KDD* process (software requirements, equipment, etc.) must be considered.

(2) *Collecting the QUT_C data*. The QUT_C process relies on different qualitative usability data collected during the application of the same *QUT* process to S_C. Thus, performing the method M' within all the elements in S_C is a central task. For the sake of simplicity, we are assuming that only one evaluation team will carry out the usability evaluation of every system in S_C. However, probably more than one evaluation team can be involved in such evaluation without invalidating the methodology, as shown in [20]. As the evaluation team carries out the usability evaluation of the elements in S_C, the obtained results (qualitative data) must be collected in an information repository (a database, a spreadsheet, a data warehouse, etc.). Note that this repository can be used later on to obtain statistical information that could complement results coming from the QUT_C process as discussed in Section 1.

(3) *Applying KDD techniques within the QUT_C process.* Once the qualitative data related to usability problems has been collected, the next step in the QUT_C process will consist in discovering patterns in that data by means of *KDD* techniques. To do this it will be necessary to automatically transform the data repository obtained in the previous step in a transactional database [58]. The following steps will be carried out, as described in Section 3:

- *Preprocessing the qualitative usability data*. As pointed out in [58], there does not exist a fully automatic system capable to preprocess the data to clean and integrate them. In the case of the QUT_C process cleaning and integrating the data is worthwhile, as the result can be used later on to obtain statistical information as explained in the previous step (Collecting the data). Note that data integration will be necessary only if more than one transactional database is generated (i.e. in the case of multiple teams performing M').
- *Mining the data to obtain usability problem patterns.* Once the data has been treated as described before, the evaluation team can use the selected *KDD* software platform and the associated DMQL in order to pose queries using the DMQL Front-end. In our approach the evaluation team can pose queries oriented towards obtaining two kinds of models for patterns: association rules and decision trees. Association rules will express (possibly hidden) relationships among attributes present in the qualitative information of DB, whereas decision trees will represent a predictive usability pattern characterizing the qualitative usability behavior of the context of use C.

In a first stage, the evaluation team will post a starting query just to detect the most relevant relationships (expressed as association rules) among the attributes in the database DB on the basis of a threshold value for support, confidence and a maximum number of rules to be computed. As a result, a ranked list R_1 of association rules will be automatically obtained, containing the most relevant information (with possibly hidden relationships) that needed to be considered by the evaluation team. By performing the next step (*Visualizing and analyzing QUT_C results*) the evaluation team will be able to observe and analyze the different attributes and relationships present in the association rules of R_1. In the case of decision trees, queries will include the target value. By default, 2/3 of the available data in DB will be randomly assigned to the training set, whereas the rest of the data will be used as the test set. However, as explained in the next step of the QUT_C process, this proportion can be modified by the evaluation team if required. It must be remarked that, according to our experiments, the computation of decision trees in this first stage does not make much sense, as at the beginning of the *KDD* process any of the attributes present in DB has the same chance to be considered a target attribute.

Next, on the basis of information available in R_1, the evaluation team can pose more refined queries expressed in a MSQL language by means of the DMQL Front-end. This time the aim will be to deepen the analysis of the information collected in the database DB. Thus, each new query for mining association rules will produce additional and more focused information, characterized as new ranked lists R_2, $R_3, \ldots$, R_k. Note that all these ranked lists are available to the Evaluation Team. For example, given the context of use of some electronic banking and a database BANK_EVALUATION with the information about the usability evaluation of the website of 50 banks, the following query in MSQL language can be performed:

```
GetRules(BANK_EVALUATION) where
  [Antecedent has{(profession=*)and... and age > 30}
  and Consequent has{error_type=*}
  and support >0.8 and confidence >0.9]
```

The outcome of this query will be a ranked list of usability problem patterns showing relationships between the profession of those users who are more than 30 years old and the most common errors made by them when browsing

a bank website. In the same way, the following query in the WEKA Command Line language:

```
java weka.classifiers.trees.ID3 -t BANK_EVALUATION.xls
-c 7 -d r.txt
```

will generate a decision tree (given as an output in the file `r.txt`) using the ID3 algorithm, the file BANK_EVALUATION.xls as the database DB, and the seventh attribute in the file BANK_EVALUATION.xls (the attribute error_type) as the target attribute. This tree will allow the evaluation team to characterize a usability problem for the context of use of the electronic banking based on predicting the behavior of the attribute error_type.

(4) *Visualizing and Analyzing QUT_C results.* As in the usual *QUT* process, an important part of the QUT_C process will involve the analysis of the usability problem patterns obtained in the previous step. The use of the visualization tools provided for most *KDD* platforms (such as WEKA [72] or Orange [17]) can help members of the evaluation team to observe different graphics and charts representing the obtained patterns. For the elements in ranked lists, results could be displayed as a sequence of association rules (each of them with the format described in Section 3.1). Additionally, a number of Cartesian charts can be generated, each of them depicting relationships between different attributes and their values present in the input database. Decision trees are so expressive by themselves that usually their tree-chart representation suffices. Due to the possibly interdisciplinary nature of the evaluation team, having different alternatives for visualizing the usability problem patterns detected above will promote a better understanding of them.

The goal of the evaluation team analysis will be to decide whether every pattern found depicts a usability problem of *C*. Indeed, each detected pattern will be discussed on the basis of its relevance and significance. For example, the relative position of every association rule in a ranked list expresses these features, jointly with its interestingness (support and confidence). Note that the relative position of an association rule with respect to the general ranking obtained (i.e. its position in the ranked list) is directly related to the prioritization of the usability problem pattern depicted by the rule. Moreover, metrics such as support and confidence guarantee that only significant patterns will be obtained, according to the criteria of the usability evaluation team. For decision trees, the validation process mentioned in Section 3.2 is used in the QUT_C process to determine whether a particular tree is valid or not. Indeed, the members of the evaluation team can adopt a minimal threshold value for accepting each decision tree (e.g. taking into account only those trees with at least 80% of correct predictions). The selection of the minimal threshold will depend both on the context of use under consideration and the criteria adopted by members of the evaluation team. Those trees which are acceptable will be included in the final usability report. In the case of a tree which is not accepted, members of the evaluation team can come back to the previous step in the QUT_C process to redefine the criteria for building the training and test set in the particular query used to generate the tree.

(5) *Reporting QUT_C conclusions.* The QUT_C process returns as an output a set of justified usability problems patterns detected for the current context of use *C*. It is desirable to rephrase each pattern as a usability problem expressed in some standard format similar to the one shown in Fig. 4. If necessary, also the different visualizations of the usability problem patterns generated during the QUT_C process can be added to the final output using this format.

5. Experimentation and discussion

In this Section, we will summarize the results obtained after using the proposed approach to evaluate the usability of the context of use formed by academic web sites in Spanish-speaking countries.[6] In what follows we will refer to this context as C_{UsP}. Note that C_{UsP} involves hundreds of millions of potential users (persons who speak Spanish either natively or by adoption) whose cultural background is primarily associated with the Spanish language and culture, regardless of ethnic and geographical differences. In particular, the QUT_C process was carried out by processing qualitative information stored in 3450 records which were collected by means of a Heuristic Evaluation (*HE*). This experimentation was carried out within the Second Stage of the *UsabAIPO* Project (*UsP*),[7] a project focused on usability research which involved the participation of more than 15 university research groups specialized in Human–Computer Interaction. In *UsP* four different categories were considered, namely *Design Category*,*Content Category*, *Navigation Category* and *Search Category*. Besides, during the First Stage of the *UsP*, the partners of the project decided to consider the web sites of all universities listed in the *Universia* portal[8] (69 universities in total) as the sample which will represent C_{UsP} in every stage of *UsP*.

Concerning the Second Stage of *UsP*, the evaluation team was formed by three usability experts and two Computer Science advanced students with solid knowledge about *UE* and usability. Two of the experts were university professors and the other was a PhD student. All members were also frequent users of academic web sites in Spanish-speaking countries. First, as established within the *UsP* Project, the web sites of the 69 universities listed in the *Universia* portal were considered as the sample of C_{UsP} to be

[6] The Spanish-speaking countries are formed by Spain and all Spanish-speaking countries in the Americas, including almost 35 million people living in USA with Spanish-related cultural origins.

[7] See www.aipo.es (webpage of the Asociación Interacción Persona Ordenador – AIPO)

[8] The *Universia* portal is a widely used official portal about universities available for Spanish-speaking countries. See www.universia.es

used. Second, *HE* [49] was selected as the evaluation method M to be applied. In order to define M' (a specialization of M), two general heuristic principles for each of the above categories were defined on the basis of 300 answers corresponding to a poll carried among users belonging to C_{UsP} and results coming from the *UsP* First Stage [42] (in which the usability of the homepages in C_{UsP} was assessed by means of a *HE* specialized for homepages). Each general heuristic principle was decomposed in several heuristic-related questions, all of them related to features to be tested. As a result, 25 questions were defined (from question #1 to question #25), (see Fig. 10). Ranges and ideal values were defined for each possible heuristic-related answer. As an example, Figs. 11 and 12 summarize the heuristic-related questions corresponding to the Design Category.

Since there was no suitable software package available for the application of the method M' (which involves the jointly *HE* of the 69 selected webpages on the basis of the 25 heuristic-related questions mentioned before), we developed a specialized software tool called *UsabAIPO Heuristic Management System* (*UHMS*). Fig. 13 shows a *UHMS* screenshot. This tool was mainly intended to support the usability evaluation of the 69 selected websites on the basis of the 25 heuristic-related questions that were defined. Indeed, the *UHMS* provides a simultaneous visualization of the current website that is being evaluated, the current heuristic-related question that needs to be answered (including possible answers), its significance within the website being considered and a pull-down menu to carry out the evaluation. To do this, every heuristic-related question was linked with an attribute in an output spreadsheet called *Usability_Universities*, which stores the result of the evaluation (as an example, see the correspondence between heuristic-related questions concerning the Design category and attributes in *UHMS* in Fig. 15). Every row in *Usability_Universities* represents the whole usability evaluation of a particular university. Besides, the tool provides a special form (see Fig. 14) that allows the evaluator to add his/her comments expressed in natural language, which will be automatically stored as part of the answer. When a user introduces some comment in *UHMS*, a pair of the form `<Attribute_Name: evaluator_comment>` is added to the last column of *Usability_Universities* in the row associated with the university under evaluation. If necessary, different comments related to different Heuristic-related questions can be concatenated in the same row, separated by a slash ("/"). In order to perform the *KDD* tasks required for our experimentation the WEKA platform [72] was chosen, as it provides the implementation of the datamining techniques that are needed (induction of decision trees and mining of association rules) along with an embedded Front-End with a very simple Command-line interface for posing queries and a visualization module to improve the interpretation of final results.

Next, the evaluation team was divided in two sub-teams in order to carry out two independent usability evaluation of the 69 websites considered by applying the method M' described before. The browsers MS *Internet Explorer 6.0*, *Mozilla Firefox 5.0* and *Netscape 8.0.1* were used to visualize the websites. The strategy used for each *HE* execution included a first overview of the website under evaluation for about 10 min, followed by an in-depth evaluation supported by the *UHMS*. As a result, each sub-team obtained 1725 qualitative answers (25 heuristic-related questions applied to assess 69 websites). that were stored in two temporary spreadsheets *Usability_Universities*_1 and *Usability_Universities*_2. During the preprocessing of the data stored in the two spreadsheets, each evaluation sub-team controlled the data produced by the other sub-team. Afterwards both spreadsheets were condensed in a final spreadsheet called *Usability_Universities*, which contained 3450 records automatically generated by the *UHMS*. As an example, a partial view of the sheet corresponding to the Design Category in *Usability_Universities* is shown in Fig. 16. Except for the comment column, all the rest of the spreadsheet was automatically transformed into the Attribute-Relation File Format (`arff`)[9] suitable for being processed by WEKA. Note that with the usual *QUT* process, a throughout analysis of all these answers from a qualitative viewpoint is not a feasible task, and going beyond the generation of some pie- or bar-charts is hardly tractable for the evaluation team. At this stage of the experimentation, the evaluation team was able to apply the proposed approach to process and analyze the information stored in *Usability_Universities* (the qualitative answers coming form M').

Different kinds of queries were posted by the evaluation team to mine the data stored in *Usability_Universities*. First, a general ranked list of association rules for each category of the UsabAIPO Project was computed using the Apriori algorithm provided by WEKA (with a support of 60%, a confidence of 90% and by selecting the best fifteen rules obtained). As an example, Fig. 17 shows the ranked list L_1 obtained corresponding to the Design Category. Note that the rules in L_1 provide a first guideline for identifying which attributes were involved in the discovered patterns.

In order to focus the detection of usability problems patterns within the context of use under evaluation, only those association rules containing some attribute whose value denoted a usability problem were considered. These attributes were tagged as *"problematic attributes"*. Besides association rules expressing similar usability problems were jointly considered, taking into account the possibility of involving one association rule in more than one usability problem pattern group (if the rule had more than one attribute with values denoting usability problems). For example, the association rules #1, #2, #3, #5, #12 and #15 in L_1 showed that, according to the information stored in the spreadsheet *Usability_Universities*, there exist usability

[9] See http://www.cs.waikato.ac.nz/ml/weka/arff.html

Cat.	Heuristic	Related questions
D	Graphic Design	- *Friendly Interface:* Does the website have a friendly interface, with uniform colors in most pages, matching with the university corporate image? -*Clean Interface:* Does the website offer a clean interface, without visual noise and with a correct usage of space? - *Text Design:* Does the text have a simple design, with enough contrast between background and text, limiting the font type and other text attributes? - *Liquid Design:* Is a liquid design being used?
	Images areas	- *Labelled Images:* Are images labelled? Does their title appear when the mouse is moved over them? - *Animated elements:* do animated elements exist? - *Image Resolution:* Has image resolution been taken into account, so that images are not pixeled and have an appropriate size for their correct visualization?
N	Navegation areas	-*Number of elements in menus:* Has the number of elements and terms per element been checked in order not to produce memory overload? - *Visibility:* Is the totality of the elements in the navigation area visible without requiring any interaction from the user? - *Existence of site map:* Is there a site map on the website? - *Affordance from links to applications:* If a link leads to an application, is that clearly indicated?
	Orientation	- *Access to homepage:* can the homepage always be reached from any navigation level? - *Orientation elements:* are there elements that allow the user to know exactly where he/she is in the website and how to move back? (breadcrumbs) - XIdenLinks
C	Information	- *News update:* Are academic news properly updated? Do they have publication date? - *Contact data:* Is it easy to access to the information in the different areas of the university (secretary, departments, etc.)? Is there clear information about the data needed to contact some particular area (phone, email, etc)? - *News clarity:* Are news published on a salient place on the website, with a link to the news associated with the news headline, and with a clear abstract of the contents of the news? - *Affordance from links to applications:* If a link leads to an application, is that clearly indicated?
	Internacionalization	- *Language:* Does the website offer a multilanguage option? - *Scope:* The academic information available for the different languages is one page or is it the most part of the website?
S	Search area	- *Visibility and Simplicity:* Is it easy to start a search? Is the textbox used to search terms on the website to be found on the homepage? Is it easily accesible from any place of the website? - *Size:* Is the text entry box for search wide enough? (it must contain between 15 and 30 visible characters) - *Complexity:* Is there an advanced search option? Are there enough options to carry out a properly bounded search ? - *Engines:* Are links to Internet search engines included?
	Search result	- *Comprensibility:* Are search results shown in a clear form which is comprehensible enough for the user? - *Assistance:* is the user assisted when no results can be found?

Fig. 10. Heuristic-related questions for the *UsabAIPO* Project (D, N, C and S corresponds to Design, Navigation, Content and Search Categories).

Heuristic-related Questions	Possible answers
Question #1 (Friendly Interface): does the website have an interface with uniform colors in most of its pages, strongly related with the corporative image the university has in the real world?	- *No*: the interface does not have a uniform color design in most of its pages, or the used colors are not related with the the corporative image the university has in the real world - *Sometimes*: only those pages belonging to the intermediate hierarchy of the website map do not have an interface with a uniform color design or are not related with the corporative image the university has in the real world - *Yes*: the interface colors are coherent with the corporative image of the university in the whole website
Question #2 (Clean Interface): does the website offer a balanced use of the screen space, without "visual noise" such as unnecessary and/or unpleasant elements, excessive use of blank space, unnecessary decorative images, excessive ornamentation or elements in movement which distract the user attention without any relevant purpose?	- *No*: most of the pages in the interface present some "visual noise", disturbing the user attention - *Sometimes*: there exist some "visual noise" in less than the half of the whole website, and this "visual noise" is not critical for disturbing user attention - *Yes*: the whole interface does not have "visual noise" elements
Question #3 (Text Design): is the text designed to facilitate its reading, with sufficient contrast between color text and background, and less than five different format styles (color, size, underlining, bold, italic, line spacing, etc.)?	- *No contrast*: there exists relevant text without sufficient contrast - *Sometimes*: there are more than five different format styles for the text or abrupt changes in the text format when browsing the website - *Yes*: text in every page in the interface has sufficient contrast, and there are no more than five different format styles used in the whole interface. Besides, the text style remains similar between the different pages without presenting sudden changes when browsing the website.
Question #4 (Liquid Design): does the website interface automatically adapt itself to different screen resolutions and sizes?	- *No*: less than 30% of the interface supports liquid design - *Middle Level*: between 30% and 50% of the interface supports liquid design - *High level*: between 50% and 90% of the interface supports liquid design - *Yes*: more than 90% of the interface supports liquid design

Fig. 11. Heuristic-related questions concerning Graphical Design (Design Category).

problem patterns related with the absence of a liquid design on the websites of the context of use under evaluation (i.e. C_{UsP}).

Note that the usability problem patterns in rules #1, #2, #3, #5, #12 and #15 in L_1 not only pointed out which are the "problematic attributes" of *Usability_Universities* related with the absence of a liquid design (considered as a usability problem), but also allowed to detect the most relevant relationships among them. This way, in the future these patterns will provide valuable information for assessing the usability of a particular system which is not included on the *Universia* portal but belongs to the context under consideration. Indeed, these usability problem patterns highlighted (based on the correspondence shown in Fig. 15) the most relevant heuristic-related questions that should be considered when assessing the usability of S with respect to its liquid design). Moreover, these patterns can be used as guidelines when designing and developing a novel system S within the context of use of academic webpages in Spanish-speaking countries, since they reflect critical interface elements that influence or are influenced by the liquid design in this particular environment. Additionally, we want to remark that relationships on rules #1, #2, #3, #5, #12 and #15 in L_1 were non trivial and therefore not easily detectable for the evaluation team from the available information. Thus, the results obtained follow from the successful application of the *KDD*-based procedure when performing the proposed method M'.

On the basis of the "problematic attributes" present in the association rules and the evaluators' comments in natural language stored in the spreadsheet *Usability_Universities*, members of the evaluation team could post more specific queries to mine the data in the spreadsheet. The queries included different values for support and confidence according to the criteria established by the members of the evaluation team. As an example, the queries shown below were basically intended to analyze some "problematic attributes" present in the association rules of L_1.

```
GetRules (Usability_Universities) where
  [Antecedent has {FriendlyInterface=sometimes}
  and support > 0.8 and confidence > 0.9]}
```

Please cite this article in press as: M.P. González et al., Enhancing usability testing through datamining techniques: ..., Inform. Softw. Technol. (2007), doi:10.1016/j.infsof.2007.06.001

Heuristic-related Questions	Possible answers
Question #5 (Labeled Images): are the images consistently tagged and/or an appropriate title is displayed when passing with the mouse or keys over them?	- *No*: more than 50% of the images are not tagged or the associated label has no a clear relation with the image itself - *Sometimes*: between 50% and 80% of the images are tagged with consistent tags - *Yes*: more than 80% of the images are tagged with appropriate tags
Question #6 (Too many Animated Elements): there exist animated images or elements in the website?	- *No*: there are no animated images or elements in the interface (ideal answer) - *Homepage*: only the homepage of the website presents less than two animated elements that do not disturb excessively the user attention - *Yes*: the number of animated elements is excessive
Question #7 (Adequate Image Resolution): do the images have an appropriate size and resolution to be visualized without pixelation?	- *No*: more than 50% of the images are too small to be correctly visualized and/or they are pixelated. There exists at least one image in the website homepage without an appropriate resolution and/or size. - *Sometimes*: between 50% and 90% of the images have an adequate resolution and can be correctly visualized. Besides, the inappropriate images are not in the website homepage - *Yes*: more than 90% of the images have an adequate resolution and can be correctly visualized

Fig. 12. Heuristic-related questions concerning Images (Design Category).

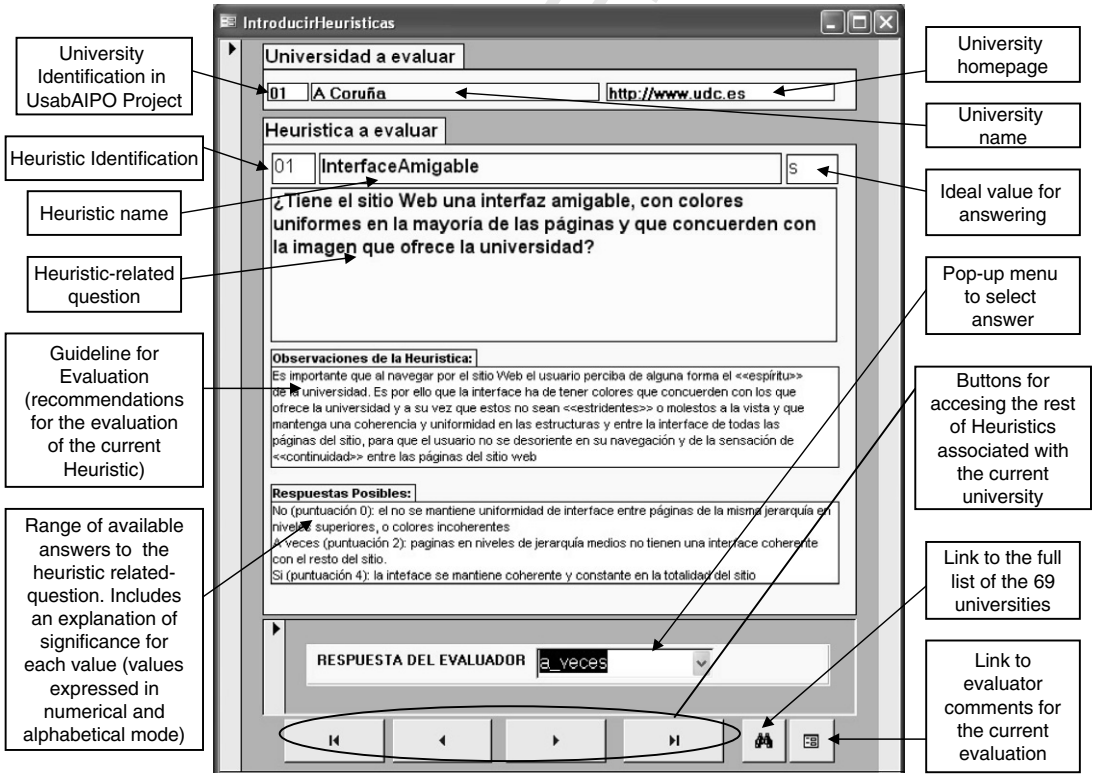

Fig. 13. Interface of the *UsabAIPO Heuristic Management System*. Form for introducing results from usability evaluation.

GetRules (Usability_Universities) where
Consequent has {(CleanInterface=no) or
(CleanInterface=sometimes)}
and support $>$ 0.6 and confidence $>$ 0.7]}

Besides, using the attributes present in the the fifteen association rules in L_1 as targets, the evaluation team was able to post more specific queries to obtain decision trees which helped to predict the behavior of the context of use under

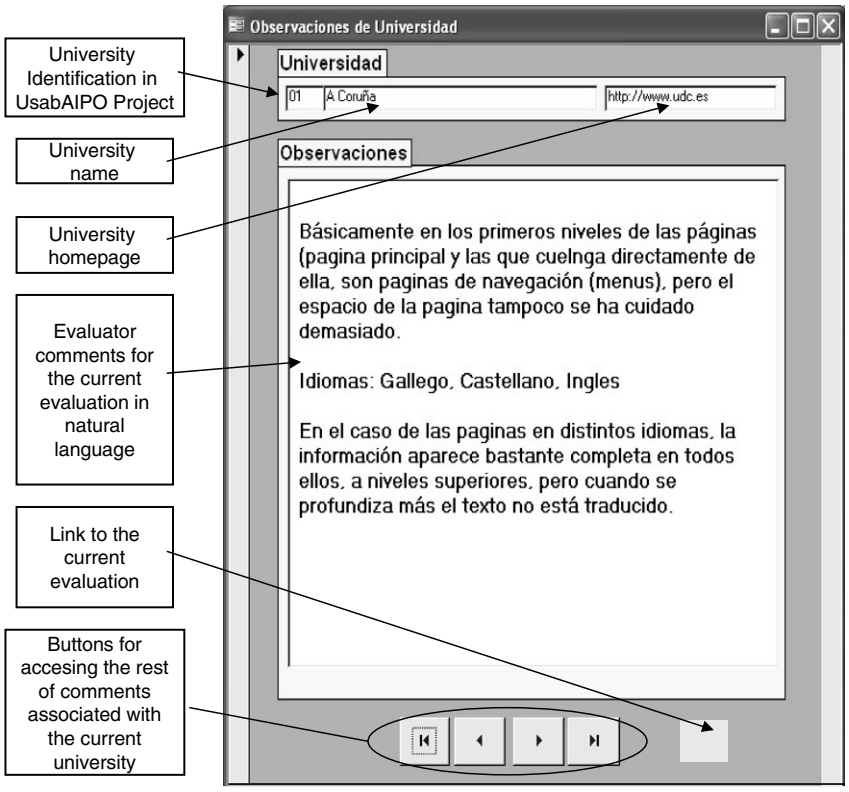

Fig. 14. Interface of the *UsabAIPO Heuristic Management System*. Form for adding comments to the usability evaluation.

Heuristic Related Question Id	Associated Attribute Name
#1	FriendlyInterface
#2	CleanInterface
#3	TextDesign
#4	LiquidDesign
#5	LabeledImages
#6	AnimatedElements
#7	ImageResolution

Fig. 15. Correspondence between heuristic-related questions concerning the Design category of UsabAIPO Project and attributes in *UsabAIPO Heuristic Management System*.

evaluation. In the case of this experimentation, 2/3 of the available data in *Usability_Universities* were randomly chosen by WEKA to be included in the training set, and the rest of the data was used as the test set. The threshold value for accepting decision trees was set in 70% of correctly classified instances. As a result, different decision trees were obtained, as the one shown in Fig. 18. This tree is the result of calculating the query shown below with the target `FriendlyInterface` (attribute #1 in the Design Category of the *UsP*):

```
java weka.classifiers.trees.ID3 -t usability_universities.arff
-c 1 -d r.txt
```

Fig. 18 shows the text-based representation provided by WEKA of the resulting decision tree for the target attribute `FriendlyInterface`. Every line of text shows the value assignment for an inner node (attributes in the DB) with the format "*Attribute_Name=value*", and every "|" corresponds to a new level in the tree. Those lines including *": class_value"* indicate that the target attribute has been assigned a particular class value. Note that every branch in the tree can be read as an "if-then" rule (e.g. the branch corresponding to the first three lines of the text in Fig. 18 stands for the rule "if (`LabeledImages=yes`) and (`ImageResolution=yes`) and (`LiquidDesign=no`) then (`FriendlyInterface=no`)").

Note that the decision tree in Fig. 18 showed that, according to the information stored in the spreadsheet *Usability_Universities*, the values stored in the attributes `LabeledImages` and `ImageResolution` (attributes

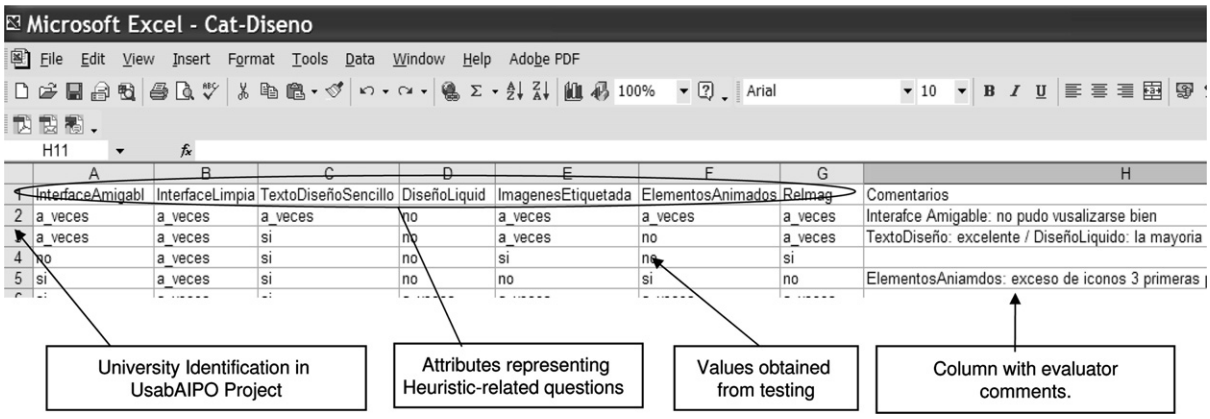

Fig. 16. Partial view of the sheet *Usability_Universities* corresponding to the Design Category.

```
1. CleanInterface=sometimes  TextDesign=yes  AnimatedElements=no 8
    ==> FriendlyInterface=yes 8 conf:(1)
2. CleanInterface=sometimes TextDesign=yes LabeledImages=yes 8
    ==> LiquidDesign=no 8 conf:(1)
3. FriendlyInterface=sometimes AnimatedElements=no ImageResolution=sometimes 8
    ==> LiquidDesign=no 8 conf:(1)
4. LiquidDesign=middle-level AnimatedElement=no ImageResolution=yes 7
    ==> CleanInterface=sometimes 7 conf:(1)
5. CleanInterface=yes AnimatedElements=no ImageResolution=sometimes7
    ==> LiquidDesign=no 7 conf:(1)
6. FriendlyInterface=no LiquidDesign=no ImageResolution=yes 7
    ==> CleanInterface=sometimes  7 conf:(1)
7. FriendlyInterface=no TextDesign=yes AnimatedElements=sometimes 7
    ==> CleanInterface=sometimes 7 conf:(1)
8. TextDesign=yes LiquidDesign=no AnimatedElements=sometimes 13
    ==> CleanInterface=sometimes 12 conf:(0.92)
9. FriendlyInterface=sometimes CleanInterface=sometimes AnimatedElements=sometimes 12
    ==> LiquidDesign=no 11 conf:(0.92)
10. FriendlyInterface=sometimes LiquidDesign=middle-level AnimatedElement=somewtimes 12
    ==> CleanInterface=sometimes 11 conf:(0.92)
11. AnimatedElements=no ImageResolution=yes 12 ==>
    TextDesign=yes 11 conf:(0.92)
12. TextDesign=yes LabeledImages=yes 12
    ==> FriendlyInterface=sometimes LiquidDesign=no 11 conf:(0.92)
13. FriendlyInterface=sometimes CleanInterface=yes LiquidDesign=no AnimatedElements=no 11
     ==>  FriendlyInterface=sometimes TextDesign=yes 10 conf:(0.91)
14. CleanInterface=yes LiquidDesign=middle-level AnimatedElements=no 9
    ==>  FriendlyInterface=sometimes 8 conf: (0.91)
15. ImageResolution=sometimes FriendlyInterface=sometimes 11
    ==> CleanInterface=no 10 conf:(0.90)
```

Fig. 17. Ranked list L_1 related with the Design Category of the UsabAIPO Project (visualization provided by the WEKA platform).

#5 and #7 concerning the Design category of the *UsP* Project) were crucial to predict the value of the attribute `FriendlyInterface` (as they are close to the root of the decision tree in Fig. 18). On the contrary, the value stored in the `TextDesign` attribute (attribute #3 concerning the Design category) was not extremely relevant when predicting the value for `FriendlyInterface`. We want to remark that the information stored in this decision tree was previously unknown for the evaluation team, as it was not intuitive enough to be discovered by means of statistics or simply by observation. Consequently, for the evaluation team this tree represents new *knowledge* about usability problems concerning the context of use under study.

At this stage the evaluation team was able to analyze the results obtained using the WEKA visualization module. On the one hand, ranked lists of association rules were visualized as shown in Fig. 17. On the other hand, decision trees were examined (as the one depicted in Fig. 18). Additionally, the evaluation team generated different Cartesian charts using the WEKA visualization facilities. Everyone of such *XY*-charts represents the frequency of different possible values associated with an attribute A_Z with respect to the values of two other attributes A_X and A_Y in *Usability_Universities*.

```
LabeledImages = yes
| ImageResolution = yes
| | LiquidDesign = no: no
| | LiquidDesign = middle-level: no
| | LiquidDesign = high-level: sometimes
| | LiquidDesign = yes: sometimes
| ImageResolution = no
| | LiquidDesign = no: sometimes
| | LiquidDesign = middle-level: no
| | LiquidDesign = high-level: sometimes
| | LiquidDesign = yes: null
| ImageResolution = sometimes: sometimes
LabeledImages = no
| ImageResolution = yes
| | CleanInterface = yes
| | | AnimatedElements = yes
| | | | TextDesign = yes: yes
| | | | TextDesign = no: sometimes
| | | | TextDesign = sometimes: null
| | | AnimatedElements = no: no
| | | AnimatedElements = sometimes
| | | | TextDesign = yes:no
| | | | TextDesign =no: yes
| | | | TextDesign = sometimes: null
| | CleanInterface= no: null
| | CleanInterface= sometimes: sometimes
| ImageResolution = no
| | TextDesign = yes: yes
| | TextDesign = no: sometimes
| | TextDesign = sometimes: null
| ImageResolution = sometimes: sometimes
| | LiquidDesign = no: yes
| | LiquidDesign = middle-level: null
| | LiquidDesign = high-level: null
| | LiquidDesign = yes
| | | AnimatedElements = yes: yes
| | | AnimatedElements = no: sometimes
| | | AnimatedElements = sometimes: yes
LabeledImages = sometimes
| ImageResolution = yes
| | LiquidDesign = no
| | | AnimatedElements = yes: sometimes
| | | AnimatedElements = no: null
| | | AnimatedElements = sometimes: yes
| | LiquidDesign = middle-level: sometimes
| | LiquidDesign = high-level: sometimes
| | LiquidDesign = yes: sometimes
| ImageResolution = no
| | CleanInterface = no: null
| | CleanInterface = yes: sometimes
| | CleanInterface = sometimes
| | | LiquidDesign = no: no
| | | LiquidDesign = middle-level: null
| | | LiquidDesign = high-level: null
| | | LiquidDesign = yes: sometimes
| ImageResolution = sometimes
| | TextDesign = yes: sometimes
| | TextDesign =no
| | | AnimatedElements = yes: sometimes
| | | AnimatedElements = no: null
| | | AnimatedElements = sometimes: yes
| | | TextDesign = sometimes: sometimes
```

Fig. 18. Decision tree #4 (target attribute *FriendlyInterface* with possible values {*yes, no, sometimes, null*}) corresponding to the Navigation Category within the UsabAIPO Project (visualization provided by the WEKA platform).

Values on the axes X and Y correspond to the possible values of A_X and A_Y. A number of marks or symbols is use to denote different possible values of the attribute A_Z. The size of every symbol used for representing values of A_Z is related to its frequency in *Usability_Universities* (the bigger the size is, the more frequent that value occurs in *Usability_Universities*). For example, Fig. 19 shows the relations among the attributes `CleanInterface`, `TextDesign` and `FriendlyInterface` present in the ranked list L_1.

To give a formal account of the usability problem patterns that were obtained, the evaluation team rephrased them using a format similar to the one shown in Fig. 4 (see two examples related to the Design Category in Fig. 20). A report describing the results was written [28]. As explained before, sometimes similar association rules were grouped and jointly rephrased. The scope (global or local) and severity degree of each usability problem pattern were discarded as relevant features, as within this experimentation they can be extrapolated from the values stored in the pattern attributes. The scope was defined by which data records were included in the database used as an input for the datamining process. The severity was determined by the *KDD*-metrics used for computing the patterns. Frequency was also disregarded in the case of decision trees, as it is not significant in a predictive tree-like chart. In the case of association rules, the frequency of every pattern was calculated on the basis of a weighted percentage which represents the number of rules supporting the pattern adjusted in function of the support and confidence of the rule. The explanation was replaced by a list of alphanumeric characters identifying association rules, ranked lists and decision trees supporting the usability problem pattern. Also the corresponding comment of the evaluation team (automatically stored by *UHMS* in the last column of *Usability_Universities*) was included.

6. Related work

To the best of our knowledge, there are no other approaches to extend the traditional *QUT* process by

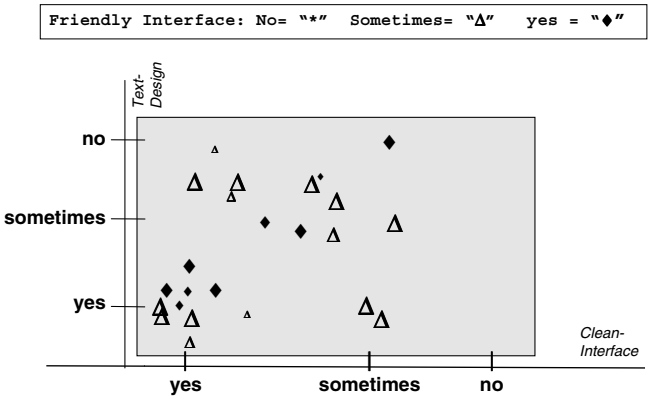

Fig. 19. Relation between three attributes present in the association rule #1 of the ranked list shown in Fig. 17 (visualization provided by the WEKA platform).

Usability Problem Pattern #1	The absence of a liquid design is related to non clean interfaces
Frequency (1 to 5)	4
Justification	- association rules #2,#5,#6,#8,#9,#10,and #14 in ranked list #1
Evaluation team Comment	The usability problem pattern involves most pages of problematic websites
Recommendations	• Recommend a liquid design to ensure a correct usage of space • The excess of images, animations and visual noise becomes critic if the web site does not includes a liquid design

Usability Problem Pattern #10	The lack of an adequate corporative image (friendly interface) and the use of non-labeled images result in an imbalanced use of the screen space
Frequency (1 to 5)	3
Justification	- association rule #15 in ranked list #1 - attributes close to the root in decision tree #4 (computed to predict "friendly interface" as a target attribute)
Evaluation team Comment	The usability problem pattern can be better observed in the web homepages
Recommendations	• Include a study of the corporative image of the university within the Requirement Analysis and use results as guidelines for the webpage graphic design. • Labeling images and image resolution are crucial to enhance corporative image

Fig. 20. A sample table reporting an usability problem pattern [19].

incorporating *KDD* techniques as presented in this paper. Typical models for usability evaluation under Usability Engineering [6,14,25,44,61,66,67] are defined for the evaluation of particular interactive systems rather than for the usability evaluation of a context of use as a whole. Several recent articles discuss the state of the art of the existing methodologies and methods for usability testing, and in particular for qualitative usability evaluation [4,8,10,21,23,34,37,43,54,61,62]. None of these survey papers discusses the relationship between the usability evaluation process and the use of *KDD* techniques as a tool for finding (possibly unknown) relevant patterns related to usability problems.

There exist some approaches which include association rule mining for assessing usability, but always oriented towards the evaluation of a single interactive system. For example, the AWUSA framework [68] presents an automatic tool for evaluating the usability of a web site by combining logging techniques and datamining, along with the static structure of the web site. Association rules allow to detect problematic patterns in the attributes of the resources requested by users. Another approach where association rules are used to test the usability of a single system is described in [2]. In this case, logging techniques are based on browsing activities performed by users. Recommendation models are defined as sets of association rules which allow to improve the usability of the system.

The potential application of decision trees as a tool for improving usability evaluation has been analyzed in [22]. In this case, decision trees are used to model user profiles using as an input the user interaction with the system, predicting interface features on the basis of such interaction. However, such approach is not intended to test the usability of a context of use as a whole. In [65] a similar approach is presented. In this case decision trees are used as a methodology to evaluate the interactive quality of a system beyond the usability testing. The approach is based on the notion of "user satisfaction", which is determined by computing a decision tree representing a hierarchy of the user's expected benefits when using the system. Techniques for statistical analysis (such as correlation) can also be applied to find relationships in usability testing on the basis of the available data. However, this kind of techniques are more adequate for performing quantitative analysis than

for being used in proposals like the QUT_C process presented in this paper, which focuses on a more qualitative perspective. Besides, approaches to computing correlation between two random variables are usually based on numeric data rather than on purely categorical data (such as those attributes involved in qualitative descriptions). In addition, note that in our proposal the variables to be analyzed do not need to be identified explicitly, as they are found automatically by the association rule mining algorithm.

With respect to the development of the specialized software tool *UHMS* (see screenshots in Fig. 13), it must be noted that a similar tool is available in the open source *uzReview* sidebar of the Mozilla browser. This sidebar, which was designed to facilitate *HE* by the logging of heuristics used against an URL or a keyword (this one allowing the review of processes or workflows), incorporates a rich text editor to support drag and drop of content from the browsed pages and the edition of user comments. However, in available versions of the *uzReview* sidebar it is not possible to make the user vision of the interface that is being currently evaluated independent from the *uzReview* screenshots, making difficult the perception of this interface as a whole. Indeed, the *uzReview* screenshots needed to be displayed over the interface under evaluation in order to capture its current usage. Even when this characteristic of the *uzReview* sidebar can be considered as an advantage, in the case of inspection methods like *HE* the closer to the real user perception the usability evaluation is performed, the more relevant will be the obtained results. It must be remarked that other platforms similar to the *UHMS* are intended to test only one interactive system or are defined for specific contexts of use (different from the one considered in the experimentation included in this paper). For example, a preliminary study to develop a *HE* tool for assessing the usability of Healthcare Information Systems is discussed in [45]. In this case, the aim is to work with clinical systems, without establishing clearly if the final product will be able to perform the usability evaluation of more than one system simultaneously or not.

7. Conclusions and future work

Identifying usability problem patterns in a given context of use is a challenging problem nowadays, and providing an appropriate solution for such a problem can be extremely helpful for software development and Usability Engineering (*UE*) in several respects. On the one hand, usability experts can rely on those usability problem patterns detected for a given context *C* in order to evaluate new interfaces of software systems from the same context *C*. On the other hand, the contextual knowledge about usability problems can also help software developers to prevent possible usability problems when novel interactive systems are under development in that context *C*.

QUT methods provide a powerful tool for assessing the usability of interactive software systems. However, as we have discussed in this paper, these methods are rather limited when analyzing a context of use *C* as a whole on the basis of the joint analysis of a large number of interactive systems belonging to it. To cope with this problem we have proposed the QUT_C approach, a novel characterization of the *QUT* process based on the integration of the traditional *UE* methodologies and *KDD* techniques. Although the QUT_C process involves a more complex procedure, according to our experiments the associated costs did not increase significatively. In that respect, it must be noted that during the traditional Evaluation Stage under *UE* different data generated during the evaluation processes are usually compiled into databases (as such databases are required for the statistical analysis performed during the quantitative usability evaluation). The re-utilization of these databases minimizes the cost associated with the generation of the documentation needed to carry out the proposed approach.

An additional advantage of the QUT_C approach is that the computation of *KDD* algorithms should be performed only one time to achieve satisfactory results. This computation should be performed either on the basis of the previously decided parameters (values for support, confidence or other metrics in the case of the association rules; particular "target attributes" in the case of decision trees) or on the basis of the queries posted by the evaluation team. The high level of automatization in the *KDD* stage within the QUT_C process is also a highly desirable feature [37]. Such automatization enhances the systematization and the predictability in the findings of this usability evaluation, minimizing the bias of the evaluation team when considering a large amount of qualitative data. However, it must be noted that the evaluation team is always in charge of controlling the QUT_C process as a whole. Even when the QUT_C methodology improves the decision making capabilities of the evaluation team, it does not replace the final discussion within the team about the results obtained after the usability testing of a context of use.

From our experiments we can conclude that the integration of a *KDD*-based methodology for assessing qualitative usability of a context of use *C* can be performed successfully as part of a real-world case (the assessment of usability problem patterns in the *Universia* portal). One important advantage of our proposal is that many intuitions that were informally stated by the evaluation team during the *QUT* process can now be appropriately analyzed through *KDD*-based techniques. On the other hand, the Evaluation Stage under *UE* can be enriched with the detection of hidden relationships among qualitative data that are detected and documented with a formal basis (e.g. when there is a particular association rule supporting a given relationship). Another interesting application of our proposal is to detect the qualitative usability problems of a prototype belonging to a particular context of use *C* by focusing its usability evaluation on the critical patterns obtained through the application of the QUT_C process over

C. Detecting such situations at early stages can reduce considerably software development efforts in time and money. In summary, the integration of *KDD*-based techniques into the traditional *QUT* process allows that *QUT* capabilities go further than the testing of one individual system, making possible the qualitative usability estimation of a context of use as a whole.

Part of our future work will be focused on testing different ranking functions for association rules, evaluating their applicability in the QUT_C process. In particular, we are interested in modeling rule prioritization by taking into account the cost associated with software development, as suggested in [13]. To achieve this, more powerful *KDD* software platforms will be required. In that respect, recent research has been oriented towards the integration of *KDD* and query languages for developing higher level systems, as the one proposed in this paper. In this context, the KDDML platform [60] integrates a mixture of data access, data preprocessing, datamining models extraction and deployment, providing a powerful middleware mark up language for *KDD*. The language of KDDML is XML-based, both for query syntax and data/model representation. As stressed in [60], this kind of platform can be seen as an evolution of *KDD* engines like WEKA, on top of which higher abstraction levels or final applications can be built. We think that through such extended *KDD* platforms the modeling of alternative ranking functions can be better analyzed and constrasted.

Another research line we are currently exploring is the development of a usability evaluation module based in the QUT_C process capable to be integrated in a *Usability Evaluation Management System*. This management system should allow to carry out the usability evaluation of a context of use by providing different alternatives for choosing usability evaluation methods. In that respect, note that the database required in the QUT_C process to compute the *KDD*-based steps could be used to calculate metrics or other quantitative usability results. In particular, the integration of our proposal with SketchiXML [15] (a multiagent interactive application that enables designers and end users to sketch user interfaces with different levels of details and support for different contexts of use) is under consideration. In this setting, the QUT_C process could be included as an additional feature of SketchiXML, helping to identify relevant usability problem patterns for an interface under design. Research in this direction is currently being pursued.

8. Uncited references

Q3 [3,5,9,12,16,24,30,31,35,38–41,52,53,55–57,63,64,69,70,73].

References

[1] ISO 9241-11. Ergonomic requirements. Part 11: Guidance on usability, 1998.
[2] J. Alipio, J. Poças, P. Azevedo, Recommendation with association rules: a web mining application, Data Mining and Warehouses Conf. IS-2002 (2002).
[3] P. Alreck, R. Settle, The Survey Research Handbook, McGraw-Hill, 1994.
[4] R. Bailey, R. Molich, J. Dumas, J.M. Spool, Usability in Practice: Formative Usability Evaluations, ACM CHI2002 3 2002.
[5] P. Bauersfeld, Software by Design: Creating People Friendly Software, M&T Books, 1994.
[6] H. Beyer, K. Holtzblatt, Contextual Design. Defining Customer–Centered Systems, M. Kaufmann, 1998.
[7] R. Bias, Usability Inspection Methods, Chapter The Pluralistic Usability Walkthrough: Coordinated Empathies, John Wiley, 1994, pp. 63–76.
[8] T. Brink, D. Gergle, S. Wood, Design Web Sites that Work: Usability for the Web, Morgan-Kaufmann, 2002.
[9] J. Brooke, Usability Evaluation in Industry, CHAPTER=SUS: A Quick and Dirty Usability Scale, Taylor & Francis, 1996, pp. 189–194.
[10] M. Cecelia Buchanan, Polle T. Zellweger, Automatic temporal layout mechanisms revisited, ACM Trans. Multimedia Comput. Commun. Appl. 1 (1) (2005) 60–88.
[11] J. Carroll, Making Use:Scenario-Based Design of Human-Computer Interactions, MIT Press, 2000.
[12] J.P. Chin, V.A. Diehl, K.L. Norman, Development of an instrument measuring user satisfaction of the human-computer interface, in: CHI '88: Proc. SIGCHI Conf. Human Factors Computing Systems, ACM Press, 1988, pp. 213–218.
[13] D. Choir, B. Ahn, S. Kim, Priorization of association rules in DM: Multiple criteria decission approach, Expert System with Applications 29 (2005) 867–878.
[14] L. Constantine, L. Lockwood, Software for Use. A Practical Guide to the Models and Methods of Usage-Centered Design, Addison-Wesley, 1999.
[15] A. Coyette, J. Vanderdonckt. A Sketching tool for designing anyuser, anyplatform, anywhere user interfaces, in: LNCS. Proc Human-Computer Interaction –INTERACT '05 Int. Conf.
[16] F.D. Davis, Perceived usefulness, perceived ease of use, and user acceptance of information technology, vol. 13, MIS Quarterly, 1989.
[17] J. Demsar, B. Zupan, G. Leban, Orange: From Experimental Machine Learning to Interactive Data Mining White Paper, Faculty of Computer and Information Science, University of Ljubljana, 2004.
[18] R. Stanley Dicks, Mis-usability: on the uses and misuses of usability testing, in: SIGDOC '02: Proc. 20th Ann. Intl. Conf. Computer Documentation, New York, NY, USA, 2002, ACM Press, pp. 26–30.
[19] J.S. Dumas, J.C. Redish, A Practical Guide to Usability Testing, Intl. Specialized Book Service Inc., 2000.
[20] J.S. Dumas, R. Molich, R. Jeffries, Describing usability problems. Are we sending the right message? Interactions (2004) 24–29.
[21] X. Ferré, N. Juristo, A.M. Moreno, Framework for integrating usability practices into the software process, Lecture Notes in Computer Science, 6th Int. Conf., PROFES 2005, 3547, (2005) 202–215
[22] J. Finlay, Machine learning: A tool to support improved usability, Applied Artificial Intelligence 11 (1997) 633–665.
[23] A. Floria Cortes, Recopilation of usability methodologies (available in Spanish). Technical report, Desing Engineering and Production Department. Centro Politécnico Superior, Zaragoza University (Spain), 2000.
[24] E. Frøkjær, K. Hornbæk, Cooperative usability testing: complementing usability tests with user-supported interpretation sessions, in: CHI'05 Ext. Ab. on H. Factors in Computing Systems, ACM Press, 2005, pp. 1383–1386.
[25] G. Gaffney, Usability techniques series: Participatory design, Information and Design (1999).
[26] M.P. González, T. Granollers, J. Lorés. Métricas predictivas de la usabilidad: un nuevo enfoque para su ponderación cualitativa

(available in Spanish), in: Proc. VI Spanish Conf. Human Computer Interact. (INTERACCION' 2005), 2005, pp. 233–241.
[27] M.P. González, T. Granollers, J. Lorés, A hybrid approach for modelling early prototype evaluation under user-centred design through association rules, in: Proc. XIII Intl. Workshop Design, Specification and Verification of Interactive System DSV-IS '06. Sponsored by ACM SIGCHI, Eurographics, and IFIP WG 13.5. Dublin, Ireland (to appear in Lecture Notes in Computer Science Springer Series), 2006.
[28] M.P. González, J. Lorés, A. Pascual, T. Granollers, UsabAIPO Project. Evaluation of the context of use as a whole by means of Heuristic Evaluation and Knoweldge Discovering in Databases Techniques. Technical report, AIPO Association, Spain (available in Spanish), 2006.
[29] T. Granollers, J. Lorés, HCI Related Papers of Interaccion 2004, chapter Incorporation of Users in the Evaluation of Usability by Cognitive Walkthrough, Springer-Verlag, 2005, pp. 103–119.
[30] T. Greenbaum, The Handbook for Focus Group Research, Sage Pubns, 1997.
[31] Z. Guan, Sh. Lee, E. Cuddihy, J. Ramey, The validity of the stimulated retrospective think-aloud method as measured by eye tracking, in: CHI '06: Proc. SIGCHI Conf. Human Factors Computing Systems, ACM Press, 2006, pp. 1253–1262.
[32] J. Han, M. Kamber, Data Mining: Concepts and Techniques, M. Kaufmann, 2000.
[33] K. Holtzblatt, H. Beyer. Getting started with contextual techniques, in: Proc. of CHI Int. Conf., 1996.
[34] K. Hornbæk, Current practice in measuring usability: challenges to usability studies and research, Int. J. Human Computer Studies 64 (2) (2006) 79–102.
[35] K. Hornbæk, E. Frøkjær, Evaluating user interfaces with metaphors of human thinkingLNCS Proc. User Interfaces for All, 2615, Springer Verlag, 2002, pp. 486–507.
[36] T. Imielinski, A. Virmani, MSQL: A query language for database mining, Data Min. Knowl. Discov. 3 (4) (1999) 373–408.
[37] M.Y. Ivory, M.A. Hearst, The state of the art in automating usability evaluation of user interfaces, ACM Comput. Surv. 33 (2001) 470–516.
[38] J.R. Lewis, Psychometric evaluation of an after-scenario questionnaire for computer usability studies: The ASQSIGCHI Bull., 23, ACM Press, 1991, pp. 78–81.
[39] J.R. Lewis, Psichometric evaluation of the post-study system usability questionnaire: The PSSUQ, in: Proc. of 36th Meeting Human Factors Soc. 1992, pp.1259–1263.
[40] J.R. Lewis, Ibm computer usability satisfaction questionnaires: Psychometric evaluation and instructions for use, Int. J. Human-Computer Interact. 7 (1) (1995) 57–78.
[41] H.X. Lin, Yee-Yin Choong, G. Salvendy, Usability evaluation methods, volume 16, chapter A Proposed Index of Usability: A Method for Comparing the Relative Usability of Different Software Systems, pp. 267–278. Behaviour and Information Technology Find, 1997.
[42] J. Lorés, M.P. González, A. Pascual, Primera fase de análisis del Proyecto UsabAIPO (available in Spanish), in: Proc. VI Cong. de Interacción Persona Ordenador (INTERACCION 2005), 2005, pp. 217–221.
[43] J. Mao, K. Vredenburg, P. Smith, T. Carey, The state of user-centered design practice, Commun. ACM, pp. 105–109.
[44] D.J. Mayhew, The Usability Engineering Lifecycle. A practioner's handbook for user interface design, M. Kaufmann, 1999.
[45] K. McGrow, A. Horsman Brennan, J. Preece, Development of a tool for heuristic evaluation of healthcare information systems, Computers, Informatics, Nursing, J. Hospice & Palliative Nursing (2004).
[46] T. Mitchell, Machine Learning, McGraw Hill, 1997.
[47] M.J. Muller, L. Matheson, C. Page, R. Gallup, Methods & tools: participatory heuristic evaluation, Interactions 5 (5) (1998) 13–18.
[48] A. Netz, J. Bernhardt, S. Chaudhuri, U. Fayyad, Integrating data mining with sql databases: OLE DB for data mining. in: Proc. 17th Int. Conf. Data Engineering, 2001, pp. 379–387.
[49] J. Nielsen, Usability Engineering, M. Kaufman, 1993.
[50] J. Nielsen, Enhancing the explanatory power of usability heuristics, in: CHI'94: Conf. Companion Human Factors Computing Systems, 1994.
[51] J. Nielsen, Scenario-based Design: Envisioning Work and Technology in System Development, chapter Scenarios in Discount Usability Engineering, Jon Wiley & Sons, 1995, pp. 59–83.
[52] J. Nielsen, T. Clemmensen, C. Yssing, Getting access to what goes on in people's heads?: reflections on the think-aloud technique, in: NordiCHI'02: Proc. Second Nordic Conference Human-computer Interact., ACM Press, 2002, pp. 101–110.
[53] J. Nielsen, R. Molich, Heuristic evaluation of user interfaces, in: CHI'90: Proc. SIGCHI Conf. Human Factors Computing Systems, ACM Press, New York, NY, USA, 1990, pp. 249–256.
[54] F. Paternó, Model-based Design and Evaluation of Interactive Application, Springer-Verlag, 2000.
[55] G. Perlman, Practical usability evaluation, CHI Extended Abstracts (1997) 168–169.
[56] D. Pinelle, C. Gutwin, Groupware walkthrough: adding context to groupware usability evaluation, in: CHI '02: Proc. SIGCHI Conf. Human Factors Computing Systems, ACM Press, New York, NY, USA, 2002, pp. 455–462.
[57] J. Preece, Y. Rogers, H. Sharp, D. Beyond, S. Holland, T. Carey, Human-Computer Interaction, Addison-Wesley, 1994.
[58] D. Pyle, Data Preparation for Data Mining, M. Kaufmann, 1999.
[59] ANSI Test Reports. ANSI NCITS 354-2001. Common Industry Format for Usability test report, 2001.
[60] A. Romei, S. Ruggieri, F. Turini, KDDML: a middleware language and system for knowledge discovery in databases, Data and Knowledge Engineering 57 (2) (2006) 179–220.
[61] M. Rosson, J. Carroll, Usability Engineering: Scenario-based Develoepement of HCI, M. Kaufmann, 2002.
[62] J. Sauro, E. Kindlund, A method to standardize usability metrics into a single score, ACM HCI05, 2005.
[63] J. Scholtz. Adaptation of traditional usability testing methods for remote testing. in: HICSS '01: Proc. 34th Annual Int. Conf. System Sciences, vol.5, p. 5030. IEEE Computer Society, 2001.
[64] V.S. Segawa, V.M. Sugimura, V.K. Ishigaki, New web-usability evaluation method: Scenario-based walkthrough, Fujitsu Scientific & Technical J. Special Issue in Universal Design 41 (1) (2005).
[65] M. Sikorski, Beyond product usability: user satisfaction and quality management, in: CHI '00: CHI '00 Extended Abstracts Human Factors Computing Systems, ACM Press, New York, NY, USA, 2000, pp. 61–62.
[66] C. Snyder, Paper Prototyping. The fast and easy way to design and refine user interfaces, Morgan-Kaufmann, 2003.
[67] A. Sutcliffe, User-Centred Requirements Engineering. Theory and Practice, Springer-Verlag, 2002.
[68] T. Tiedtke, C. Märtin, N. Gerth. Awusa, A tool for automated website usability analysis. 9th Int. Workshop DSVIS, 2002.
[69] E. van Veenendaal, Low-cost usability testing, Software Quality and Software Testing in Internet Times (2002) 153–164.
[70] P. Vora and M. Helander. Symbiosis of Human and Artifact, in: Proc. Sixth Int. Conf. Human-Computer Interact., chapter A Teaching method as an alternative to the concurrent think-aloud method for usablity testing, Elsevier Science Publishing Company, pp. 375–380, 1995.
[71] C. Wharton, J. Rieman, C. Lewis, P. Polson. Usability Inspection Methods, chapter The Cognitive Walkthrough Method: A practitioners guide, p. 1994. John Wiley & Sons Inc, 1994.
[72] Ian H. Witten, Eibe Frank, Data Mining: Practical machine learning tools and techniques, M. Kaufmann, 2005.
[73] Z. Zhang, V. Basili, B. Shneiderman, Perspective-based usability inspection: An empirical validation of efficacy, Empirical Softw. Eng. 4 (1) (1999) 43–69.

Please cite this article in press as: M.P. González et al., Enhancing usability testing through datamining techniques: ..., Inform. Softw. Technol. (2007), doi:10.1016/j.infsof.2007.06.001