DEPARTMENT OF COMPUTER SCIENCE
SCHOOL OF SCIENCES
UNIVERSITY OF CRETE

Design, Development and Evaluation of a Web-based Inspection Tool for the Assessment of the User-Experience of Online Services

Pantelis Mandilaras

Master of Science

Heraklion, November 2007

DEPARTMENT OF COMPUTER SCIENCE
SCHOOL OF SCIENCES
UNIVERSITY OF CRETE

Design, Development and Evaluation of a Web-based Inspection Tool for the Assessment of the User-Experience of Online Services

Submitted to the
Department of Computer Science
in partial fulfillment of the requirements for the degree of
Master of Science

November 23, 2007

Author:

Pantelis Mandilaras
Department of Computer Science

Board
of enquiry:
Supervisor

Constantine Stephanidis
Professor

Member

Dimitris Plexousakis
Professor

Member

Panagiotis Trahanias
Professor

Accepted by:
Chairman of the
Graduate Studies Committee

Panagiotis Trahanias
Professor
Heraklion, November 2007

Acknowledgements

I am grateful to my supervisor Professor Constantine Stephanidis for his assistance, encouragement and support.

I would also like to thank the Department of Computer Science of the University of Crete and the Institute of Computer Science (Human Computer Interaction Laboratory) of the Foundation for Research and Technology – Hellas (FORTH) for providing me with the means for setting and achieving my research goals.

For their support and guidance, I would like to thank many friends and colleagues, and in particular, Dr. Margherita Antona, Mr. Alexandros Mourouzis, Mr. Andreas Dimakis, Mr. Giannis Metaxas and Mr. Panagiotis Papadakos.

Finally, I would like to thank close friends and my family who stood by me the past few years and reminded me that there's light at the end of every tunnel, when I needed it the most.

Dedicated to my mother,
Vasiliki,
for all that she's done
and still does.

Design, Development and Evaluation of a Web-based Inspection Tool for the Assessment of the User-Experience of Online Services

Pantelis Mandilaras

Master of Science

Department of Computer Science
School of Sciences
University of Crete

Abstract

Due to the rapid evolution of information technology, evaluation has become a critical factor for the success of the development process. Therefore, evaluation tools acquire increased importance. However, currently available evaluation tools are limited, and usually focus on single aspects of a user interface, such as usability, availability, accessibility, etc. This thesis presents the design, implementation and evaluation of a web-based evaluation tool, based on a paper-based inspection tool for evaluating the user experience, aimed at providing a structured way for assessing in combination (i.e., holistically) various aspects of user interfaces. The design and development of the tool are described and discussed. The tool has been evaluated through both an expert-based and a user-based experiment, and the results are discussed.

The web-based inspection tool, named ORIENT, is easy to learn and use and easily available over the net to the general public. The current implemented version will be made available online in order to test the inspection tool in real conditions of use and findings from studies carried out with it may be used in the creation of design guidelines for usable and accessible systems.

Σχεδιασμός, Υλοποίηση και Αξιολόγηση ενός Δικτυακού Εργαλείου Αξιολόγησης για την Επισκόπηση Συμμόρφωσης Δικτυακών Υπηρεσιών με τις Απαιτήσεις των Χρηστών

Παντελής Μανδηλαράς

Μεταπτυχιακό Δίπλωμα Ειδίκευσης

Τμήμα Επιστήμης Υπολογιστών
Σχολή Θετικών Επιστημών
Πανεπιστήμιο Κρήτης

Περίληψη

Δεδομένης της ραγδαίας εξέλιξης των πληροφοριακών τεχνολογιών, η αξιολόγηση έχει αναχθεί σε καθοριστικό παράγοντα για την επιτυχία της διαδικασίας ανάπτυξης. Επομένως, τα εργαλεία αξιολόγησης γίνονται ακόμα πιο σημαντικά. Ωστόσο, τα διαθέσιμα υπάρχοντα εργαλεία υστερούν σε ορισμένους τομείς και συνήθως επικεντρώνονται σε μεμονωμένα χαρακτηριστικά μιας διεπαφής, όπως η ευχρηστία, η διαθεσιμότητα, η προσβασιμότητα, κλπ. Αυτή η μεταπτυχιακή εργασία παρουσιάζει τον σχεδιασμό, την υλοποίηση και την αξιολόγηση ενός δικτυακού εργαλείου αξιολόγησης, το οποίο βασίζεται σε ένα paper-based εργαλείο αξιολόγησης για την αποτίμηση της συμμόρφωσης ως προς των αναγκών των χρηστών. Στόχος αυτού του εργαλείου είναι να παρέχει ένα δομημένο τρόπο αξιολόγησης διάφορων χαρακτηριστικών των διεπαφών σε συνδυασμό (δηλαδή, ολιστικά). Η σχεδίαση και υλοποίηση του εργαλείου περιγράφονται και αναλύονται. Η αποτίμηση της ευχρηστίας του εργαλείου έχει πιστοποιηθεί τόσο με αξιολόγηση από ειδικούς, όσο και με εμπειρική αξιολόγηση και τα αποτελέσματα αυτά περιγράφονται και αναλύονται.

Τόσο η μάθηση, όσο και η χρήση, του δικτυακού εργαλείου αξιολόγησης, το οποίο ονομάζεται ORIENT, χαρακτηρίζεται εύκολη, και το εργαλείο είναι ευκόλως προσβάσιμο από το ευρύ κοινό διαμέσου του Διαδικτύου. Η τρέχουσα υλοποιημένη έκδοση του εργαλείου θα γίνει διαθέσιμη στο ευρύ κοινό μέσω του Διαδικτύου με σκοπό να ελεγχθεί το εργαλείο αξιολόγησης κάτω από πραγματικές συνθήκες χρήσης και τα ευρήματα των μελετών που θα διεξαχθούν με το εργαλείο θα μπορούν να χρησιμοποιηθούν για την δημιουργία οδηγιών για την σχεδίαση εύχρηστων και προσβάσιμων συστημάτων.

Table of Contents

List of Figures

List of Tables

1 Introduction

Nowadays, due to the rapid evolution of information technology, evaluation has become a critical part of the development process. Several inspection methods and tools are available for assessing systems and their user interfaces.

User questionnaires are one of the most typical and consolidated tools to evaluate user interfaces [19]. A well-designed questionnaire can give good insight into the problems of the tested application, even if very detailed information is difficult to obtain. Questionnaires commonly used include CELLO [20], the Questionnaire for Interaction Satisfaction (QUIS) [21], the Software Usability Measurement Inventory (SUMI) [22] and the System Usability Scale (SUS) [23]. Semi-automatic inspection tools, such as a multidisciplinary tool at the evaluation & training portal of the Finnish Virtual University (FVU) [24], Systematic Usability Inspection Tool (SUIT) [25] and Usability Problem Inspector (UPI) [26], are partially automated and usually guide evaluators through the inspection process by a step-by-step procedure (typically documented in a theoretical framework). Automatic tools, a more recent approach, examine source code of web pages to derive adherence to universally accepted stylistic and objective guidelines. A sample of common automatic tools includes Web Static Analyzer Tool (WebSAT) [27], Bobby [28], Wave [29] and Cynthia Says [30].

Although most of these tools possess certain strengths, they usually perform assessments in an arbitrary way and often focus on single aspects of a UI, such as usability, availability, accessibility, etc. The User Experience Evaluation Framework [36] is aimed at providing a structured way for assessing in combination (i.e., holistically) various aspects including those of visibility, perceived usefulness and ease of use, accessibility, etc.

A prototype, paper-based inspection tool, called ORIENT, has been developed to facilitate experts in employing the framework in practice. However due to its current form, it is difficult and time consuming to use in practice. Under the light of the above, this work is aimed at developing a Web-based version of the tool, which will improve the ease of use of the inspection tool, as well as its visibility and availability. Additionally, the web-based version of ORIENT will include communication features supporting collaboration in the context of evaluation cases, and evaluation practice in general.

Furthermore, it is likely that its sustainability will be enhanced as well (since a web-based version of the tool will support reusability).

The current implemented version will be made available online in order to test the inspection tool in real conditions of use and findings from studies carried out with it may be user in the creation of design guidelines for usable and accessible systems.

1.1. Thesis Structure

This thesis is organized as follows: Chapter 2 introduces the two main criteria of almost every evaluation, usability and accessibility. After a brief definition of the two concepts and a presentation of the main methods for assessing them, the concept of inspection tool is introduced. Commonly used as well as research inspection tools are briefly presented, divided in three groups according to the degree of automation they support - paper- based, semi-automatic and fully automatic, with their strengths and weaknesses. This presentation of tools highlights desired characteristics that an inspection tool should possess.

Chapter 3 introduces the User Experience Evaluation Framework and its underlying theory. The paper-based prototype of the inspection tool is then presented and the step-by-step process of using it is explained. Finally, a reference to the pilot application of the inspection tool and method is made.

Chapter 4 addresses the design of the ORIENT inspection tool. The interviews of developers and users of the method and tool, which lead to an initial set of requirements, are reported. User groups of the tool are identified and documented, as well as the functions each of them is expected to carry out with the tool. This analysis enriches the aforementioned initial set of requirements. The database that supports the inspection tool is outlined by the presentation of its entity-relationship model. Afterwards, tasks are decomposed with hierarchical task analysis (HTA), and finally preliminary user interface mock-ups are created.

Chapter 5 documents the implementation of the tool, covering the creation of database tables, according to the database model, and interfaces, according to the mock-ups from the design phase, as well as the implementation of the underlying functionality that enables the communication of the data and the presentation level. Chapter 6 describes the process of evaluating the usability of the inspection tool itself. The first stage comprises of a heuristic evaluation to sort out the majority of usability issues and the second stage involves users actually interacting with the system according to a test scenario towards an end. Results of the evaluation process of ORIENT are both quantitative and qualitative.

Chapter 7 concludes this thesis highlighting the strong points of the inspection tool, as validated by its assessment, and briefly mentions future work and recommendations.

2 Background

2.1 Usability

Some may argue that coming up with a clear and concise definition of usability "can be aptly compared to attempts to nail a blob of Jell-O to the wall" [13]. However, most commonly used definitions are similar. For instance:

- "The extent to which a product can be used by specified users to achieve specified goals in a specified context of use with effectiveness, efficiency, and satisfaction" [16].
- "The measure of the quality of the user experience when interacting with something - whether a Web site, a traditional software application, or any other device the user can operate in some way or another" [26].
- "Usability means that the people who use the product can do so quickly and easily to accomplish their own tasks" [29].

Jakob Nielsen combined all theses ideas and defined usability by identifying its five components / attributes: learnability, efficiency, memorability, errors and satisfaction [26] (figure 2.1)**.** A usable system should be easy to learn so that the user can rapidly start doing some work (*learnability*), efficient to use, so that once it is learned, the user can achieve a high level of productivity (*efficiency*), easy to remember, so that the casual user is able to return to the system after a time and not have to learn it all over again (*memorability*), have a low error rate, so that users make few errors and can easily recover from them (*errors*) and be pleasant to use, so that users enjoy using it (*satisfaction*).

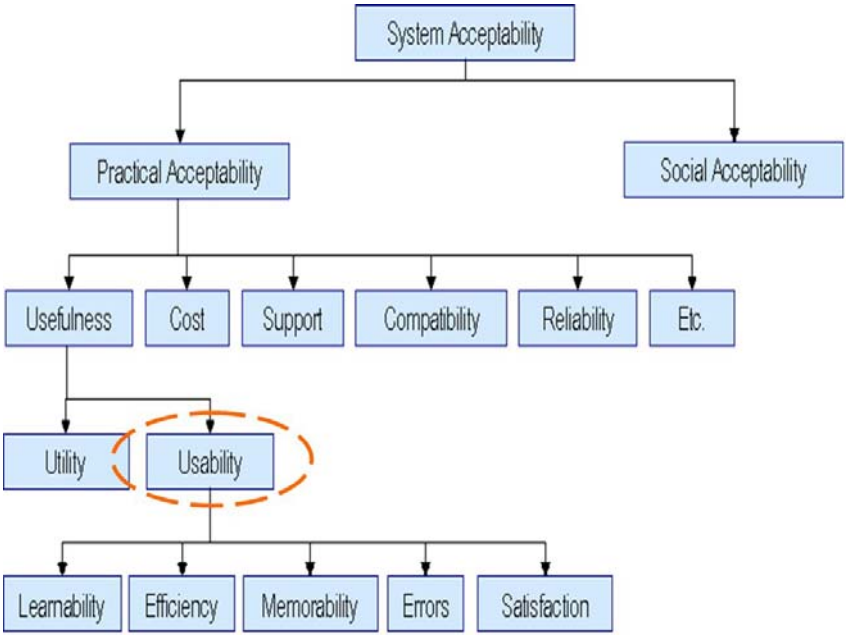

Figure 2.1: The five components of Usability, according to Jakob Nielsen.

In all these definitions of usability, the focus is on the user, not on the product. A product's usability is determined by the user's perception of the quality of the product, based on the user's ease of use, ease of learning and relearning, the product's intuitiveness for the user, and the user's appreciation of the usefulness of a product. In every case, usability must be understood as matching the needs of a particular user for a particular use. If the product doesn't add value to the way in which the user currently performs tasks, then the user will have no use for the product.

Furthermore, the lack of usability can cause problems which may range from simply annoying or frustrating the user to life-threatening situations.

Most, if not all, of us have come across products in our everyday lives that we can't use or which cause difficulties. Products are intended to facilitate people and make their lives more easy and pleasant. However, if they are difficult to use, they end up doing exactly the opposite, annoying and frustrating the very people they were intended to help.

Users may have compromised with lack of usability, in the past, as the price to pay for products with ample functionality. The tides are changing, however, and as public awareness of usability issues increases, usability is becoming a significant factor in purchase decisions. Good design practices, including usability, may be one of the ways for manufacturers to gain significant advantages over their competitors. Moreover, unusable products in the working environment waste time and money. The usability of products used in the workplace can affect the employees' level of job satisfaction within the organization, especially when the former spend a great deal of their time using a particular product.

In some cases, the usability of a product can affect the safety of those using the product, as well as the safety of others. For example, a GPS car navigation system is expected to be operated by the user whilst driving the vehicle. Thus, the user's hands and eyes are engaged in actually driving the vehicle and the consequences of distracting his/her attention from the driving task are potentially disastrous.

It is safe to assume that usability is very important, but good intentions are not always enough. Designers, developers, and product managers may come up with extravagant patterns and tricks to create what, in their view, will be the definition of a usable product. However, the notion of usability may differ a little (to a lot) between that of the former and that of the actual end users. This is where evaluation comes to fill the gap.

Usability engineering is the research and design process that ensures a product has good usability [39]. In the field of HCI, there are three basic evaluation methods: expert-based, model-based and user-based [8]. Expert-based approaches assess an interface for compliance with known design principles and guidelines. Model-based evaluation is the application of theoretical models to specific design questions. User-based approaches, as the name suggests, involve testing an interface with a sample of representative users in an appropriate context. There are many variations on this theme, ranging from controlled laboratory testing to field-based explorations derived from anthropological methods.

Another categorization of the evaluation process, according to the stage in the development life-cycle where it may take place, is formative and summative evaluation [34]. In culinary terms, the difference between the latter could be explained as "when the cook tastes the soup, that's formative; when the guests taste the soup, that's summative" (Robert Stakes). In other words, summative evaluation takes place at or near the completion of the development cycle, whereas formative evaluation enables evaluation to begin even before there is a product to test and can continue late into the development process before the product is released.

2.1.1 Expert-based evaluations

The term *usability inspection* applies to situations where experts "inspect" or "examine" usability-related aspects of a product [27]. If expert-based evaluations are used in the early stage of product development, problems of design, flow and content can be located quickly, making the necessary changes, to remedy them, relatively easy.

The most common types of usability inspection are heuristic evaluations and cognitive walkthroughs. These have gained widespread acceptance because they are inexpensive, do not require special equipment or a usability lab, can be integrated easily into

the product development lifecycle, and finally provide quick results as the identified problems and the respective suggested recommendations are generated without delay [27].

Heuristic Evaluation

Heuristic evaluation is the most popular of the usability inspection methods. Nielsen describes heuristic evaluation as one of two "discount" usability methods, the other being usability testing with a small number of participants. Using a small set of evaluators, typically three to five (figure 2.2), produces a high degree of overlap in their findings [24].

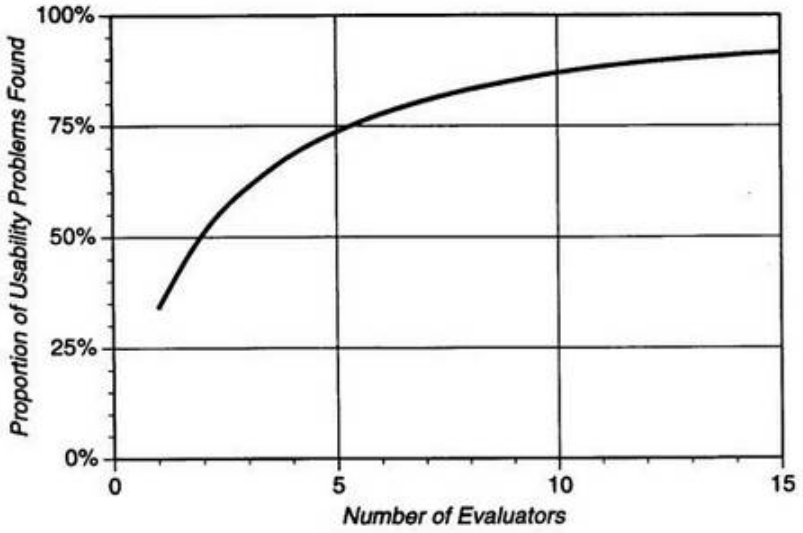

Figure 2.2: The proportion of usability problems found according the number of evaluators participating in the heuristic evaluation.

In a heuristic evaluation, each evaluator works alone to inspect the product against a set of rules or principles, known as *heuristics*. Since evaluators are not necessarily using the product to perform real tasks, heuristic evaluation may be used for interfaces that exist only on paper, allowing the inspection to take place early in the product development cycle.

The results of each evaluator's individual inspection are collected and documented in a report, either by an observer or by the inspectors themselves. The report produces a list of usability problems with explanations of the principle violated by each one. Although it is not necessary to include a list of recommendations, these are frequently obvious. For example, if the evaluator determines that the system does not provide feedback to the user at a critical point, the apparent solution to the problem would be to do just that. A debriefing session with the developers following the heuristic evaluation allows for quick solutions to be generated for the less obvious problems.

Cognitive Walkthroughs

A cognitive walkthrough is a usability inspection method designed to evaluate "ease of learning, particularly by exploration" [38]. In a cognitive walkthrough, the evaluators assess the interface in the context of tasks that users would perform.

Although cognitive walkthroughs focus on one aspect of usability (ease of learning) they naturally uncover issues related to ease of use and the functionality of the application. The method finds inconsistencies in the designers' plan and the users' use of the product. These could be related to poor word choice (a word or term that doesn't match the user's vocabulary), inconsistent word choice, or lack of feedback when an action is performed.

The main advantage of cognitive walkthroughs is that they help to identify problems with the design very early in development, before the design is ready to be tested by actual users. The key disadvantage, on the other hand, is that the process is effective only when the evaluators are trained in cognitive psychology or the process of the cognitive walkthrough. Untrained evaluators produce poor results, as research shows when comparing findings by software engineers not trained in the discipline to cognitive psychologists with background and training in the discipline [38].

2.2 Accessibility

An accessible web site is a site that can be perceived, operated and understood by users despite congenital or induced disabilities [33, 35]. Accessibility evaluation includes, but is not restricted to, assessing conformance to accessibility standards. Conformance to accessibility standards is important as in some cases it's a legal requirement and in others it provides a good way to help check if the design and implementation of a web site adequately addresses the range of accessibility issues [14].

Effective accessibility evaluation includes both evaluation expertise and the experience of people with disabilities. The participation of people with disabilities (when these are available, such as employees in the same building) in informal evaluations on low-fidelity design prototypes helps identifying serious accessibility hazards early on in the development cycle. An alternative way is the involvement of accessibility specialists (i.e., an accessibility expert with first-hand experience of how people with different disabilities interact with products). These specialists can:

- Evaluate accessibility issues for a broad range of users, which might not be found by a few individual users in usability testing,
- Help fix any known accessibility barriers before bringing in users, and

- Focus usability testing or informal evaluation with users on potential areas of concern.

Accessibility standards and guidelines are available from international standards organizations, national, state and local governments, industry groups and individual organizations. Most web accessibility evaluation tools assess how web pages conform to W3C WAI Web Content Accessibility Guidelines (WCAG) [38], and sometimes national standards such as Section 508 [17].

Software tools are available to help evaluate web pages and some elements of software. Since most of the aforementioned guidelines are not written in a formalized manner, different tools may have different "interpretations" of what these rules mean [7]. Therefore, while the tools provide some automated review, human evaluation is still necessary. Most of the tools are commercially available, a few are free, and several have limited functionality available free online.

Although evaluation tools can identify some accessibility issues, they alone can not determine if a product meets standards and is accessible. A good example of what tools can and can not do is evaluate equivalent alternative (alt) text for images on a web page. Tools can identify images that are missing alt text. However, they can not determine if existing alt text is indeed equivalent (i.e., determine if the alternative text provides the same information in text as the image provides visually). Judging if the alt text is equivalent requires human evaluation.

Web accessibility evaluation tools can increase the efficiency of evaluation by saving time and effort; however, they can not replace knowledgeable human evaluators.

2.3 Inspection tools

The number of web sites today is still growing and a large amount of web sites (private or commercial) exist that, concerning usability aspects, are very badly designed. It is well known that the average quality of websites is poor, "lack of navigability" being the #1 cause of user dissatisfaction [23]. The pressures of design place demands on HCI professionals to come up with fast answers, and cognitive scientists have worked on problems of improving test method reliability and validity. Current emphases include deriving better expert-based evaluation methods to overcome the rather poor validity of such methods (testers employing these methods tend to overestimate the number of problems users actually experience, that is, they label as problems many aspects of interfaces that users subsequently perceive as acceptable) [8].

Similarly, effort has been spent trying to package formal methods into tools that can be used effectively by non-cognitive scientists to predict usability. The objective of this

approach is to develop software tools which designers would use to calculate the learning effort or time to perform a task, without the designer having to know the details of how such an estimate is derived. The analogy is frequently made to the use engineers can make of the principles of physics. To date, few such tools have made the transition from research laboratory to design practice [8].

For the purposes of this master thesis, inspection tools will be divided into three groups according to the degree of automation they demonstrate. Inspection tools in their most traditional form are paper-based, such as questionnaires. They are based on a well-established usability evaluation practice and consolidated knowledge is available on each working step, from choosing the questions to assessing the results. However, these methods require caution in selecting the right questions and in processing the results.

In terms of automation, the exact opposite of paper-based inspection tools would be fully automated tools, which aim to provide software support for evaluation. Nevertheless, in the same ways in which we do not always accept the results of a spell and grammar check, inspection requires more than just automated tools. It requires human judgement. [39].

Finally, the third group are semi-automated inspection tools, which aim to provide a compromise between the two latter groups. They offer some level of automation, facilitating the inspection process. However, inspectors perform the actual inspection and reach to specific conclusions, thus eliminating the need to validate the results provided by an automated inspection tool.

2.3.1 Paper-based inspection tools

User questionnaires are one of the most typical and consolidated tools to evaluate user interfaces [30]. They can give valuable feedback from the user's point of view, but they must satisfy some important requirements. As a general rule, questions should be well formulated, i.e., clear and significant for the evaluation context. Moreover, results should be carefully analyzed and interpreted. A well-designed questionnaire can give good insight into the problems of the tested application, even if very detailed information is difficult to obtain.

CELLO

CELLO is a paper and pencil tool derived to a large extent from the expert-based heuristic method promoted by Jacob Nielsen. It is similar to heuristic or expert evaluation, except that it is collaborative, in that multiple experts, guided by a defined list of design criteria, work together to evaluate the system in question [10]. The criteria may be principles, heuristics or recommendations which define good practice in design and are likely to lead to high quality in use.

CELLO can be used throughout the development lifecycle, but it is most useful when applied early. At the conclusion of the inspection, an evaluation report is created that details how specific functions or features of the system contravene the inspection criteria, and may provide recommendations as to how the design should be changed in order to meet a criterion or criteria.

CELLO is a fast and simple first step towards usability testing. It relates strongly to well-understood practices, such as the use of design guidelines or principles and specialist third-party consultancy. On the other hand, user trials in context will give more specific information about the defects of a particular application. CELLO does not provide metric output, rather only design feedback information is provided.

Questionnaire for Interaction Satisfaction (QUIS)

Quis is a tool developed by the University of Maryland, and has been designed to access users' subjective satisfaction with specific aspects of the human-computer interface. It contains a demographic questionnaire, a measure of overall system satisfaction along six scales, and hierarchically organized measures of eleven specific interface factors (screen factors, terminology and system feedback, learning factors, system capabilities, technical manuals, on-line tutorials, multimedia, voice recognition, virtual environments, internet access and software installation) [15]. Each area measures the users' overall satisfaction with that facet of the interface, as well as the factors that make up that facet, on a 9-point scale. The questionnaire is designed to be configured according to the needs of each interface analysis by including only the sections that are of interest to the user.

Software Usability Measurement Inventory (SUMI)

SUMI is a generic usability tool comprising a validated 50 item paper-based questionnaire in which respondents score each item on a three-point scale (i.e. agree, undecided, disagree) [31]. SUMI measures software quality from the end user's point of view. The questionnaire is designed to measure scales of:

1. Affect – the respondents' emotional feelings towards the software (e.g., warm, happy).
2. Efficiency – the sense of the degree to which the software enables the task to be completed in a timely, effective and economical fashion.
3. Learnability – the feeling that it is relatively straightforward to become familiar with the software.
4. Helpfulness – the perception that the software communicates in a helpful way to assist in the resolution of difficulties.

5. Control – the feeling that the software responds to user inputs in a consistent way and that its workings can easily be internalized.

SUMI provides an objective way of assessing user satisfaction. Because SUMI scores are based on a standardized questionnaire, SUMI results can be compared across different systems. SUMI is mentioned in the ISO 9241 standard as a recognized method of testing user satisfaction.

The results produced by SUMI are only valid if the sample used is representative of the user population, if the questionnaire has been administered in the same way to all users sampled, and if the results are carefully interpreted. Experience interpreting the results of SUMI outputs is essential. Questionnaires can only provide information of a general nature; they do not identify specific problems which can be related to designers.

System Usability Scale (SUS)

SUS is a 10-item questionnaire that employs a Likert scale to obtain an overview of user satisfaction with software [6]. It was developed by John Brooke to:

- Provide an easy test for subjects to complete (i.e., minimal number of questions).
- Be easy to score, and
- Allow cross-product comparisons.

Measures of effectiveness and efficiency are context specific. Effectiveness in using a system for controlling a continuous industrial process would generally be measured in very different terms to, for example, effectiveness in using a word processor. Thus, it can be difficult, if not impossible, to answer the question “is system A more usable than system B”, because the measures of effectiveness and efficiency may be very different. However, it can be argued that given a sufficiently high-level definition of subjective assessments of usability, comparisons can be made between systems.

SUS is generally considered as a means of carrying out comparisons of usability between systems. Because it yields a single score on a scale of 0-100, it can be used to compare even systems that are outwardly dissimilar. This one-dimensional aspect of the SUS is both a benefit and a drawback, because the questionnaire is necessarily quite general.

2.3.2 Semi-automatic inspection tools

Semi-automatic inspection tools attempt to make the job of evaluators easier. They are partially automated, i.e., they automatically carry out tasks that do not need any active involvement from the users / inspectors in the first place (e.g., calculation of average scores, aggregation of comments, etc.). Furthermore, they usually guide evaluators through the

inspection process by a step-by-step procedure (typically documented in a theoretical framework).

Multidisciplinary tool at the Evaluation & Training Portal of the Finnish Virtual University (FVU)

This tool was developed to address the fact that any given evaluator is unlikely to be an expert in all the fields of a science needed in evaluation [32]. It is aimed at the evaluation of web-based learning environments. The tool takes a systematic account of the most important factors of accessibility, informational quality, usability and pedagogical usability. The usability section included sections to evaluate visual design, the use of multimedia elements, technical issues, support for online reading and navigation, error prevention and support for recovery from errors. All sections include around 4-12 criteria.

To find out the degree at which a particular criterion is met, the evaluator has to answer several questions (around 5-10). Answers range between 1 (poor) and 5 (excellent) or N/A (not applicable). Evaluators may also add relevant comments. The questions on each criterion are presented to the evaluator in a semi-intelligent form. It is possible for the evaluator to filter out questions which are irrelevant for certain systems.

After the evaluation, the tool produces a report on the results, composed of the overall profile of the web-based learning environment, a summary of the good features and guidelines on how to develop the particular learning environment.

The evaluation is partly subjective. In other words, the better the evaluator knows the substance, learning material and learning environment, the better the evaluation will be.

Systematic Usability Inspection Tool (SUIT)

SUIT is an internet-based tool that supports the evaluators during the usability inspection of software applications. The inspection technique underlying SUIT is the Systematic Usability Evaluation technique (SUE) [18], proposed to overcome the drawbacks of heuristic evaluation. SUIT makes it possible to reach inspectors everywhere, guiding them in their activities. Inspectors can perform asynchronous peer reviews of their inspection works in a discussion forum. The inspectors are coordinated by an expert inspector who has the role of the manager of the entire inspection process. SUIT can also support other inspection techniques, such as heuristic evaluation.

SUIT has been designed to support the evaluators performing usability inspections, trying to overcome time bottlenecks due to paper-based activities and face-to-face meetings. SUIT is an Internet application developed with open-source technologies. All features are implemented with dynamic web pages. Event notification is performed by automatic generation of emails. All structured and persistent data are stored in a database. SUIT uses

general-purpose applications, such as web browsers and email readers, for client-side communication and coordination.

Usability Problem Inspector (UPI)

UPI [1] is based on the User Action Framework (UAF), which classifies problems based on user interaction activities. Its goal is to help inspectors conduct a highly focused inspection on a target application, resulting in a list of usability problems that users will potentially have with the application.

The UPI method is an expert-based usability inspection method, such as the traditional kind of usability evaluation performed in a usability laboratory with users as participants. In the UPI, the evaluator plays both roles by conducting the inspection and also representing the vehicle for identifying problems that users will potentially have in the application (figure 2.3). UPI uses HTML and Active Server Pages.

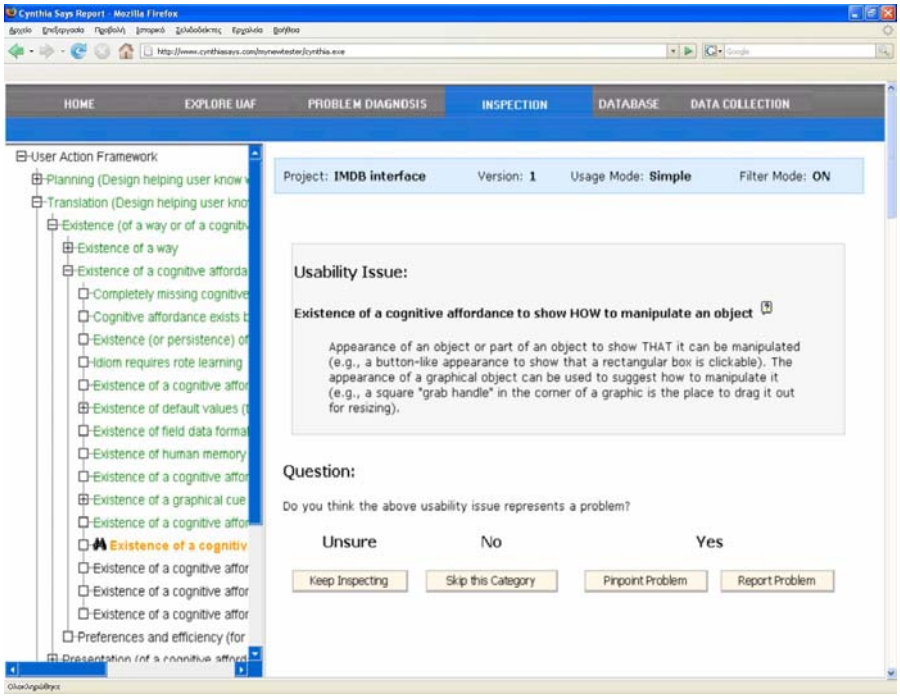

Figure 2.3: Screenshot of the UPI tool during an inspection.

As an inspection tool, UPI brings together aspects of both the heuristic evaluation and cognitive walkthrough. UPI intends to capture the ease of use from the heuristic evaluation, while also providing interaction-based structure as in the cognitive walkthrough. However, unlike heuristic evaluation, UPI provides more specific explanations of the problems because of the organized structure of usability concepts and issues found in the UAF.

2.3.3 Automatic evaluation tools

Automatic tools examine source code of web pages to derive adherence to universally accepted stylistic and objective guidelines. This approach is more recent and specifically bound to the characteristics of hypertext / mark-up languages used to create web pages.

Web Static Analyzer Tool (WebSAT)

The Web Static Analyzer Tool is a prototype tool that inspects the HTML composition of web pages for potential usability problems [22]. WebSAT allows the usability engineer to investigate these potential problems so as to determine whether they should be eliminated from the design of the web pages.

WebSAT inspects the HTML composition of web pages against numerous usability guidelines.

It can perform inspection using either its own set of usability rules (a set of heuristic rules, grouped into six categories as follows: accessibility, form use, performance, maintainability, navigation and readability) or those of the IEEE Std. 2001-1999 according to the specifications in its P2001/D8.01 Draft. In either case, WebSAT expects as input the URL of a single web page or of an entire site. The time required depends on the previous choice and on the number of pages that comprise the site.

Bobby

Watchfire Bobby [43] is a web accessibility desktop testing tool designed to help expose barriers to accessibility and encourage compliance with existing accessibility guidelines, including Section 508 of the U.S. Rehabilitation Act and the W3C's Web Content Accessibility Guidelines (WCAG).

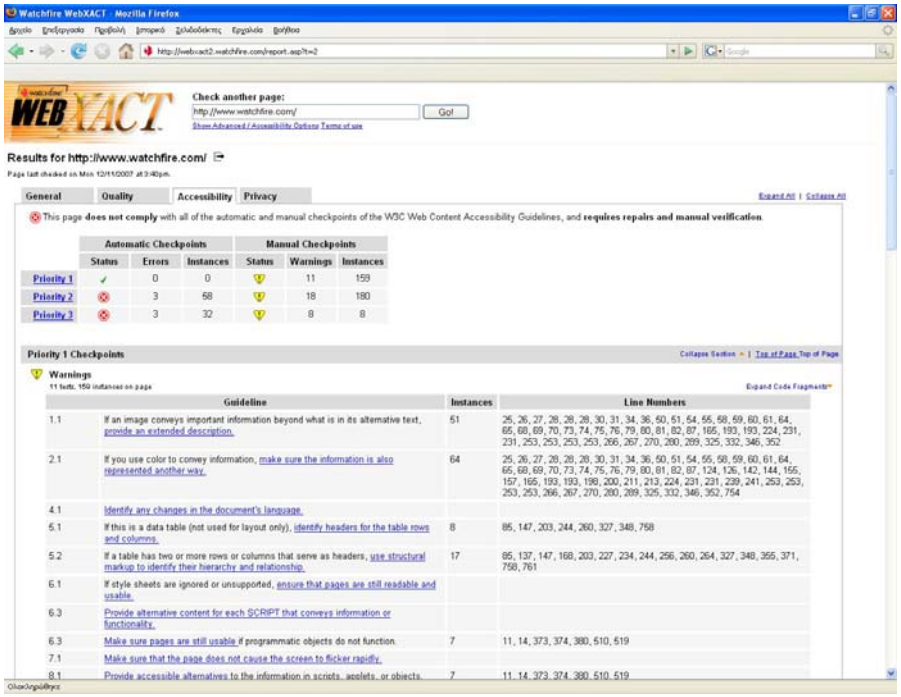

Figure 2.4: Structure of reports on a web site's accessibility, generated by Bobby. Errors and warnings are presented by ascending priority.

Bobby spiders through a website and tests to see if it meets accessibility requirements, including readability by screen readers, the provision of text equivalents for all images, animated elements and audio and video displays. During a scan, Bobby checks HTML against selected accessibility guidelines and then reports on the accessibility of each page. Reports are usually long, but well structured (figure 2.4).

Wave

WAVE is an acronym for "Web Accessibility Versatile Evaluator" and is a guideline review tool with support for automated critique. Human effort needed for tool usage is minimal.

This tool is implemented as a web service and in order to evaluate a web site, one needs only submit the URI of the online web site or the web site files [41]. There are four possibilities to do this.

1. The user visits the WAVE web site and types the URI of the web site to be evaluated into the specified form.
2. The user selects local files of an offline web site using a special button on the WAVE web site and uploads them for evaluation.
3. The user accesses WAVE through a plug-in for the web browser.
4. The user adds a special WAVE bookmark to the browser, which will launch the evaluation of the currently seen web site.

WAVE uses two famous sets of guidelines, namely the Web Content Accessibility Guidelines 1.0 (WCAG 1.0) [38] and the U.S. government regulation Section 508 guidelines [17]. Both sets focus on accessibility, and especially on making web content accessible to people with disabilities. But these guidelines also improve usability for people without disabilities, e.g., they help users find information more quickly.

The WAVE tool also helps to find actual and potential accessibility problems that arise due to poor HTML syntax (e.g., missing page elements) and it provides suggestions and descriptions on how to fix these problems. These suggestions are quite short and in some cases they could be insufficient for people with limited knowledge about web development.

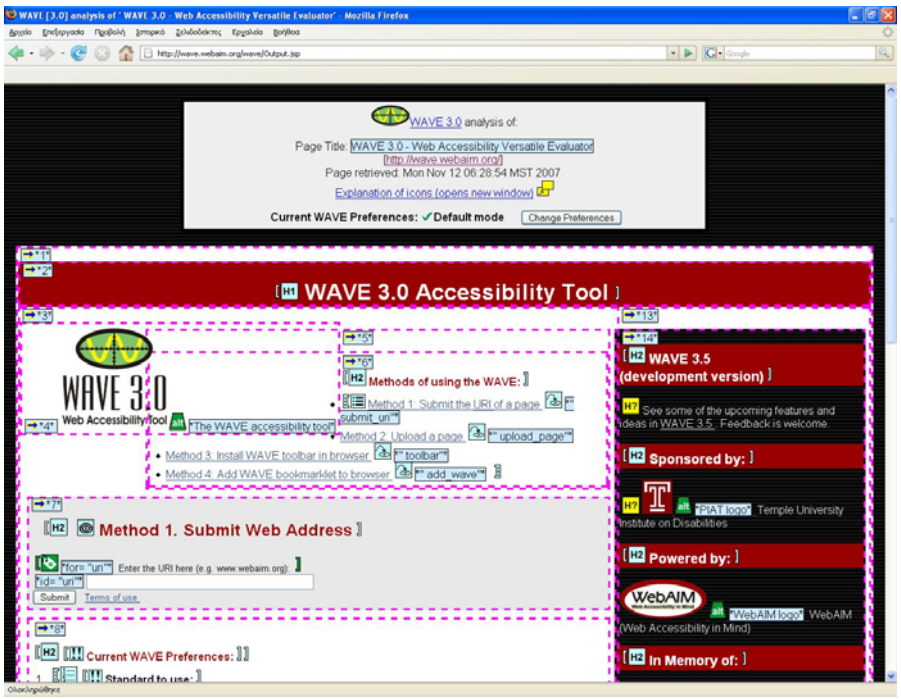

Figure 2.5: WAVE demonstrates results by marking problems directly on the original web site using icons.

Problem occurrences are marked directly on the original web site using icons, which are clearly explained (figure 2.5). WAVE mainly checks accessibility. This influences usability, but other usability aspects, such as consistency and information organization are not addressed by this tool.

Cynthia Says

The HiSoftware Cynthia Says web portal is a web content accessibility validation solution [40]. It is designed to identify errors in a web page's content related to Section 508 standards and/or the WCAG guidelines. Results of the evaluation are presented in a report detailing the accessibility errors in the page (figure 2.6).

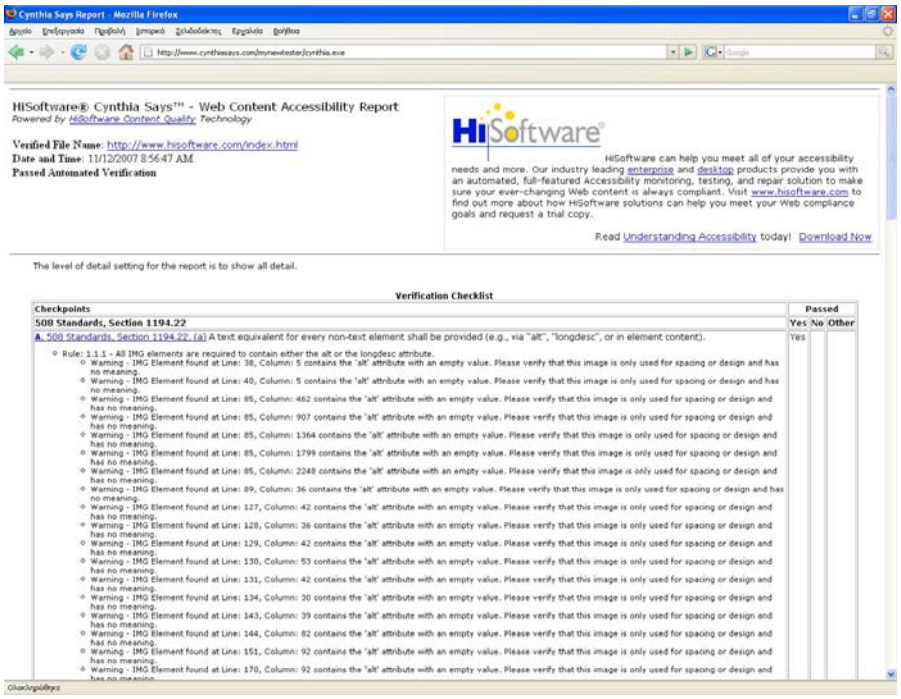

Figure 2.6: Structure of reports generated by Cynthia Says validation tool.

At the top of the report the name, date, pass or fail notification, and browser used are displayed and underneath them, is the verification checklist. This checklist is based on the accessibility guidelines the user had selected before they submitted the page. Each checkpoint in the checklist lists the accessibility guideline used to check the page and if specific elements of the web page passed or failed. The greatest disadvantage of Cynthia Says is that it can validate only one page at a time.

2. 4 Discussion

Certain qualities and characteristics that an inspection tool should demonstrate are becoming apparent taking into consideration the presentation of the most common inspection tools in 2.3.

As far as automation is concerned, the golden section (i.e., semi-automated tools) seems to be the optimal route. They offer more speed than paper-based tools and at the same time are more reliable than fully automated tools. Besides this, an obvious requirement for ORIENT would be that it addresses both usability and accessibility issues.

ORIENT will be based on the User Experience Evaluation Framework, a well-documented theoretical framework, thus inheriting the strengths of the inspection method described in the framework. Inspecting a system with ORIENT will follow the step-by-step process defined in the framework, which means that users of ORIENT are going to be guided

through the process making it easier for them to complete their assessment. Even if users have no prior experience with the method or the inspection tool, they will be able to inspect a system following the step-by-step process.

Although the method described in the User Experience Evaluation Framework calls for usability experts in the best case scenario, this does not exclude other users from participating as inspectors. After all, the selection of novice, moderate or expert users in an evaluation depends on the target users of the system under assessment [46].

ORIENT will produce both qualitative and quantitative results. Reports of the inspection tool will follow the pattern specified by the framework, which will make them well-structured. Problems will be presented in full text divided in groups according to the five system attributes that the framework uses for the inspection. Furthermore, scores for the identified problems are going to be available both in numerical form and in summative tables encoded in colour for easier recognition and comprehension of the results. This presentation of results in two forms, full-text (qualitative) and scores (quantitative), facilitates the production of design recommendations by a group of designers and, at the same time, makes the comparison between two systems evaluated with the same tool easier. Finally, summative results are produced partially automatically by the system, making their analysis more accurate (i.e., minimizes the possibility for human error).

Moreover, the User Experience Evaluation Framework uses a small set of system attributes (5) for its inspection as opposed to paper-based inspection tools that rely on questionnaires of up to 100 questions.

Finally, most inspection tools that were presented in 2.3 (with the exception of SUIT) do not support the communication between the members of the inspection team. Even SUIT supports the exchange of messages by means of an e-mail client (i.e., an external to the inspection tool application). Therefore, a useful addition to ORIENT would be the ability for members to contact one another from within the inspection tool. Such a feature would allow for inspection team members to be located in great distances and still cooperate on an inspection, as communication would be integrated in the actual inspection tool. Furthermore, the evaluation of certain systems is sometimes impeded by the fact that the target user population includes different cultures that the geographically secluded inspection team finds extremely difficult to simulate. Making ORIENT web-based and enriching it with features such as those of an online community (member profiles) solves this problem. Even if an inspection taking place in Greece required the participation of Chinese users, the solution would lay just a few clicks away.

3 Underlying Framework

3.1 The User-experience Evaluation Framework

Nowadays, computer technology is transcending into an empowering expression and communication medium with an increasing pervasiveness in the entire sphere of human activities (work, leisure time, entertainment, education and training, etc.). The potential benefits are clear. Accessible, fast, cheap, personalized and efficient information and service delivery for all, including people with different cultural, educational, training and employment background, novice or experiences users, the very young and the elderly and people with different types of disability.

Despite significant efforts worldwide, reportedly, the realization of the electronic services vision has proven hard and the offered systems show controversial degrees of success. Both practitioners and researchers have a strong interest in understanding why people may resist using computers, in order to develop better methods for designing technology, evaluating systems and predicting how users will respond to new technology [12]. Previous research has identified a number of reasons why "customers" use, or do not use, a computer-based system. Utility and usability, for instance, have long been considered by the scientific community and practitioners as salient system adoption factors. Accessibility is another key determinant for the acceptance of a system. Similarly, other important factors in this respect include, but are not limited to, findability (the ease or difficulty that potential users have in finding the type of system that they are interested in) and affordability (the degree to which potential users can afford the cost to access and use a product or service). Unfortunately, such aspects influencing a system's acceptability are hardly stressed out in today's perspectives and approaches, and rarely, if not at all, addressed holistically.

Typically, traditional evaluation methods and techniques are introduced in late development stages of diverse user interfaces and adapted on a case-by-case basis [5]. Unfortunately, such limited evaluation approaches are often proved inefficient and ineffective in assessing accessibility or other system qualities – such as utility and usability – of systems. Admittedly, the evaluation of modern UIs, which are more oriented to the public and diverse users and contexts of use than ever before, requires rigorous methods and systematic approaches.

The User-experience Evaluation Framework [20] is targeted to measure the degree to which the user needs and requirements are met throughout the user experience lifecycle, and allows for assessing user perceived qualities, such as visibility-findability, perceived usefulness prior access and use, availability-approachability, interaction qualities (i.e., qualities perceived

throughout the actual usage of the system and interaction with its UI, e.g. accessibility, utility and usability) and user relationship maintainability, at various depths, including at the level of system and system parts such as system functions, interaction controls, etc. Among the important aspects of the framework in question is that this model incorporates accessibility as a basic determinant of acceptability and long-term adoption of interactive technologies [3], it is generic and is claimed to apply to all types of (computer) products and services, including universally accessible systems, as well as systems especially developed for people with disability.

3.2 Measuring user experience

User-experience (figure 3.1) is measured by the extent to which:

- The product is made visible to non-users (visibility),
- Non-users are motivated to gain a personal experience of the system (perceived usefulness & ease of use),
- Actual users find it easy and acceptable to reach the product (availability / approachability),
- Actual users find it useful, easy and acceptable to interact with the product (quality of interaction experience),
- Previous users are motivated to become long term users (relationship maintainability and subjective usefulness & ease of use),
- Product users are not offered more promising and satisfying alternatives (competitiveness).

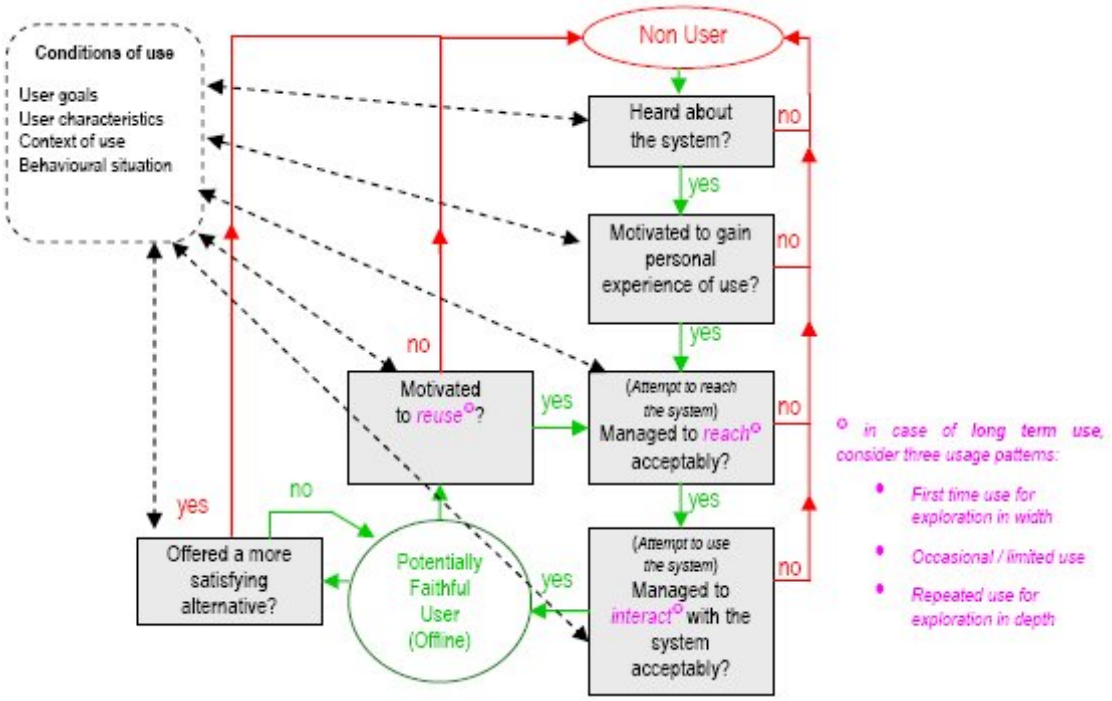

Figure 3.1: Overview of the user-experience evaluation framework.

Therefore, user-experience reflects the overall product quality perceived by users [11] of the following product qualities: visibility, perceived usefulness and ease of use, availability / approachability, quality of interaction experience, relationship maintainability and competitiveness (figure 3.2).

Visibility refers to the degree to which a system can become known to individual non-users. Obviously, the actual location of the system is a major visibility factor. Furthermore, visibility can be increased by providers through publicity strategies.

Perceived usefulness and ease of use refer to the usefulness and ease of (access and) use of the system from the viewpoint of individual non-users. These are related to the available information regarding the product and to the extent to which the product appears to be suitable with respect to the user's particular goals and needs. This also comprises a variety of tangible aspects, such as time and cost savings resulting from the product itself (rather than the way it is delivered).

Availability refers to the degree to which all types of potential individual users can reach the entry point(s) of the system. Certainly, accessibility (e.g. for anyone, at any time, from anywhere) of the carrier / storage medium of a system is a major factor for its availability / approachability.

Quality of interaction experience encompasses the quality of interaction perceived by actual individual users and refers to the degree to which a system can be used to achieve useful and quality results (i.e. lead to subjective satisfaction).

Relationship maintainability (subjective usefulness and ease of use) refers to the degree to which a good relationship with individual system users is effectively cultivated and maintained while the user is not working on the system (e.g. by means of informing the user for new functionality, content updates, changes of status, etc.).

Competitiveness is the degree to which the system is conceived by individual users to be more appropriate for them than other available alternatives.

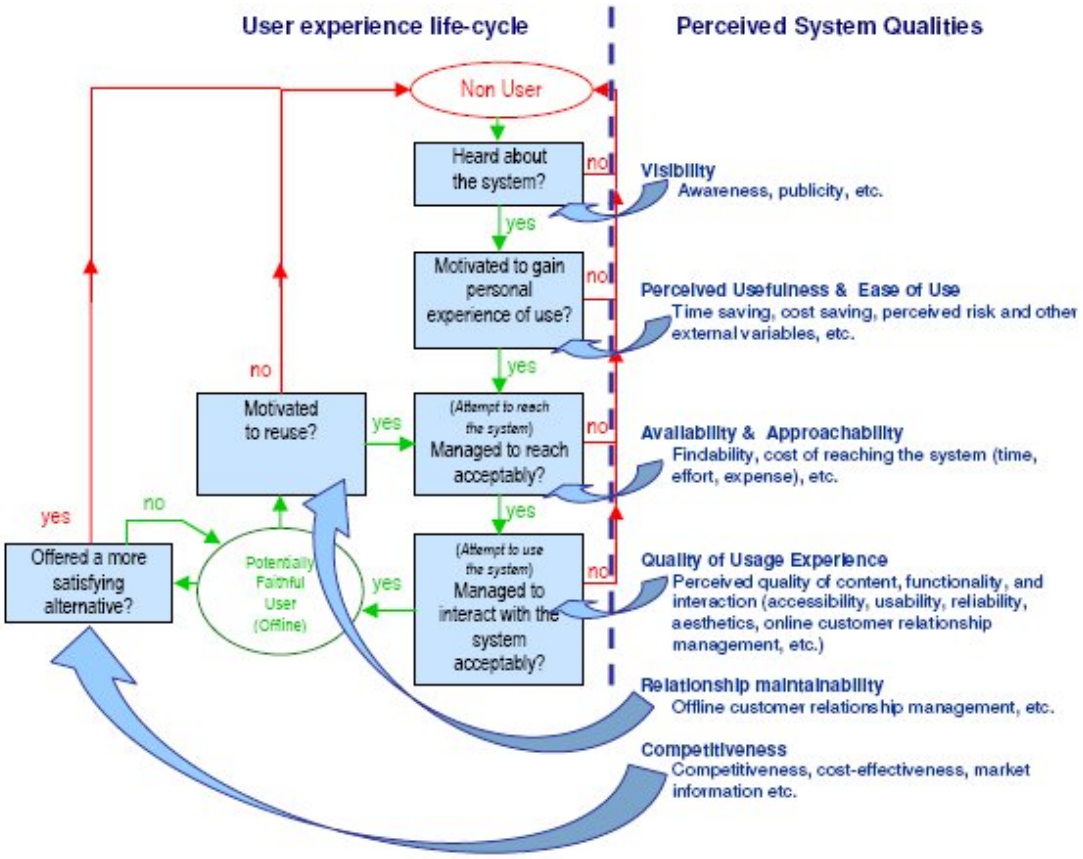

Figure 3.2: User experience lifecycle vs. user perceived system qualities.

In practice, taking into account accessibility in the usage experience lifecycle of a product or service means assessing the possibility that each individual user group (with different characteristics and requirements) has to interact with the system both as a first experience and in the longer-term. In terms of UI and user dialogue with the system, accessibility can be defined as the extent to which the sequences of input actions of a product, and the associated feedback that lead to successful product use, are possible to be performed by the user, with respect to the individual's limitations emerging from the particular conditions of use. In other words, accessibility ensures that an individual can use a product, whereas usability ensures that the individual finds it easy and satisfying to use it [21].

As mentioned earlier, the quality of interaction experience of a system (i.e., the quality of the user interface) can be perceived as the aggregate of the user-experience of the

system's individual functions that are relevant or important to the individual user. In other words, when moving deeper into the evaluation of subsystems and system functions, the framework can be iteratively applied to each corresponding UI (both physical and virtual).

In this perspective, a function of a computer-based system (e.g., of an eService) can be perceived as a system itself and thereby be assessed in terms of visibility (of the function) to non-users, perceived usefulness and ease of use to non-users (i.e. prior using the function), availability / approachability to willing users (i.e. prior using the function), quality of interaction and relationship maintainability. For example, offer the user the option of accessing and using complementary functions, offer the user the option of storing summary reports on his interactions (e.g. cost, statistics, etc.) and provide reminds and notifications.

In general, in assessing a UI and in order to claim high levels of overall user-experience, (a) each function needs to demonstrate a highly degree of user-experience individually and at the same time (b) an analogy needs to be achieved between the importance of each function to the user and the corresponding levels of user-experience of each individual function.

In these terms, the model can by slightly modified to introduce a paradigm shift from "acceptance" levels (borderline) to higher levels representing strong potential for user adoption (above borderline, i.e., more competitive levels) and thus from "potentially" to "likely" faithful users, respectively.

In general, the same model applies at various system levels, such as:

(a) clusters of systems,

(b) stand-alone systems,

(c) system sub-components,

(d) system functions, devices, interaction controls, etc.

Finally the framework can be employed effectively in evaluations, both expert and user based, of systems that are aimed to offer accessibility and usability to all (e.g., public systems) or of systems that are specifically developed for people with disability.

The User Experience Evaluation Framework has been instantiated in the paper-based ORIENT methods and inspection tool [21], which constitutes the basis of the tool developed in the context of this thesis.

3.3 The paper-based ORIENT Method and Inspection Tool

Overall, the inspection team initially seeks basic information about the system from the product providers. If necessary, the inspection team shall seek additional information from representative end-users of the system to be inspected. Finally, once the inspection is concluded, the inspection team, through the leader, provides feedback to the corresponding

product providers regarding identified good design and delivery practices, as well as design and delivery pitfalls and suggestions for overcoming them.

The outcomes of the inspection produced by means of ORIENT are mainly a list of problems identified along with their corresponding severity scores, but also, potentially, recommendations for fixing problems and thereby improving the user-experience of the system under question.

Overall, the inspection procedure by means of ORIENT involves the following phases “Pre-inspection”, “Inspection set-up”, “Inspection” and “Reporting”, further detailed in the following subsections.

Firstly, the general and specific objectives of the inspection need to be identified; secondly, the limitations of the inspection need to be specified, including available time, budget and number and expertise of inspectors; then, taking into consideration the objectives and limitations that are posed, a preliminary inspection plan of the objectives, timings and human resources and expected results can be synthesized; finally, the team of inspectors needs to be assembled, both in terms of expertise and number, and all should be given the appropriate guidance in order to commence the evaluation.

The inspection set-up phase (figure 3.3) comprises mainly of the collection of background information regarding three different aspects, namely inspection background information, information about the system and assembly of the respective context of use. During this phase, an initial approach of the system takes place in order to identify the system’s target users and categorize them into User Groups, as well as accumulating a prioritized set of system functions for each Group. Depending on the desired depth of evaluation, which has been set in the preparation phase, it is possible to further break down the system to cover sub-functions or smaller interaction items. The final step, which is also one of the most fundamental of the entire process of evaluation, involves collecting and analyzing the Conditions of Use for each User Group.

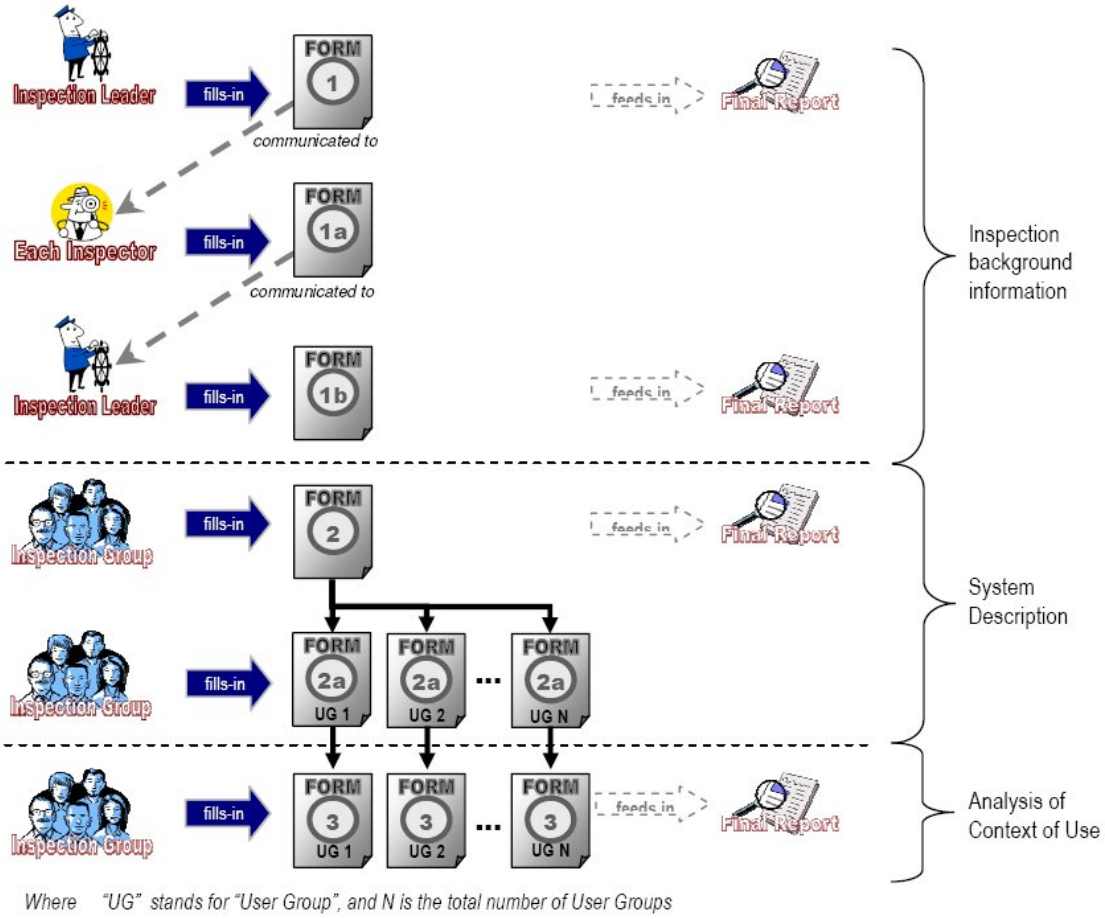

Figure 3.3: Graphical representation of the steps included in the set-up phase.

The inspection is conducted by each Individual Inspector separately, following a step-by-step process to inspect how the distinct system features may influence the users (divided in user groups according to their goals and needs). Each step of the evaluation should take under consideration the corresponding user requirements as these have been reported in analysis of the context of use in order to estimate the positive or negative impact of each feature. Each inspector gives a severity rating for each feature identified taking into consideration three variables: the frequency (is the feature rare or common?), the impact (will it be difficult for users to overcome/exploit the feature?) and the persistence (is this a one-time feature that users can overcome/exploit or will they be asked to put extra effort repeatedly?). The ratings range from positive values in case of good practice (max. 4) to negative values in case of identified problem (max. -4). The zero value here indicates borderline acceptance levels, suggesting that users will just accept to proceed to the next lifecycle stage.

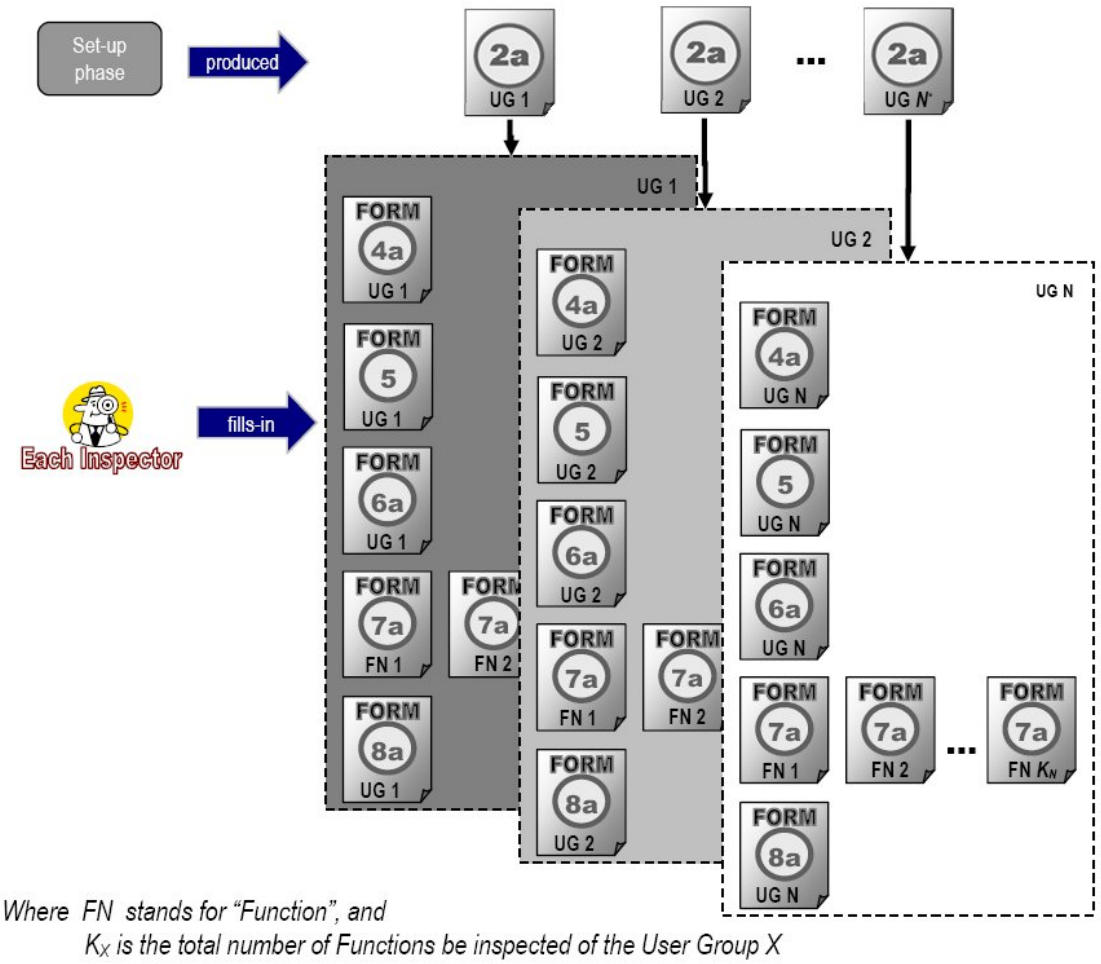

Figure 3.4: Graphical representation of the steps included in the inspection phase.

The actual process of evaluation (figure 3.4) can be separated into two stages: one for evaluating the system as a whole, based mainly on extrinsic evidence but also on intrinsic system characteristics, and one for "zooming" inside the system and examining its distinct functions, per user group. Thus, initially each inspector is asked to investigate to what degree the system is appropriately introduced to each target group and what image the users form of the system, before actually using it (or even seeing it). First the whole system's visibility is assessed [Visibility (per Inspector) – (form 4a)]. After the inspection of the system's visibility, inspectors proceed to the next extrinsic characteristic, the system's perceived usefulness and ease of use [Perceived usefulness and ease of use (per Inspector) – (form 5a)]. The next stage of the evaluation process is about locating the system and reaching its entry point (e.g. homepage) whether this is a first time user or a repeater one (Availability & approachability (per Inspector) – (form 6a)]. Subsequently, the next stage involves inspecting 'physically' the system starting with individual user groups and the respective functions selected to be assessed [Function's user experience (per Inspector) – (form 7a)]. After all functions have been assessed inspectors should have formed an overall opinion of the system

and should be able to judge whether users are likely to reuse the system [Relationship maintainability (per Inspector) – (form 8a)].

An overall note for the assessment procedure is that a distinction is made between first time and novice users, moderate users and expert users whether inspecting the whole system or separate functions. The rationale behind this is that a system's or function's barriers or facilitators to use are reflected differently upon users with varying degrees of expertise and practice with the system. Thus, the versatility of the inspection instrument is increased since not only are inspector comments tailored to different types of users but also in the occasion that a comment applies to more than one type it may be scored differently depending on the impact it has on users.

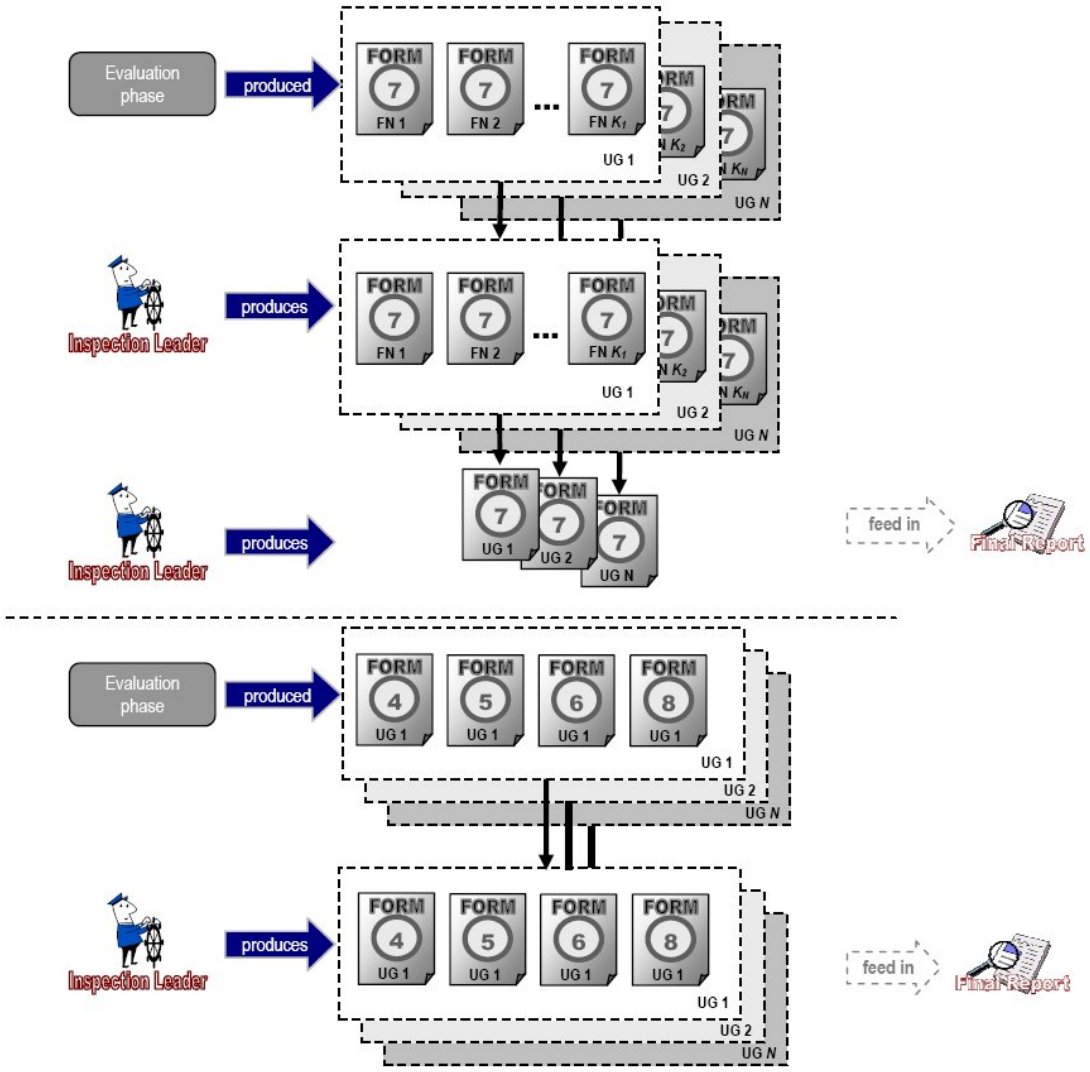

Figure 3.5: Graphical representation of the steps included in the reporting phase. The inspection leader produces summative forms from the inspectors' individual forms.

After all individual Inspectors have examined the system and given their comments and scores for its features, the Inspection Leader needs to debrief each Inspector involved

and, with the assistance of the Data Logger, produces the summary forms (figure 3.5) in the corresponding folders [forms 4, 5, 6, 7 per User Group and function, 7b per User Group and 8]. Thus, reporting forms are drawn up for each user group and finally overall summaries are produced (figure 3.6) reporting on all user groups to display the overall user-experience of the system [forms 9 and 10].

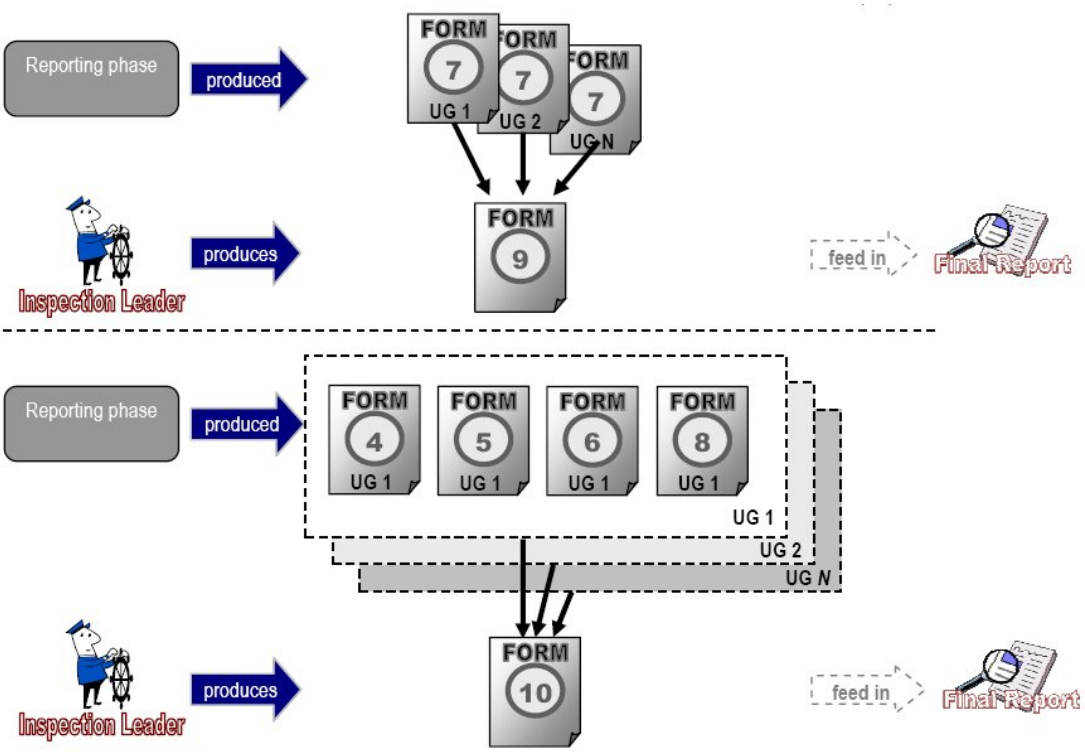

Figure 3.6: The inspection leader creates graphical representations of the study's quantitative findings in coloured tables using collective data available from respective summative forms.

In certain forms, apart from the actual scores, tables also use colour coding to facilitate their interpretation by readers from the first glance by using cool colours to indicate positive values (good practice examples) and warm colours to indicate negative values (user-experience problems).

3.4 Pilot application of the ORIENT inspection tool

The paper-based inspection tool ORIENT was used extensively in the evaluation of a sample of ten online services from both new and old EU member states, indicative of all service domains (eGovernment, eHealth and eLearning) for the purposes of the eUser project [2].

The goal of the inspection was not to make a comparison between the ten eServices at random in order to select the best among them, but to perform a pilot testing of some of the

most popular public electronic services and to reveal to what extent their design took user needs into consideration, as well as to what degree they actually help fulfill the needs of their target user groups.

The tool's pilot application produced concrete and analyzable results (quantitative and qualitative) on the match between demand and supply in relation to user needs in the online public services domain. ORIENT's ability to serve the goal of cross-referencing service characteristics with user characteristics enabled the review and assessment of both generic user factors (common issues across clusters of services and / or user groups) and specific factors (characteristics of particular services and / or user groups).

The assessment produced results and conclusions on the readiness and capacity to provide user-centered online public services through the implication of two types of outcomes: appraising comments, highlighting the instances of good design and user-experience pitfalls and deficiencies.

The findings of this evaluation may help derive the overall image of public eServices in Europe, as they are perceived by users.

4 Design of the ORIENT Online Inspection Tool

4.1 Analysis of the user requirements and derived system requirements

4.1.1 Interviews with developers and users of the method and tool

The first step of the designing process for the ORIENT online inspection tool was to conduct a series of short interviews with the developers of the method and tool, as well as a small group of people who have had prior experience with the paper-based prototype of the inspection tool in order to collect information about how they would envision the online equivalent of the inspection tool, and what characteristics and capabilities they would like it to have.

All users of the method and of the paper-based inspection tool agreed that, though the procedure of the framework was simple enough for anyone to carry out, the cost in terms of the time spent was not proportional to the results. In other words, a considerable amount of the time invested by a person in the method was wasted in organizing, moving (copying from one folder to another) and duplicating (copying from one part of a form to another part of the same or a different form) information. The need for the system to take up most of these tasks and carry them out automatically in order to facilitate the user was obvious.

Another aspect that troubled the users was the fact that there was no overall organization of the data they had produced with the paper-based inspection tool. Even if they conformed to the guidelines of the method for organizing information into folders, there would still be no easy way of accessing information quickly within the same or among different studies. Furthermore, access to the aforementioned information by a third person would require that person knowing the exact physical location of the respective folder, as well as having access to that folder (or the general system or group of systems it belonged to).

One of the most difficult and time-consuming processes when using the paper-based inspection tool is the analysis of the context of use per user group. Usually, sufficient information about the cultural differences between citizens from different countries or regions is not available. Moreover, a common analysis is reusable and reduces the need for (technical) expertise in future inspections. As a consequence, the system would have to support the storage of such information for easy reuse in the future.

Another disadvantage of the paper-based inspection tool, according to users, was the need to maintain identification attributes. For example, whenever a user would move from

one step of the evaluation process to another, they had to re-enter information such as the name of the system and their personal reference id in order to identify the respective form in a unique way. Such actions could easily be carried out by the system automatically, without the user even having to be aware of it.

Finally, according to the interviewees, the automation of the reporting phase of the inspection process (wherever possible) would greatly facilitate, enhance and make the inspection's completion faster. Besides this, it would protect users from potential errors while moving information from all the forms to the collective forms.

4.2.1 User groups

In the field of human – computer interaction, software of any type should, as a principle, meet basic standards for usability. However, given the diversity of the field, it is only common sense that an equally diverse group of people should be recruited to undertake the task of evaluating a system. Inspection teams may include usability experts, designers, developers, end users from the target user population, as well as third party participants who may not make actual use of the system, but are indirectly affected by it.

Thus, deriving from the interviews conducted and the actual specification in [21], the following user groups were identified:

- Product providers
- Inspection leader
- Inspectors
- Administrator (*of the inspection tool*)
- Designers
- Target users *(of the system to be inspected)*

Although there are references in the specification, which could result in more user groups (e.g., inspection administrator, data logger, etc.), they were not taken into account. The reason for this was twofold:

(a) As stated in the specification, these roles could be undertaken by the inspection leader of one of the individual inspectors.

(b) Most (if not all) tasks that people assigned with these roles would be called to carry out can easily be undertaken by the inspection tool itself (e.g., calculation of average scores).

In the following sections, each user-group will be presented by briefly listing its characteristics, the functions they perform (in relation to the inspection tool) and requirements they (may) have from the inspection tool.

Product providers

The term "product providers" is used to refer to the people responsible for the development, operation and promotion of the product to be inspected. More specifically, product providers can be:

a. **Product executives**, i.e., decision making people such as marketing managers, development managers, etc. They can provide significant information regarding the objectives of the system, the target user groups, the dissemination strategy, etc.
b. **Product developers**, which consist of user interface and graphic designers, programmers, content providers, etc., who can provide significant information regarding the structure, the functionality, the content management of the system, etc.
c. **System operators**, who are responsible for the installing and launching of the product, for operating and maintaining the software, as well as for data manipulation and user support.

Product providers, as can easily be concluded from the aforementioned, provide the corner stone of the inspection process, as they provide the system which will be inspected, as well as most (if not all) of the necessary information about it, its target users, context of use, etc. This type of information is the raw material fed as input into the set-up phase of the inspection.

However, product providers aren't expected, as explained in the specification of the framework [21], to make use of the inspection tool directly. Therefore, their requirements are of generic nature and focus on ways of enhancing the inspection process (i.e., the inspection tool enables rapid evaluation of a system and can produce concise and precise reports of good and bad design practices).

Inspection team members - Inspector leader

The User Experience Evaluation Framework allows for multiple levels of familiarity with the inspection method or similar inspection methods and tools in general. As a result, inspection team members are not required to have prior relevant experience, though should this were the case, such experience would greatly benefit the inspection team and process alike.

Although an identified requirement was the inspection tool to be as intuitive and self-explanatory as possible, inspection team members are expected to have a basic understanding of interaction techniques sufficient to be able to operate the inspection tool. Moreover, team inspection members should be moderately fluent in the use of English, as this is the language used in all the ORIENT forms.

Ideally, the inspection team would be comprised of evaluation experts. However, the role of inspector could also be assigned to a student, a designer, a developer or even a simple

computer/internet user. Consequently, characteristics such as age range, etc., are expected to vary significantly for the inspection team.

The most experienced member of the inspection group usually undertakes the role of inspection leader. Thus, he/she becomes responsible for overseeing the inspection process, incorporating the inspection results and composing the final report of the inspection, based on the aforementioned results. Moreover, the inspection leader may also function as an inspector, should the need arises.

In more details, an inspection leader's tasks include the following:

Table 4.1: Presentation of the tasks an inspection leader performs

Task id.	Task description	Data manipulation
T1	Manage a personal profile *(common function with inspectors group)*	Form 1a
T2	Carry out a study *(partially common function with inspectors group)*	All forms
T3	Manage my archive *(common function with inspectors group)*	All forms
T4	Track study's progress *(common function with inspectors group)*	All forms
T5	Create result tables	Forms 4, 5, 6, 7, 7b, 8
T6	Change public access of a study	D/B data
T7	Manage my messages	D/B data
T8	Compose new message	D/B data
T9	Manage my contacts	D/B data
T10	Login / Logout of the system *(common function with all user groups of ORIENT)*	D/B data

Tasks T1 and T3 will be presented in more detail in the next section.

Task T2 is in fact a complex task, describing the entire process of performing an evaluation by means of the ORIENT inspection tool. It comprises 4 sub-tasks, namely **T2.1: *"Initiate a study"***, **T2.2: *"Set-up a study"***, **T2.3: *"Perform an inspection"*** and **T2.4: *"Report study's findings"***. Task T2 is partially common with inspectors group, when an inspection leader plays the role of an inspector as well. However, since the task of inspecting is by definition (and more often) assigned to inspectors, sub-task T2.3 will be presented in more detail in the next section.

In sub-task T2.1, the inspection leader establishes a preliminary description of the study (i.e., a brief presentation of the main attributes, such as the name of the system in question, the period of assessment, type of expected results, e.g., qualitative or quantitative, etc.) and assembles the inspection team, assigning a specific role to each member. Afterwards, in task T2.2, using information from various sources (e.g., the system itself, related press articles or releases, etc.), he/she describes the system to be assessed through a brief introduction to its background, main features, objectives and technical characteristics. Additionally, the system's target user groups are presented and respective functions for each user group are enumerated, specifying which will be assessed and which not. Finally, the context of use for the system in question is analysed and documented. The final sub-task of T2 is T2.4, when the inspection leader collects all information recorded throughout the study up until that point and sums it up in collective forms.

One of the main objectives of the inspection leader is to monitor the progress of the inspection process (task T4). However, every member of the inspection team may also keep themselves up to date with the study's process. Estimations of completion are made for the study as a whole, as well as for each member of the inspection team separately.

In task T5, the inspection leader processes numeric data that have been collected through a certain process during the inspection phase and calculates average values, which in turn are represented in tabular form accompanied with colour encoding to enhance comprehension.

Having completed a given evaluation, the inspection leader has the choice of making it public or not (Task 6).

Messages' and contacts' management is comprised of Tasks 7, 8 and 9 respectively. Finally, T10 is about the process of a user logging in (out of) the ORIENT inspection tool. A sub-task of T10 is the case where a user has forgotten / lost his / her password and wishes to retrieve it.

Having studied in detail the tasks an inspection leader is responsible for, certain requirements can be identified. For instance, there should not be any use of ambiguous terms in the forms or the control labels. The inspection leader should be able to monitor the progress of a task and easily return to a specific step of the corresponding procedure.

In the paper-based prototype of the inspection tool, explanatory guidelines were provided at the beginning of each form [21]. However, to better accommodate the users' needs, it would be preferable to provide help at any given step of the inspection (both in case-sensitive and in general topic form).

Certain actions that the inspector leader was required to complete on his/her own with the paper prototype of the inspection tool, can be easily assigned to the inspection tool. For

example, the system can retrieve individual scores and calculate average / total scores itself, without requiring any action from the user.

The inspection leader is presented with the demanding task of consolidating individual comments into collective forms. In order to facilitate him/her in doing so, all necessary information should be presented simultaneously and at the same time in a discreet way, so that processing it should be relatively easy and straightforward.

Finally, in the interest of error prevention, certain input data could be standardized. Namely, certain fields are expected to receive a value within a specific range, for example an inspector's id is a numeric, integer value, greater than zero, so selecting it from a drop-down list would be more efficient and less error prone than typing it.

Inspection team members - Inspector

The role of inspector can be assigned to anyone possessing a fundamental knowledge of interface design and evaluation, and a history of active involvement in the latter. Ideally, inspectors would be evaluation experts. It is suggested that a group of at least three inspectors is formed [21]. Furthermore, each inspector should preferably possess domain knowledge unique within the inspection team, thus forming a multidisciplinary group.

An inspector's main objective is to walkthrough the system's interface with the intention of identifying possible pitfalls according to the user experience evaluation framework, by means of the inspection tool. In order to accomplish this goal, an inspector will have to perform certain tasks, which are presented briefly in the following paragraphs.

Table 4.2: Presentation of the tasks an inspection performs

Task id.	Task description	Data manipulation
T1	Manage a personal profile *(common function with inspection leader)*	Form 1a
T2	Carry out a study *(partially common function with inspection leader)*	Forms 4a, 5a, 6a, 7a, 8a
T3	Manage my archive *(common function with inspection leader)*	All forms
T4	Manage my messages	D/B data
T5	Compose new message	D/B data
T6	Manage my contacts	D/B data
T7	Login / Logout of the system *(common function with all user groups of ORIENT)*	D/B data
T8	Track study's progress *(common function with*	All forms

inspectors group)

In task T1, the inspector is called to fill in a copy of the questionnaire for providing information about his/her profile. This form focuses on the inspector's professional expertise, relation to the system and to the system providers in question, familiarity with the system and with the inspection method, as well as with similar systems and inspection methods, and finally fluency in use of the language supported by the system and of English (as this is the language used in all the ORIENT forms).

As mentioned in the previous section, only sub-task T2.2 will be briefly presented in this section. The inspector inspects the system following a step-by-step process (as described in the specification of the framework [21]) and evaluates how the distinct system features may influence user experience. Initially, the inspector assesses the entire system's visibility (*form 4a*) and the system's perceived usefulness and ease of use (*form 5a*). Afterwards, the inspector evaluates the process of locating the system and reaching its entry point (e.g., homepage) whether this is a first time user or a repeater one (*form 6a*). Subsequently, the next stage involves inspecting "physically" the system, starting with individual user groups and the respective functions selected to be assessed (*form 7a*). After all functions have been assessed, an overall opinion of the system should have been formed, and the inspector should be able to judge whether users are likely to reuse the system (*form 8a*).

In task T3, the inspector has the ability to browse through a personal archive of completed studies they participated in.

Taking into consideration the tasks an inspector has to perform, certain requirements can be identified. For instance, there should not be any use of ambiguous terms in the forms or the control labels. The inspector should be able to keep track of the progress of an inspection and easily return to a specific step of the corresponding procedure (e.g., go back and modify certain elements of a form).

In the ORIENT paper-based prototype, instructions are presented at the start of each form and these instructions address possible questions the inspector may have in completing the specific form (i.e., a kind of case-sensitive help) [21]. However, to better accommodate the users' needs, it would be preferable to provide help at any given step (where "step" could substitute for inspection, form or a specific element within a given form) of the inspection (both in case-sensitive and in general topic form).

An inspector is required to fill in a great amount of information. However, a portion of the aforementioned information is available by other means (i.e., is available from some other form or can be easily calculated by information available in other forms). Thus, certain accelerators (e.g., the inspection tool fills in automatically known information in each form,

such as inspector's id, name, system's name, etc.) can significantly facilitate the inspection process.

Finally, as far as error prevention is concerned, certain input data could be standardized. For instance, certain fields can be filled only with a value within a specific range (e.g. each example of good / bad design practice receives a score ranging from -4 to 4). Therefore, selecting this value from a drop-down list would be more efficient and less error prone than typing it.

Administrator

The administrator of the inspection tool has the responsibility of operating and maintaining the database supporting the inspection tool. More specifically, his/her tasks revolve roughly around management issues of the database, such as creating / editing / deleting records of user accounts, studies, retrieval of passwords for specific accounts, etc.

The administrator is assigned a very specific role and consequently his/her needs are also very specific. First of all, the administrator has a small (yet significant) set of responsibilities and needs not be distracted by unnecessary information. Therefore, the administrator could greatly benefit from a simplified version of the user interface.

Furthermore, the majority of his/her functions, in case of error, may result in permanent loss of information, and therefore it is essential that there are error prevention mechanisms to ensure that any removal of data occurs intentionally and not by accident. A clear and effective presentation of the data stored in the database in combination with error preventing mechanisms should significantly aid the administrator.

Designers

Designers and other experts (i.e., user interface, interaction, graphics, development and other experts) are included in inspection teams when solutions for redesign are required.

However, just as product providers, designers are not expected to make direct use of the inspection tool. Thus, their requirements are of generic nature and focus on ways of enhancing the inspection process (i.e., the inspection tool enables rapid evaluation of a system and can produce concise and precise reports of good and bad design practices, the inspection findings are presented in a structured and easily understood way that facilitates the task of finding solutions to potential problems by designers, etc.).

Target users (of the system to be inspected)

In case available sources fail to provide the inspection team with sufficient knowledge about the target users, the inspection team may need to contact a sample of potential and possibly

actual system users, in order to acquire additional, vital information about their profiles, the real context of use, etc.

It is easy to conclude that target users have no direct relationship to the inspection tool and as such, do not bring any new requirements to this design process.

4.2 Database design

4.2.1 Folder hierarchy of the paper-based form of the ORIENT tool

The User-experience Evaluation framework [21] divides the inspection process into three parts, namely the inspection set-up phase, the inspection phase and the reporting phase (figure 4.7).

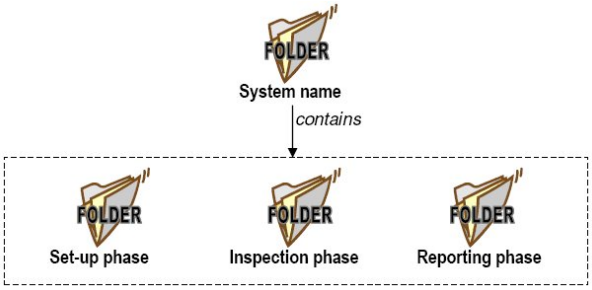

Figure 7: The three phases of an inspection according to the UOE framework

The inspection set-up phase focuses on the collection of information concerning the inspection background, the system description and the analysis of context of use (figure 4.8).

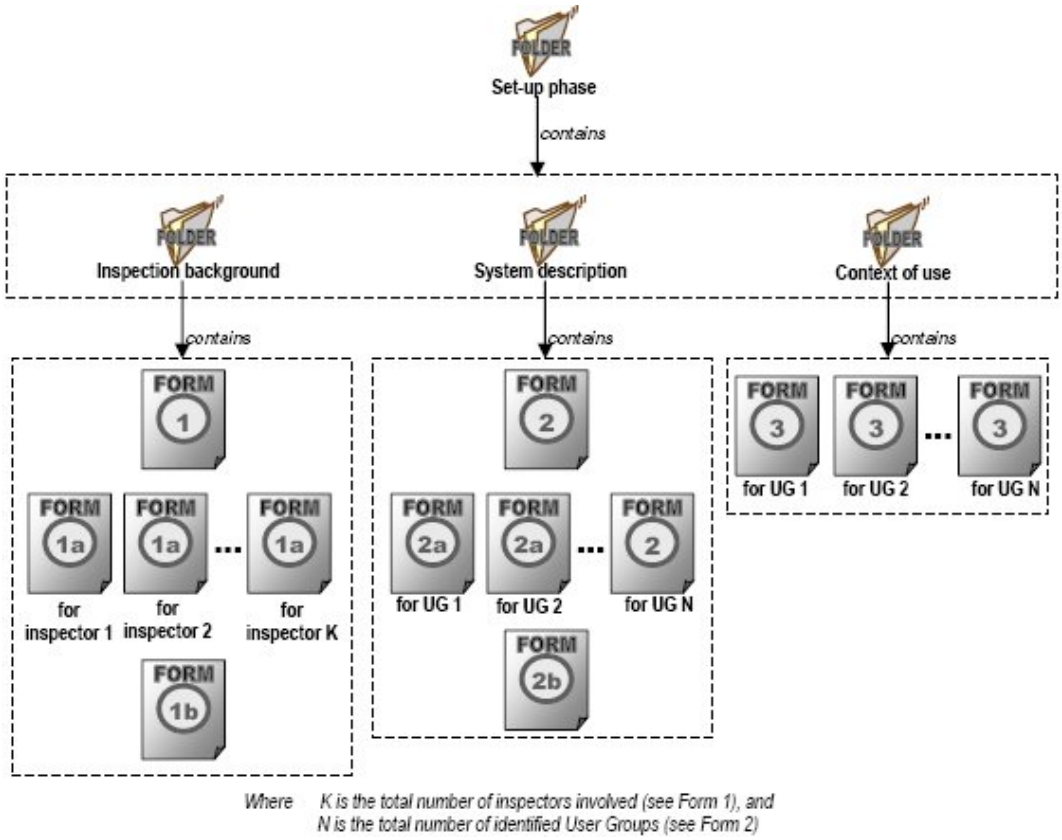

Figure 8: Contents of the Set-up phase folder

The second phase is about the actual inspection process (figure 4.9). The inspection is carried out by each individual inspector separately, following a step-by-step process to inspect how the distinct system features may influence the users (divided in user groups according to their goals and needs). The results of this inspection are documented into several forms and organised into folders, according to the proposed by the framework structure.

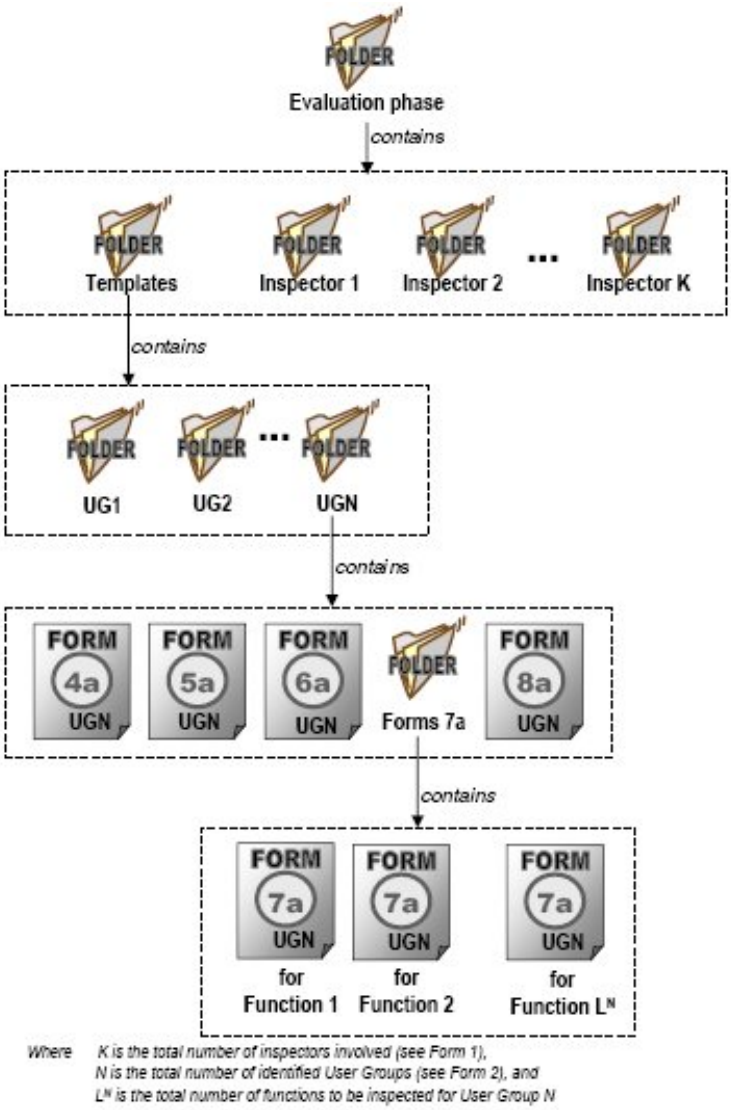

Figure 9: Contents of the Inspection phase folder

Finally, after all individual inspectors have examined the system and given their comments and scores for its features, the inspection leader produces the summary forms in the corresponding folders (figure 4.10).

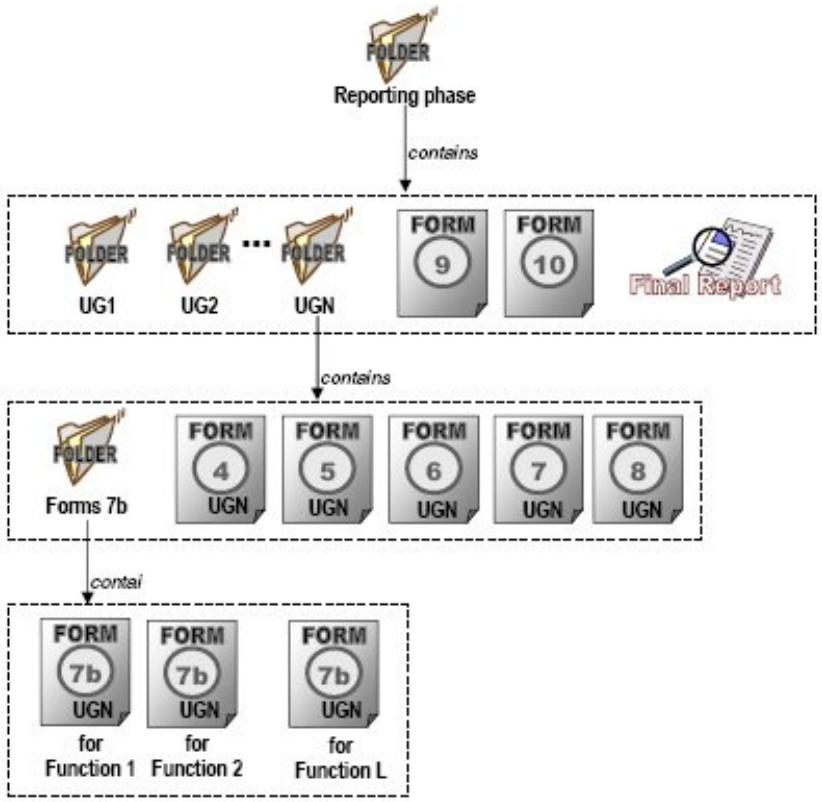

Figure 10: Contents of the Reporting phase folder

4.2.2 Database model

The briefly presented folder hierarchy points the design approach of the database into a specific direction. In this section, however, it will be described more thoroughly.

The database's purpose is to support the inspection tool, which in other words means to store and provide the information needed to perform an inspection and retrieve results and conclusions from it. To this end, certain entities and relationships between them were defined following the entity-relationship model [45].

First of all, there are three main concepts (the inspection team members, the forms involved and the actual inspection process) that make up the specific domain of interest. Team members and the inspection process were represented as two entities, namely **Inspection team member** and **Inspection**. The first one holds all relevant information to a person, whether they are assigned the role of inspector, designer or inspection leader. The second represents the actual inspection process for a specific system with specific inspection team members.

However, the existence of forms depends on the existence of instances of **Inspection team member** and **Inspection**, as there is no point in discussing about forms unless an actual team member fills them in and they help document an actual inspection. Therefore, forms

should be represented by a weak entity. However, different forms are associated in different ways to the aforementioned entities. For example, a specific instance of form 4a is linked to a specific inspector who created it and belongs to a specific inspection process, whereas a specific instance of form 9 corresponds only to a specific inspection process. As a consequence, forms were divided into three weak entities (**Set-up phase forms**, **Inspection phase forms** and **Reporting phase forms**), one for each distinct phase of the inspection according to the User-experience Evaluation Framework.

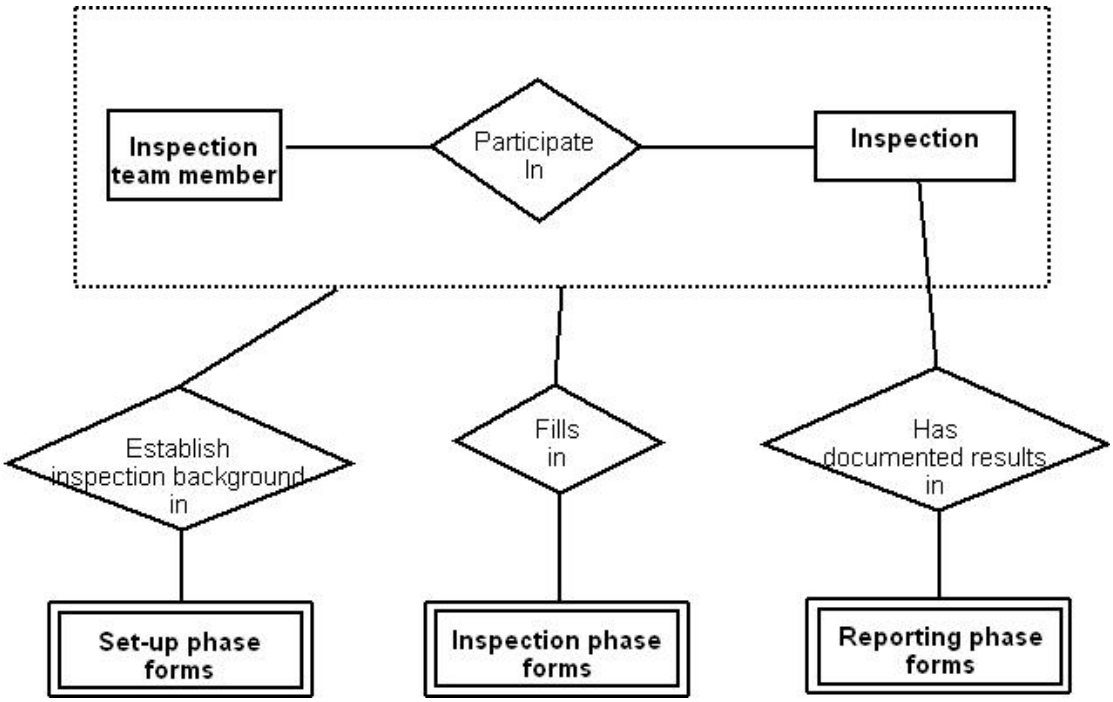

Figure 11: E-R model representing the relationship between the main entities

Each of the previous weak entities corresponds to several forms, thus defining IsA relationships. Besides this, certain forms are divided into two or more parts in order to better portray the intended information.

The **Set-up phase forms** entity contains forms 1b, 2, 2a and 3. Form 2 is divided into two parts: the system description and the potential user groups. Likewise, form 3 is divided into six parts, each of which documents user requirements induced by a different cause (e.g., user characteristics, task characteristics, user equipment characteristics, etc.).

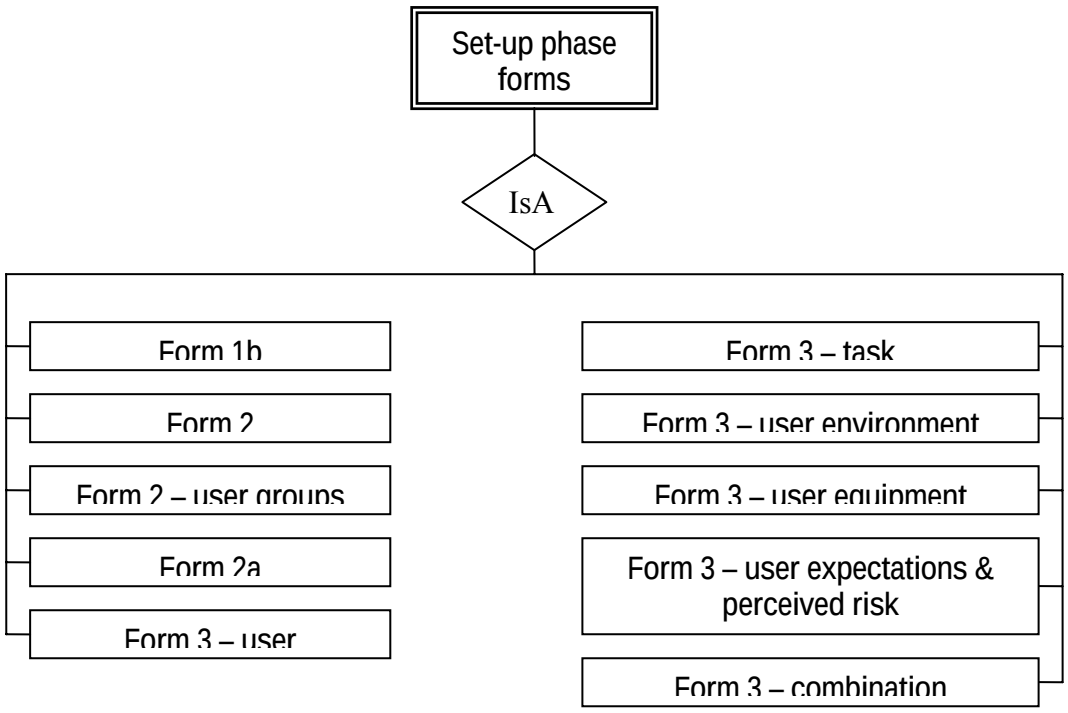

Figure 12: IsA relationship for Set-up phase form entity

The **Inspection phase forms** entity contains forms 4a, 5a, 6a, 7a and 8a. Form 6a is divided into three parts: one for the first-time and novice, one for the moderate and one for the expert users. Form 8a is similarly divided into three parts. Finally, form 7a is divided into 11 parts. The first 5 parts concern the first-time and novice users (visibility, perceived usefulness & ease of use, availability & approachability, quality of interaction and relationship maintainability), the next 3 concern the moderate users (availability & approachability, quality of interaction and relationship maintainability) and the final 3 parts address the expert users (availability & approachability, quality of interaction and relationship maintainability).

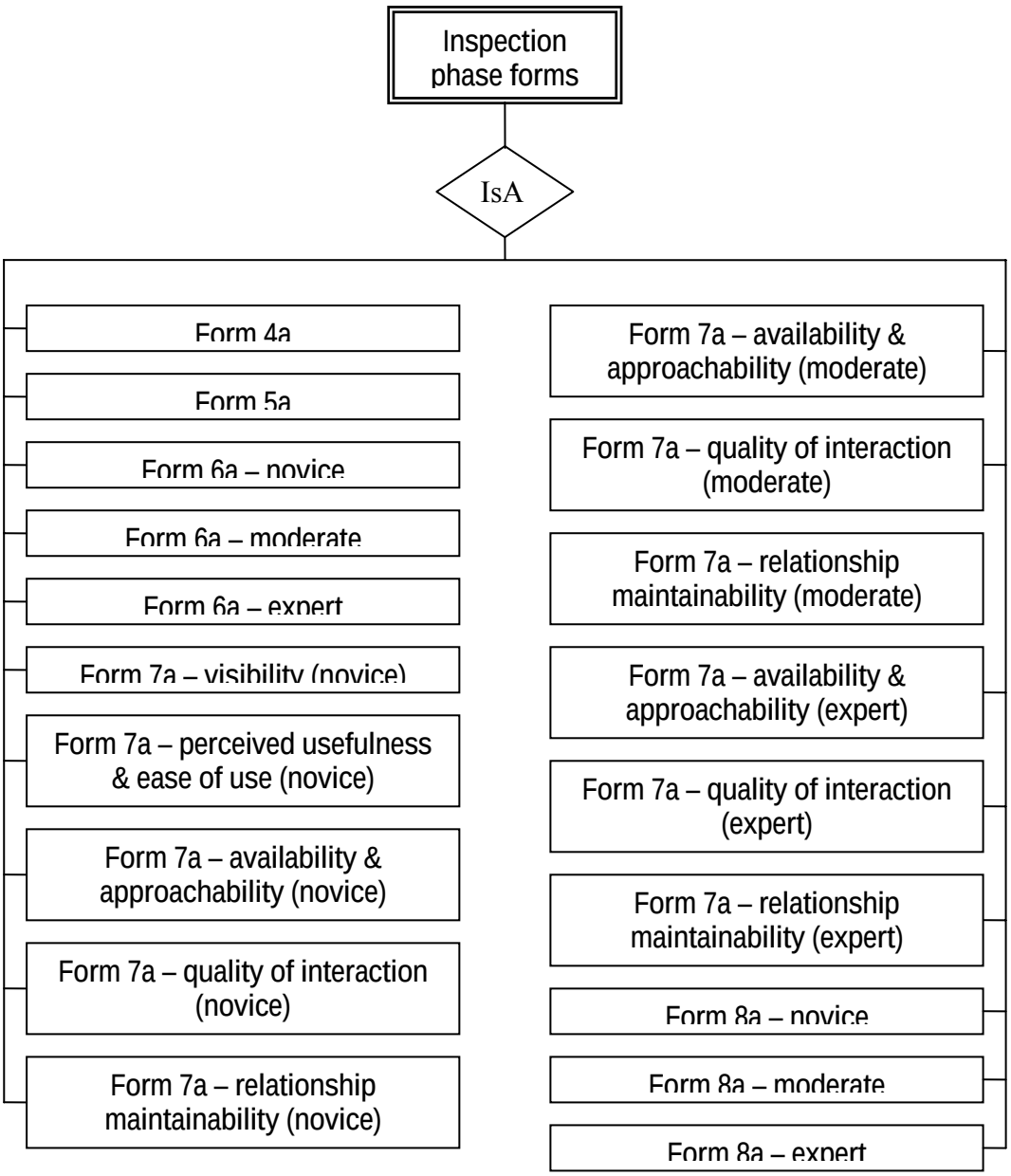

Figure 13: IsA relationship for Inspection phase form entity

The **Reporting phase forms** entity contains forms 4, 5, 6, 7, 7b, 8, 9 and 10. Form 6 is divided into three parts: one for the first-time and novice, one for the moderate and one for the expert users. Form 8 is similarly divided into three parts. Form 7 is divided into 11 parts. The first 5 parts concern the first-time and novice users (visibility, perceived usefulness & ease of use, availability & approachability, quality of interaction and relationship maintainability), the next 3 concern the moderate users (availability & approachability, quality of interaction and relationship maintainability) and the final 3 parts address the expert users (availability & approachability, quality of interaction and relationship maintainability). The remaining forms do not need to undergo any alteration.

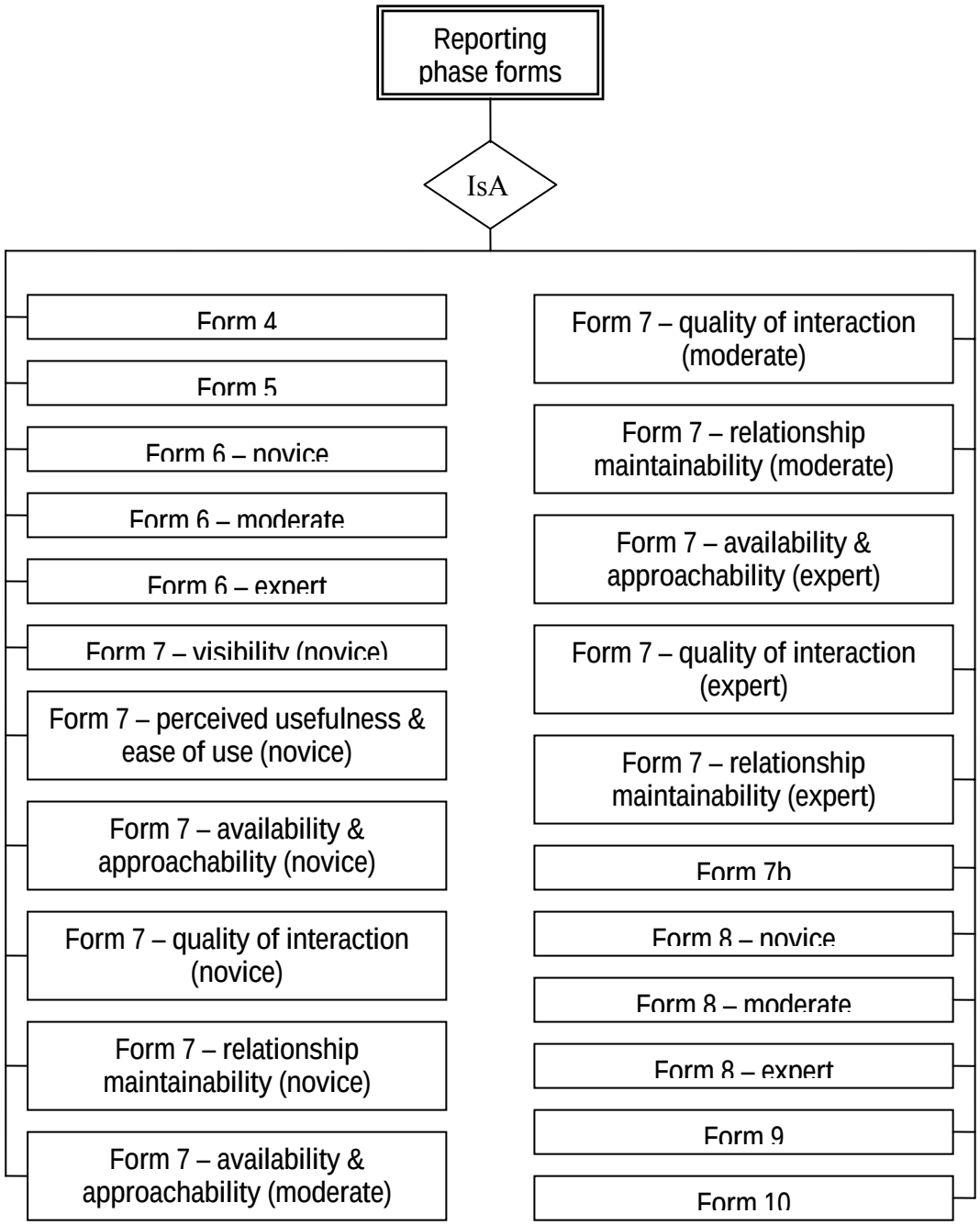

Figure 14: IsA relationship for Reporting phase form entity

In order to facilitate the managing of an inspection team during any given study, as well as to enable the team members (and users of the ORIENT online tool in general) to communicate with each other, two new concepts were introduced: **Contact** and **Message**.

The concept **Contact** is pretty much the same as that of an ordinary phone book, in effect a place of storage of contact information of people that are of use to a specific person in order to facilitate the process of the latter communicating with the former. In order to add a

person to one's contacts' list, that person has to first ***invite*** the second individual to join his contacts (***Invited Contact***). On acceptance of the invitation, each of the two persons is added to the other's contacts' lists. On refusal of the invitation, the person that declined seizes to be considered as an ***Invited Contact***. The entity **Inspection team member** is connected to itself through relationships **Invites** and **HasInContacts'List** (figure 4.15).

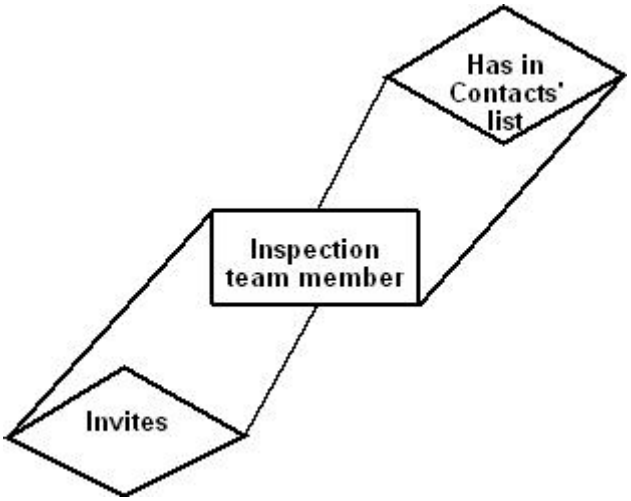

Figure 15: E-R model representing the retroactive relationships Invites and HasInContacts'List

The concept **Message** is very similar to the concept of an e-Mail, with the sole difference that the recipient(s) of a message within the limits of the ORIENT online tool are restricted to one's **Contacts**. Therefore, the entities Message and Inspection team member are connected through relationships **IsSenderOf** and **IsRecipientOf** (figure 4.16).

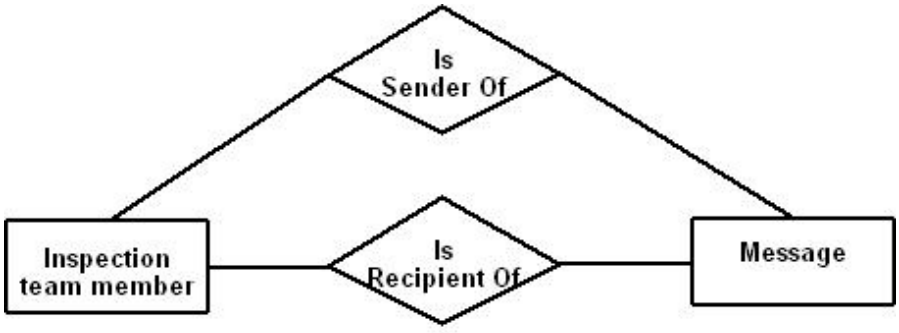

Figure 16: E-R model representing the relationship between the entities Inspection team member and Message

As one can easily notice in figures 5.6-5.8, forms 1 and 1a were not included in any of the IsA hierarchies. Forms 1 and 1a record information about the inspection and the inspectors respectively. However, certain data of form 1 (e.g., *name of the system*, *period of*

assessment, *objectives of the inspection*, etc.) provide a brief presentation of the inspection process as an entity (i.e., serve as descriptive attributes of the entity), whereas others (e.g., *inspection leader*, *inspector #1*, etc.) describe the composition of the inspection team and therefore are best protrayed as attributes of the relationship **ParticipateIn** between the entity **Inspection** and **Inspection team member**.

A portion of the information documented in forms 1a represents general profile information about the inspector as an entity (e.g., *native language*, *sex*, *background and expertise*, etc.), while another portion relates more closely to his/her actual involvement in a specific inspection (e.g., *familiarity with the system in question*, *familiarity with languages supported by the system*, etc.). The predicament is apparent. Should only one profile be stored in the database for each person, then every time it is retrieved, it certainly conveys current information (as a person can modify his/her profile at any given time). On the other hand, linking several instances of a person's profile to specific inspections in which he/she participated is optimal for providing accurate information about the person at the time period when the inspection was conducted. Adopting either of the above design solutions would trigger other problematic behaviours (i.e., loss of data accuracy in the first case and data redundancy in the second case). Therefore, a combination approach was adopted and general profile information became attributes of the entity **Inspection team member**, while information more closely related to a specific inspection were represented as attributes to the relationship **ParticipateIn** between **Inspection** and **Inspection team member**.

Besides the relationship **ParticipateIn**, three more are employed to help portray the connections between the entities. The relationship **EstablishInspectionBackgroundIn** connects forms from the set-up phase with a specific inspection conducted by specific inspection team members. Similarly, the relationship **FillsIn** identifies the inspector or inspection leader who filled in a specific form whilst participating in a specific inspection. Notably, certain forms can only be filled in by an inspection leader. Finally, the relationship **HasDocumentedResultsIn** helps maintaining a record of the findings of an inspection. The relationships' cardinalities are presented and explained in table 4.3.

Table 4.3: Cardinalities of the relationships

Entity Relationship **Entity**	**Cardinality**
InspectionTeamMember ParticipateIn **Inspection**	An inspection team member can participate in many inspections. Several inspection team members can be involved in an inspection. However, only one person may lead an inspection.

	(N-to-N or many-to-many)
[InspectionTeamMember participating in **Inspection]** EstablishInspectionBackgroundIn **Set-upPhaseForms**	An inspection team member can use several set-up phase forms in order to establish the background for an inspection. A set-up phase form can be filled in by only one inspection team member for a given inspection. **(1-to-N or 1-to-many)**
[InspectionTeamMember participating in **Inspection]** FillsIn **InspectionPhaseForms**	An inspection team member can fill in several inspection phase forms in order to carry out an inspection. An inspection phase form can be filled in by only one inspection team member for a given inspection. **(1-to-N or 1-to-many)**
Inspection **HasDocumentedResultsIn** ReportingPhaseForms	An inspection's results can be documented by several reporting phase forms. A reporting phase form can be used to store results for only one inspection. **(1-to-N or 1-to-many)**
InspectionTeamMember **Invites** InspectionTeamMember	An inspection team member may invite many inspection team members. An inspection team member may have received more than one invitations. **(N-to-N or many-to-many)**
InspectionTeamMember **HasInContacts'List** InspectionTeamMember	An inspection team member can have in his contacts more than one inspection team member. An inspection team member may belong to many contacts' lists. **(N-to-N or many-to-many)**
InspectionTeamMember **IsSenderOf** Message	An inspection team member can be the sender of many messages. A message can have only one sender. **(1-to-N or 1-to-many)**
InspectionTeamMember IsRecipientOf Message	An inspection team member can be the recipient of many messages. A message can be delivered to many recipients. **(N-to-N or many-to-many)**

4.3 Interface design

4.3.1 Hierarchical task analysis

A Hierarchical Task Analysis approach allowed the decomposition of the ORIENT's tool into sub-tasks by separating the steps of the process performed by a user, viewed at different levels of detail. Each step can be decomposed into lower-level sub-steps, thus forming a hierarchy of sub-tasks. The resulting task hierarchy can provide a better understanding of user requirements regarding interface design, allocation of duties, development of user support documentation and training.

As described in section 4.2, users are expected to perform a number of tasks by means of the ORIENT inspection tool. These include the following:

⇒ Manage a personal profile

⇒ Carry out a study

⇒ Manage my archive

⇒ Track study's progress

⇒ Create result tables

⇒ Manage messages

⇒ Compose new message

⇒ Manage my contacts

,which display a level of complexity and will be further explained by means of a hierarchical task analysis.

Manage a personal profile

The user can create a new profile from scratch (in case he/she is a new user to the system) or edit his/her existing profile.

In order to create a new profile, the user simply needs to supply certain required information. These are divided into three categories: information with the purpose to identify the user (e.g., first and last name, username, password, etc.), information that will ensure the safety of the user's account (e.g., security question & answer, used in case the user needs to recover a lost password) and information that will make up the user's profile within the ORIENT inspection tool (e.g., native language, sex, background and expertise, etc.). If the user wishes to edit their existing profile, they review the profile as it is, edit the desired pieces of stored information and verify the changes they made. Should the user want to change their password as well, they have to authenticate themselves by providing their current password.

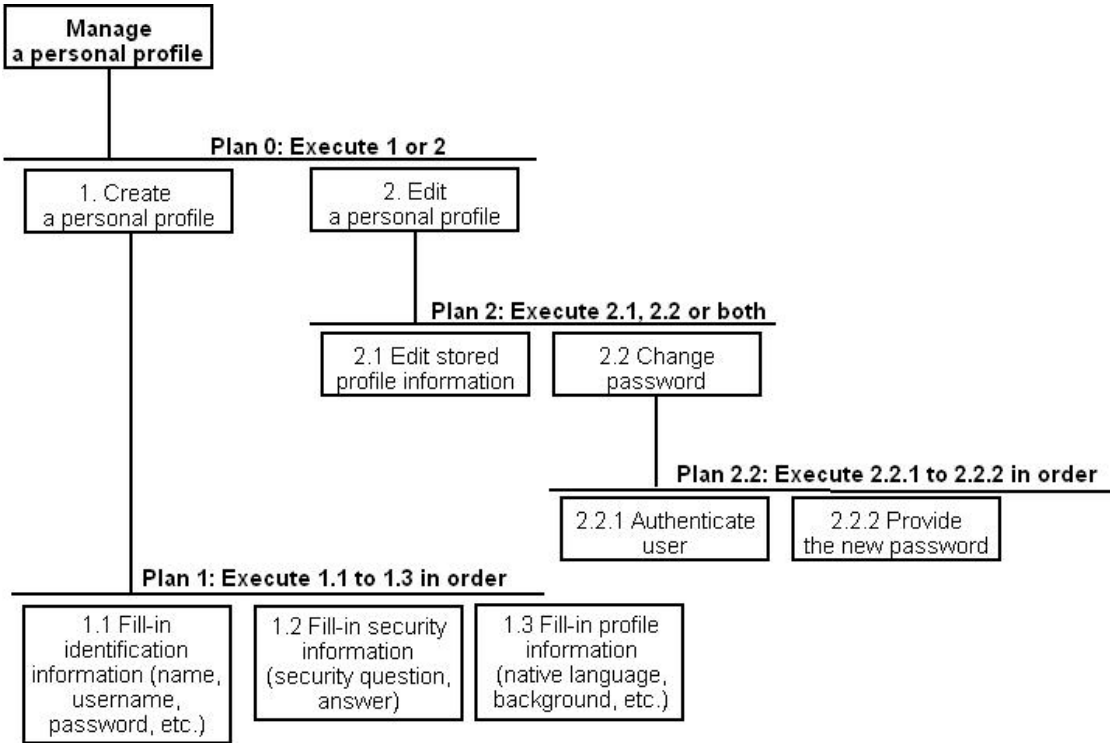

Figure 17: HTA diagram for task "Manage a personal profile"

Carry out a study

The function of carrying out an inspection consists of certain stages with specific steps being taken in every step. These are:

⇒ Initiate a study

⇒ Set-up a study

⇒ Perform an inspection

⇒ Report study's findings

⇒ Publish study (optional)

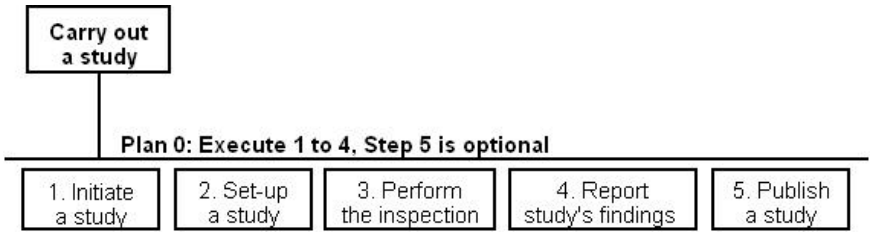

Figure 18: HTA diagram for task "Carry out a study"

The pre-mentioned process is described in detail in the User-experience evaluation framework, and will be explained only in brief in this section. In brief, an inspection team is assembled and specific goals for the study are established.

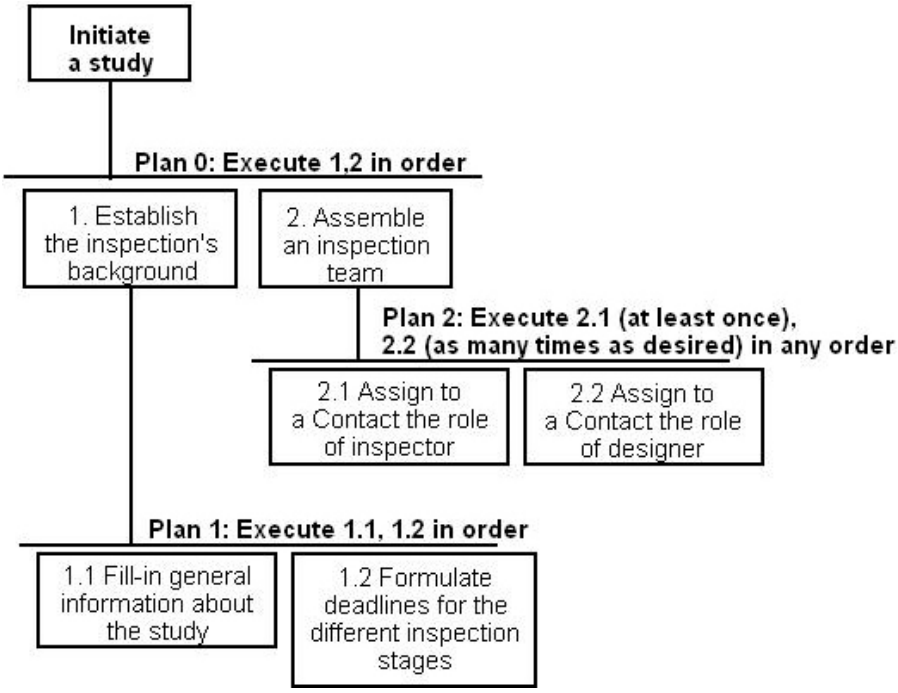

Figure 19: HTA diagram for sub-task "Initiate a study"

Afterwards, the inspection leader compiles all available information concerning the system to be inspected and records it in a specific form.

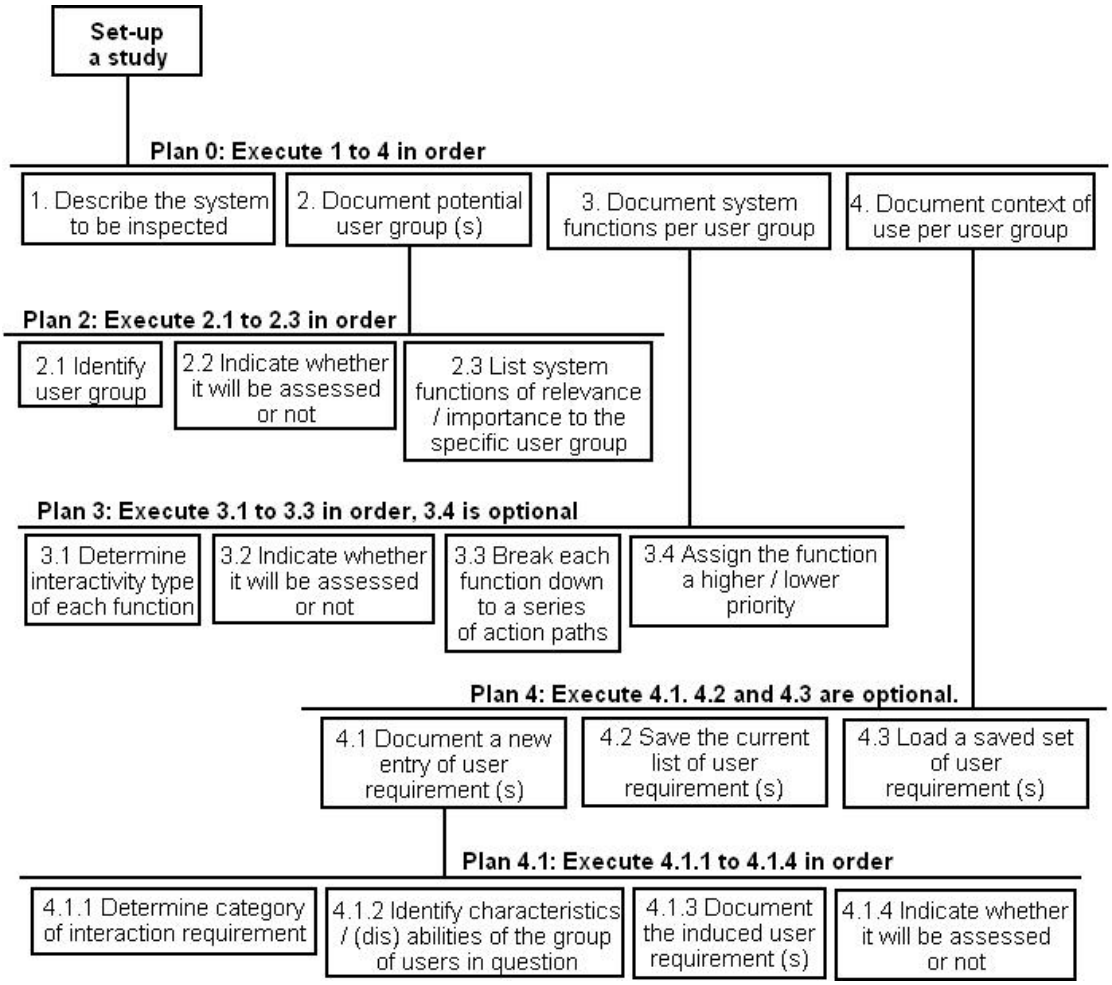

Figure 20: HTA diagram for sub-task "Set-up a study"

Based on this compiled information, the inspection team reviews the system with respect to certain quality characteristics / measurements (e.g., visibility, perceived usefulness & ease of use, etc.).

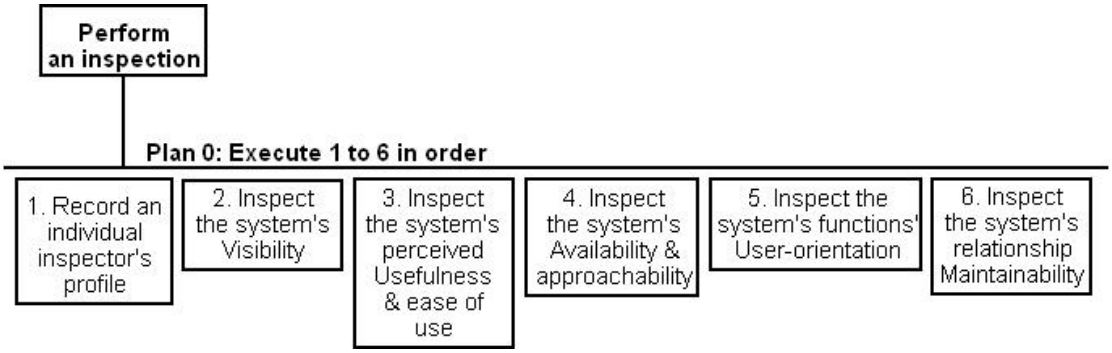

Figure 21: HTA diagram for sub-task "Perform an inspection" (part 1 of 2)

Following a step-by-step process, each inspector identifies and records good / bad practices of design and provides a severity rating for each of them.

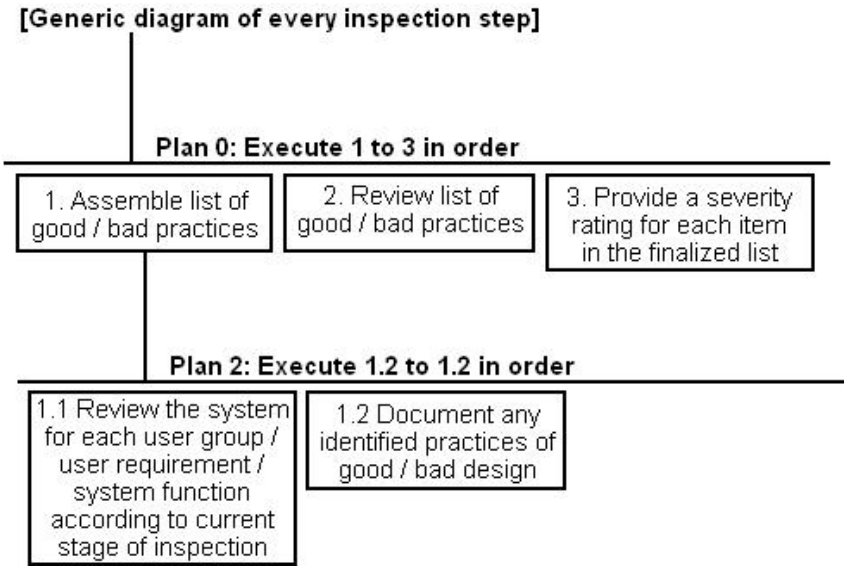

Figure 22: HTA diagram for sub-task "Perform an inspection" (part 2 of 2)

Finally, the inspection leader combines everything the inspection team has reported and creates collective forms reporting the findings of the study both in written, numerical and in graphical form.

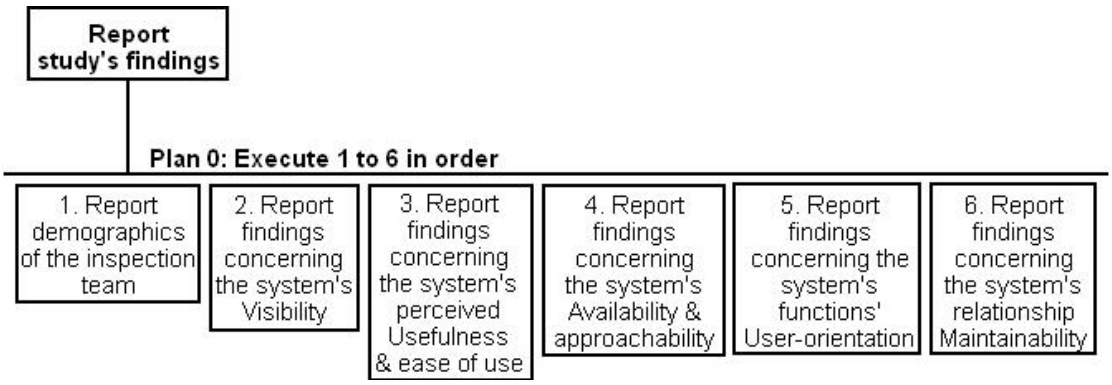

Figure 23: HTA diagram for sub-task "Report study's findings" (part 1 of 2)

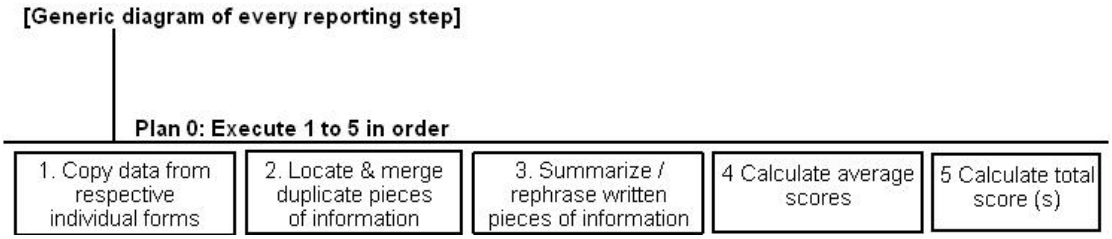

Figure 24: HTA diagram for sub-task "Report study's findings" (part 2 of 2)

Manage my archive

The user selects one of the inspections he / she has participated in, reviews all relevant forms to that inspection that he / she has filled in and retrieves the desired information.

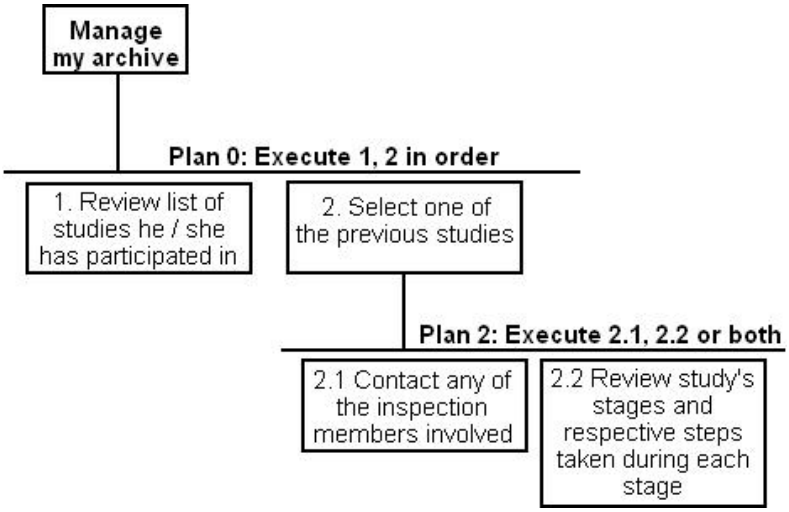

Figure 25: HTA diagram for task "Manage my archive"

Track study's progress

All members of the inspection team can retrieve information regarding progress of the study as a whole, as well as on their personal responsibilities deriving from the role they have been assigned within the study.

In order to provide estimations on personal progress, one has to take into consideration certain factors, such as role of the person in question, deadlines for the different stages of the study and the amount of work already carried out by the person in question. For example, if the person in question acts as an inspector and has failed to complete his/her inspection of the system within the specified time period, then he/she is labelled as "**late**".

From the inspection leader's point of view, estimating the study's progress is a fairly similar process. The inspection leader accesses a general overview of the study to establish what the current stage of the study is. Moreover, he/she may review the study's history log, keeping track of the steps each inspection team member has already completed and when they did so. Finally, the inspection leader may review each of the forms corresponding to these steps to ascertain the exact progress of each inspection team member.

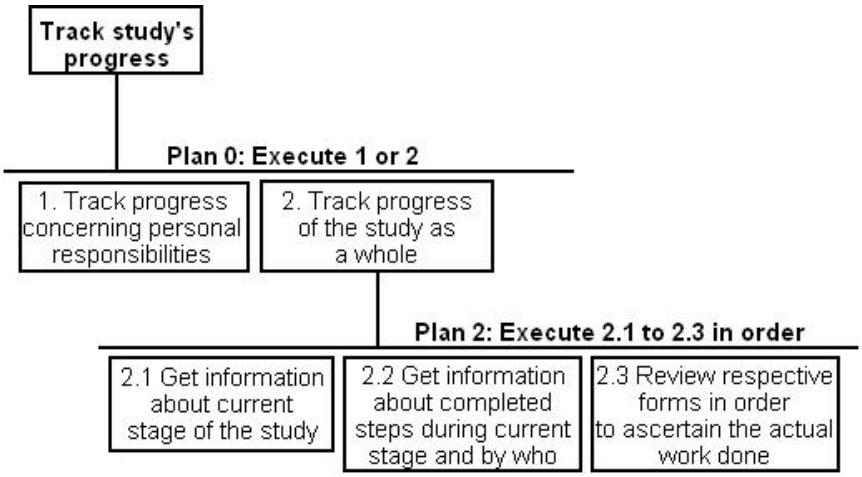

Figure 26: HTA diagram for task "Track study's progress"

Create result tables

Towards the end of the reporting phase of the study, the inspection leader creates graphical representations of the findings of the study. Using numeric data from the respective collective forms for each table, average values are calculated, matched with respective colour values (e.g., values between -1 and 1 are represented with the colour green) and displayed in the form of a table.

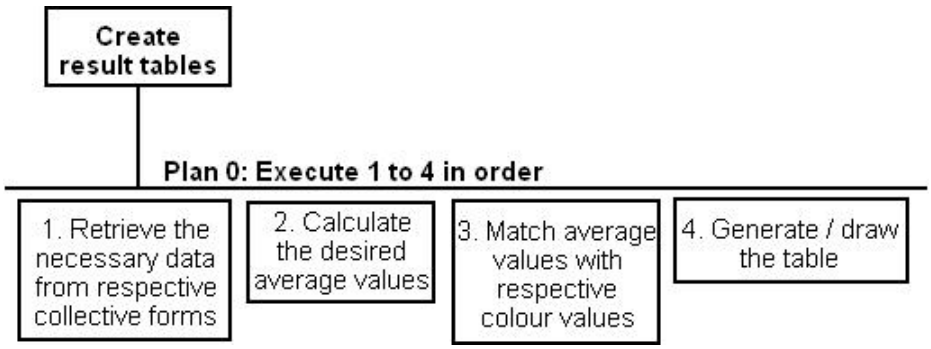

Figure 27: HTA diagram for task "Create result tables"

Manage messages

Each member of the ORIENT inspection tool is able to communicate with other members with the use of messages. By default, messages are sorted into two folders: the "Inbox" folder and the "Sent" folder.

Messages in the "Inbox" folder are originated from other accounts and are addressed to the owner of the account. The latter can read them, reply to them or delete them.

Messages in the "Sent" folder are originated from the owner of the account and are addressed to other accounts. The owner of the account can read any messages he/she has sent or delete them.

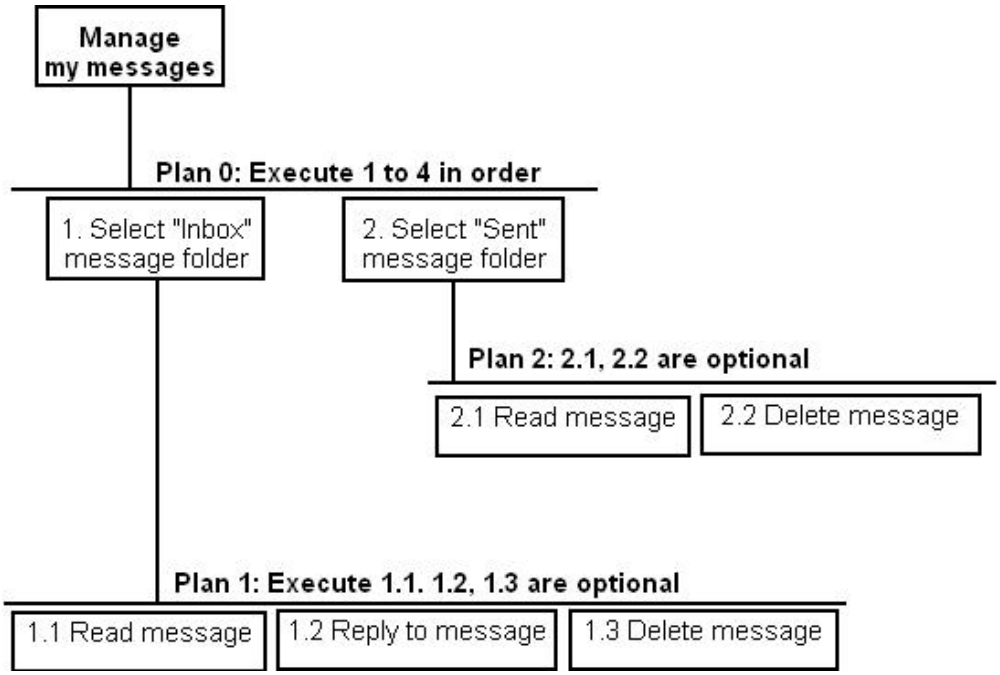

Figure 28: HTA diagram for task "Manage messages"

Compose new message

Every account owner within the ORIENT inspection tool can communicate with other members using messages.

In order to compose a message, a member of ORIENT has to first of all define the recipients' list of the message. These are divided into two categories: the direct recipients of the message (i.e., those that appear next to the "To:" field of a typical message) and the indirect recipients of the message or in other words the *carbon copy* recipients (i.e., those that appear next to the "Cc:" field of a typical message). Furthermore, to provide a level of privacy to members, so as not to receive messages from members they have not even ever heard of, the selection of recipients is limited to the list of contacts of the specific member / sender of the message.

Afterwards, the member needs only supply a subject for the message and the message's body (i.e., the actual written message), in order to complete and send the message.

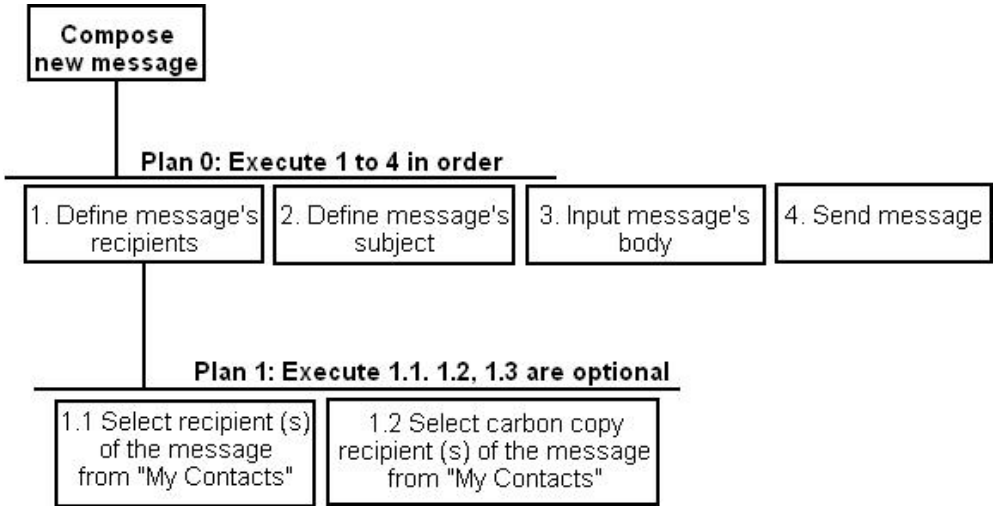

Figure 29: HTA diagram for task "Compose new message"

Manage my contacts

Every member of the ORIENT inspection tool may maintain and manage a contacts' list. This list operates more or less like a phone book. In other words, the member / owner of this list can add new entries to it, review information about or contact existing entries and remove entries from the list.

In order to add new contacts, an ORIENT member has to make use of the search function. The first step is to determine the search type, which can be search with no criteria and search with certain criteria. These may include limitations on the results' native language, degree of familiarity with ORIENT, expertise and level of fluency with the English written language. Having performed the search, the user can browse through the results and select one or more to which a message of invitation is dispatched.

The remaining functions that were mentioned at the beginning of this section are carried out in fairly the same way. The user reviews the list of his/her contacts, selects one or more (depending on the specific function) and performs the desired action (e.g., review a contact's profile, delete contact, etc.).

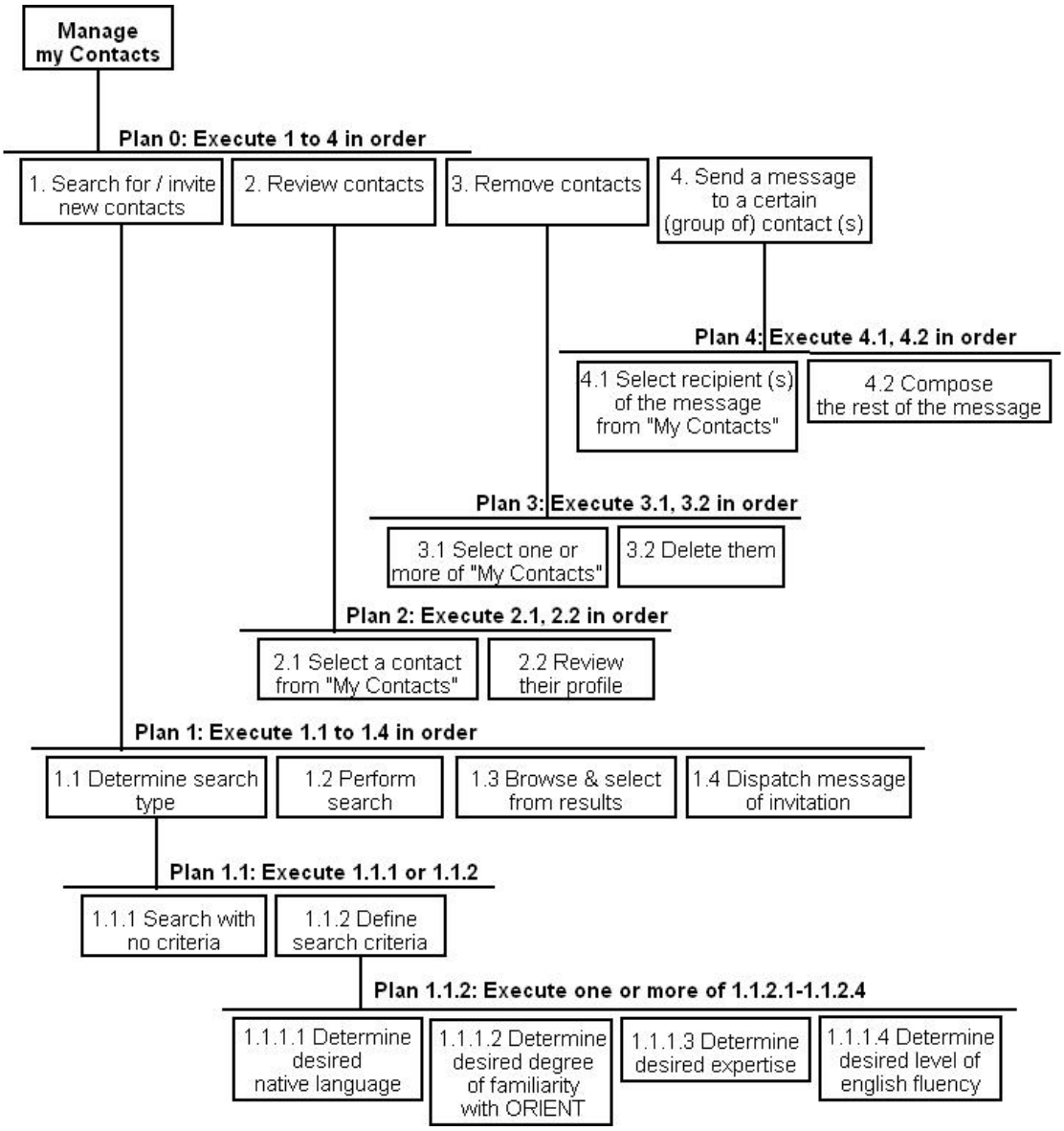

Figure 30: HTA diagram for task "Manage my contacts"

4.3.2 User interface design – General decisions

According to a recent study of screen resolutions user by internet surfers [4] which was conducted by Ascad Networks, 1024x768 is the new standard, as 45,2% of the 425 visitors tracked were using monitors with a screen resolution of 1024x768, whereas only 6% had screen resolutions of 800x600. As such, mock-ups' dimensions were set to accommodate a screen resolution of 1024x768.

All colours present in the user interface mock-ups belong to the web safe palette [44]. The web safe palette is a palette consisting of 256 colours that are displayed in the same way by all browsers. More over, in order to ensure sufficient contrast even if the system is displayed in a black and white screen, the user interface mock-ups were assessed with the use of the Vischeck simulation tool[1].

The placement of different sections of the user interface in the available visible area of the web application was decided taking into consideration conventions and standards common throughout the web. Every page, usually, is divided into four main zones: the header, the navigation, the main page and the footer.

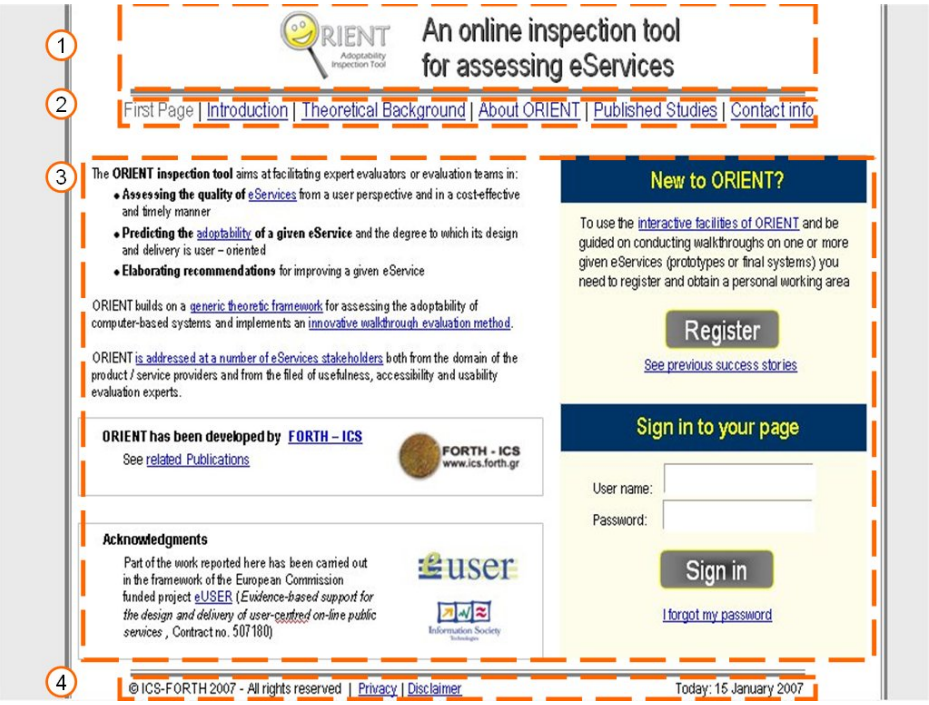

Figure 31: The placement of a page's elements into four zones.

The header zone is located at the top of the screen and is stretched across its entire length. It contains elements such as the web site's logo, the user's name, etc.

[1] Available at: http://www.vischeck.com/vischeck/.

Options that belong in the navigation zone are presented in a similar way to the header zone or in a vertical angle, according to the available options. The main page section contains the actual content of the page, as well as its title. Finally, the footer section is displayed in a similar way to the header section and contains information such as the copyright, privacy and disclaimer statements, as well as the current date.

4.3.3 User interface mock-ups

Incorporating all of the information collected during the user interface design stage, a series of graphical mock-ups by means of *Microsoft Office PowerPoint 2003* were created.

Public view of the ORIENT inspection tool

The appearance of the GUI, as well as the functionality offered to the user, depends on whether the user is a registered member of the inspection tool or a simple visitor of the webpage. The term *public view* refers to the view of the inspection tool that a simple visitor is presented with when visiting the webpage of the ORIENT inspection tool.

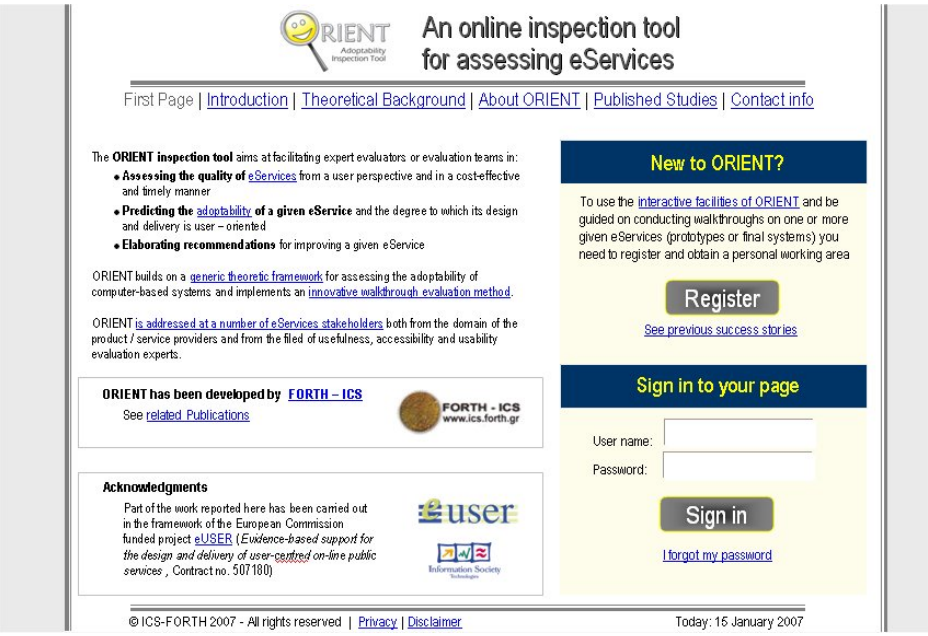

Figure 32: Mock-up of the homepage of the ORIENT inspection tool

As presented in figure 4.26, the welcome screen offers sufficient information to help the user / visitor decide whether or not the ORIENT service interests them. General information about the tool is briefly presented on the left part of the screen, whereas the **register** and **sign in** panels are located on the right part of the screen.

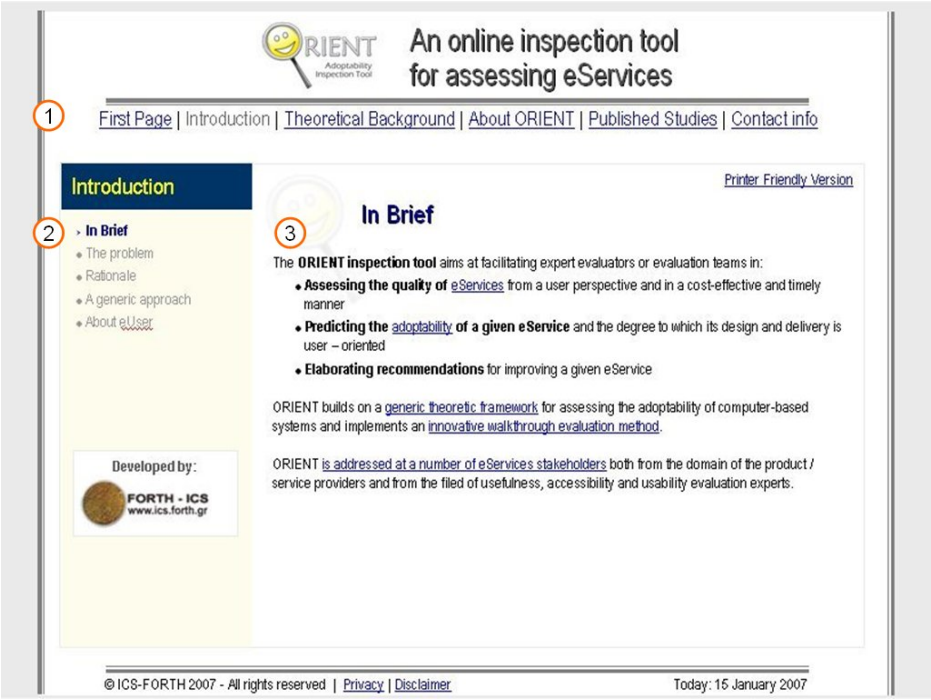

Figure 33: Mock-up of the layout used for the public sections of the ORIENT inspection tool.

The remaining sections which are publicly available to anyone are:

- Introduction
- Theoretical background
- About ORIENT
- Published studies
- Contact info.

Whenever these sections include sub-sections, the presented information will be organised as shown in figure 4.27. Main menu options are available across the top of the screen **(1)**. Sub-menu options, for navigation within the current section, are located at the left column of the screen **(2)** and finally the main content area, where the actual information is presented, is situated at the right column of the screen **(3)**.

If however the section contains such a small amount of information that needs not being divided into sub-sections, the sub-menu options are excluded from the layout and the main content area takes up the extra space.

Homepage of the ORIENT inspection tool

Once users have signed in, they are transferred to the homepage of the ORIENT inspection tool. As shown in figure 4.28, the layout of elements on the screen and the available functionality differs significantly from the ones in the public view of the inspection tool.

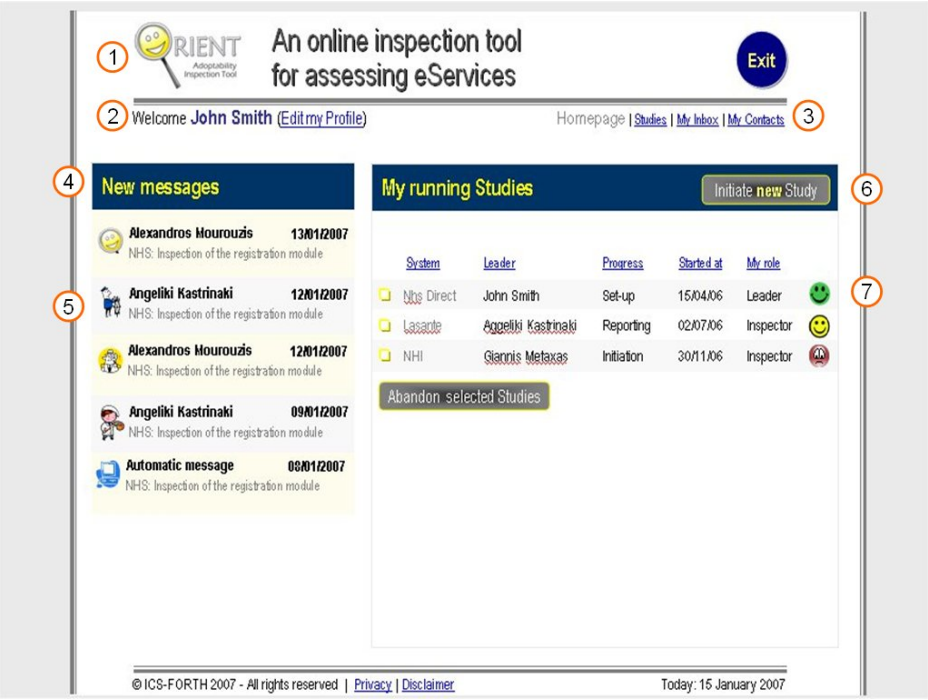

Figure 34: Mock-up of the homepage of the ORIENT inspection tool.

A logged in user of the ORIENT inspection tool has access to all of its functionality and therefore the way information is presented on the screen should alter to incorporate the new available options. The header area is slightly transformed to accommodate the "Exit" button **(1)**. The plain navigation bar, which was situated right under the header section, is divided into two sub-sections. The first one concerns information about the logged in user and his/her profile **(2)** and the second one is the main menu **(3)**. Taking into consideration that logged in users are expected to be more interested in actually using the inspection tool than reading general information about it, the options available in the main menu are different than those in the public view of the ORIENT inspection tool. All sections present in the main menu of the public view are omitted and replaced by new ones. These are:

- Homepage
- Studies
- My Inbox
- My Contacts.

Among these options, the ones that are most likely to interest the user and therefore should appear on the homepage (the first page the user sees when logging in to the inspection tool) are notifications of new messages the user has received and all the studies he/she is currently participating in.

New messages are presented on the left of the screen **(4)**, listing in a column all new messages with the required information **(5)** (sender of the message, relationship between the user and the sender, date the message was sent and topic of the message). Current studies,

which are considered the most important part for every ORIENT user, take up the center and right part of the screen **(6)**, briefly presenting studies as well as any available functionality **(7)** (according to the logged in user and their rights / privileges). The "Studies" section follows a similar layout as the one presented in figure 5.28 and presents studies divided into three groups: running, completed / archived and published.

Message folders

ORIENT members may communicate with other members with messages. These are sorted into two default folders, the "inbox" folder and the "sent" messages folder, which are presented in a similar way (figure 4.29).

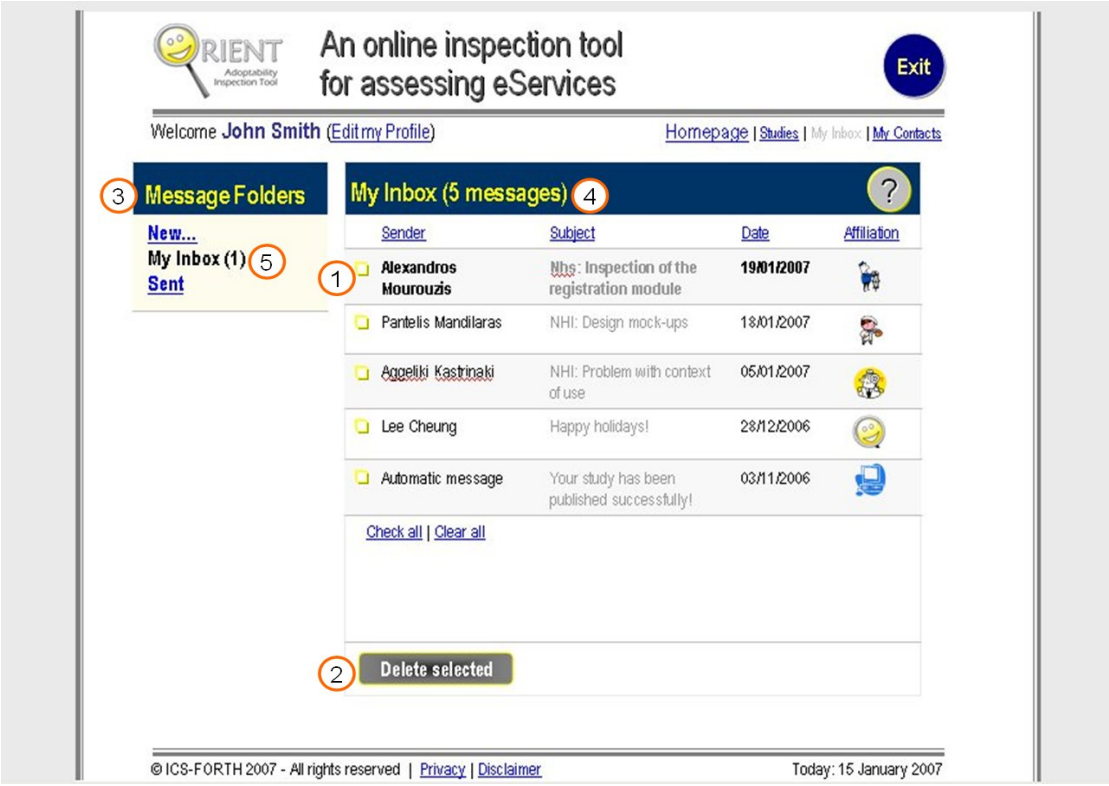

Figure 35: Mock-up of "My Inbox" folder for a given user of ORIENT.

All messages in the folder are presented in a collective table **(1)**, with information on the sender, the subject and the date the message was sent, as well as the affiliation of the sender and the current user (i.e., an indication of the relationship between the current user and the sender of the message). Users may select to read any message by selecting it or may choose to delete **(2)** one or more of them.

A sub-menu **(3)** with options related to messages is presented on the left of the screen. With its aid, the user may navigate between the two messages' folder and compose new messages.

Finally, the numeric indication next to the main heading **(4)** refers to the total number of messages in a user's inbox, whereas the indication next to the "inbox" hyperlink in the sub-menu **(5)** refers to the number of new / unread messages in a user's inbox.

Contacts' organization

ORIENT members may wish to quickly contact other members. In order to address this need, the mechanism of "contacts" is used, which is quite similar to an entry in a phonebook. Each user may look up their list of contacts, as well as ORIENT members they have invited to join their contacts, but have not yet responded.

In order to add new contacts to one's list, a user needs to use the "Find new Contacts" function (figure 4.30), which gives the user the ability to perform a search (with some or no specified search criteria) among all registered members of the ORIENT inspection tool. Users define the desired limitations **(1)** (e.g., native language, familiarity with using ORIENT, etc.) that members in the result set should satisfy and perform the search **(2)**.

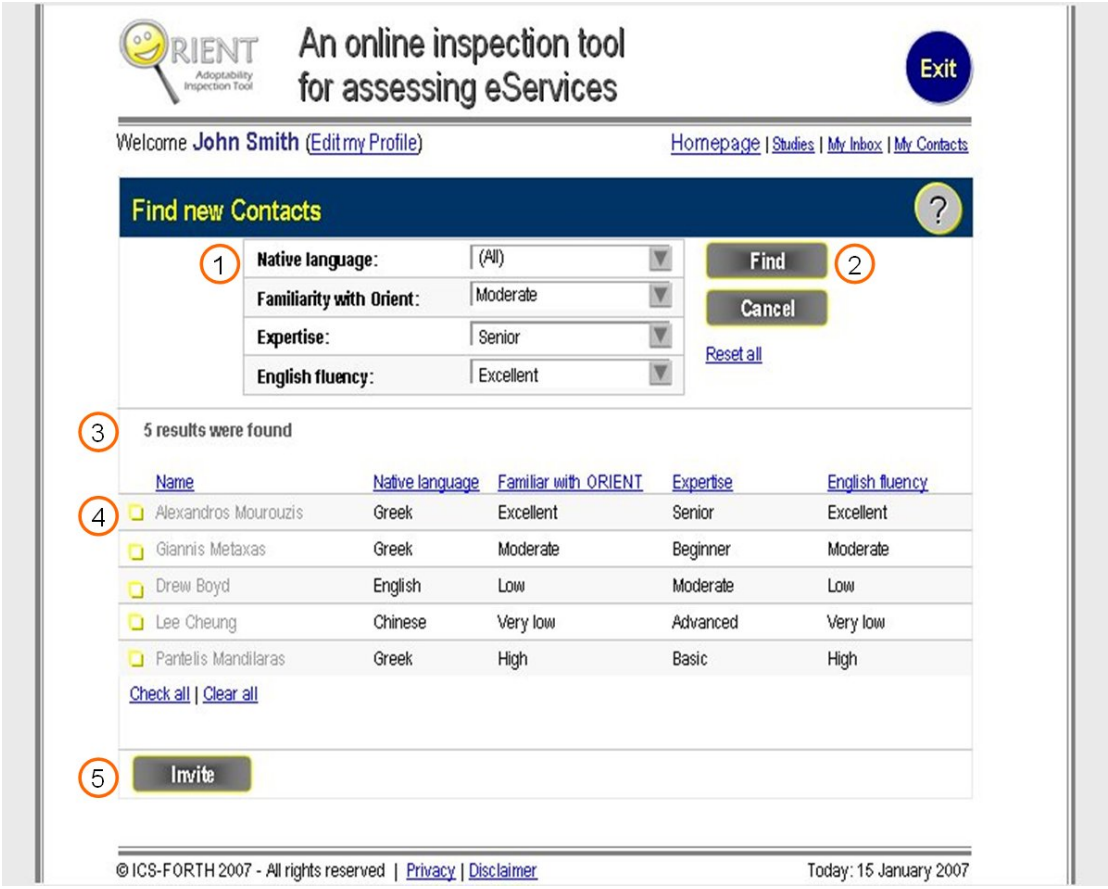

Figure 36: Mock-up of the search for new contacts function

If the search yields any results **(3)** (i.e., if there is at least one member that satisfies the search criteria), they are presented in a table **(4)** with their individual values for all the

available search criteria. Users may select one of more ORIENT members from the list and extend an invitation to them **(5)** or alter their search criteria **(1)** and perform a new search **(2)**.

Organization of a study

As mentioned in previous chapters, a study in ORIENT is broken down into several stages. In order to provide a quick way of accessing all of these different sections, the notion of a "workspace" was created. A study's workspace is like a virtual "area" in which all "documents" relevant to a study are kept. Though it is related to studies, it does not constitute a sub-category of them. Workspaces (figure 4.31) are generated for each study automatically upon their initiation, and all members of the inspection team have access rights to them.

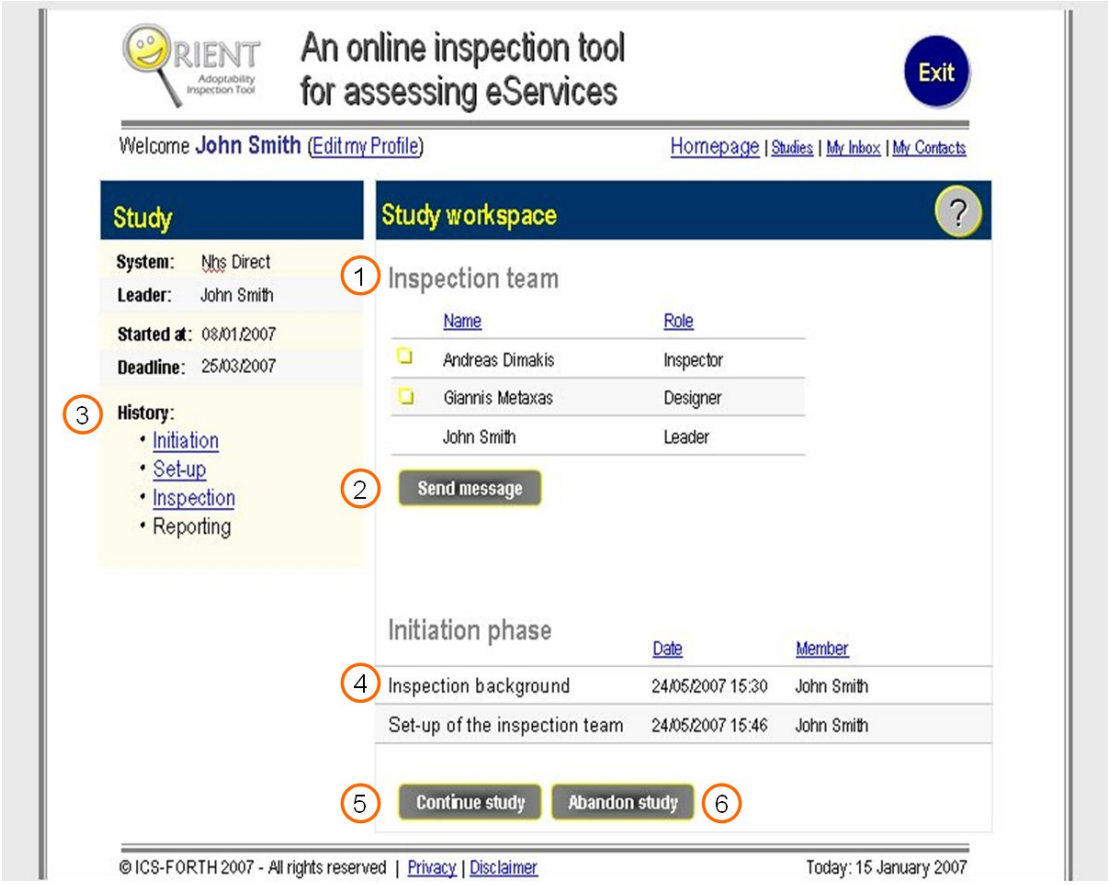

Figure 37: Mock-up of the layout of information in a Study's workspace.

Information about the inspection team's composition is presented in a workspace **(1)**. Members of the inspection team may also contact **(2)** other members from within the workspace. All members may receive updates about the current stage of the study **(3)**, as well as the level of completion **(4)** of the current stage (i.e., the "documents" that have already been created by members of the team). All members can be transferred to the next step **(5)**

they should complete, according to their role in the study. Finally, the inspection leader may select to abandon **(6)** a study from within its workspace.

Set-up phase of a Study

The first stage of a study is the "Set-up" phase. The proposed layout for every page within this stage is presented in figure 4.32.

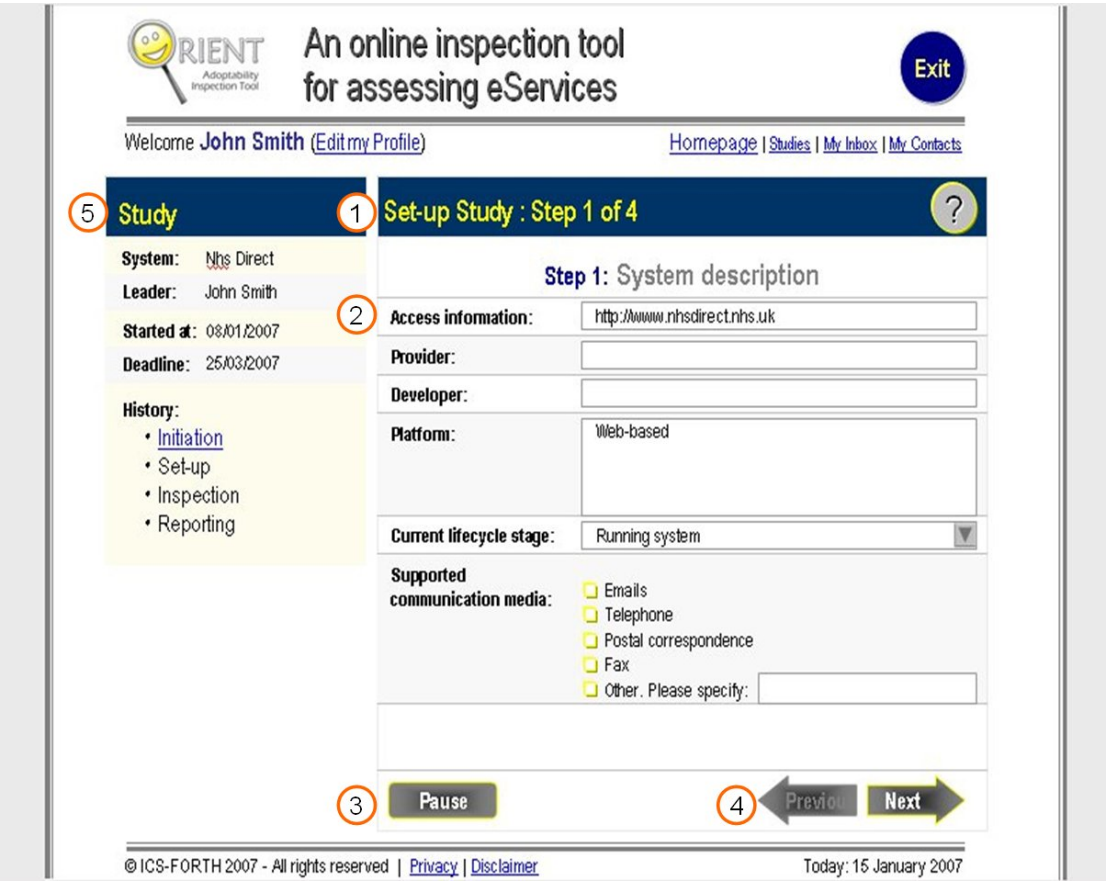

Figure 38: Mock-up of the layout for the "Set-up" phase of a Study.

Each page's heading informs the user about the current sub-step, as well as the number of remaining sub-steps to complete the current stage **(1)**. In the main area of the window, data forms suitably constructed to meet the needs of each sub-step are presented **(2)**. The user inputs the required data for every sub-step into the aforementioned forms.

Navigating between sub-steps and stages is accomplished sequentially by the previous and next arrows **(4)** located at the bottom right part of the main area of the window. Furthermore, the user may quickly "jump" forward or backward to a stage using the respective links available in the "History" section **(5)** of the "Study" panel, located at the left side of the window. Important information about the study, such as the inspection leader and the time deadlines, is always visible in the "study" panel **(5)**. Whenever the user has to carry

out non-trivial tasks, a help button is available in the header area **(1)**, providing the user with case sensitive help, as well as the ability to browse all help contents. Finally, the user may choose to "pause" the study **(3)** at any given study.

Inspection phase of a Study

Every sub-step of the inspection phase comprises of two steps, the documentation of good / bad practices identified in the system under assessment and the rating of the aforementioned practices.

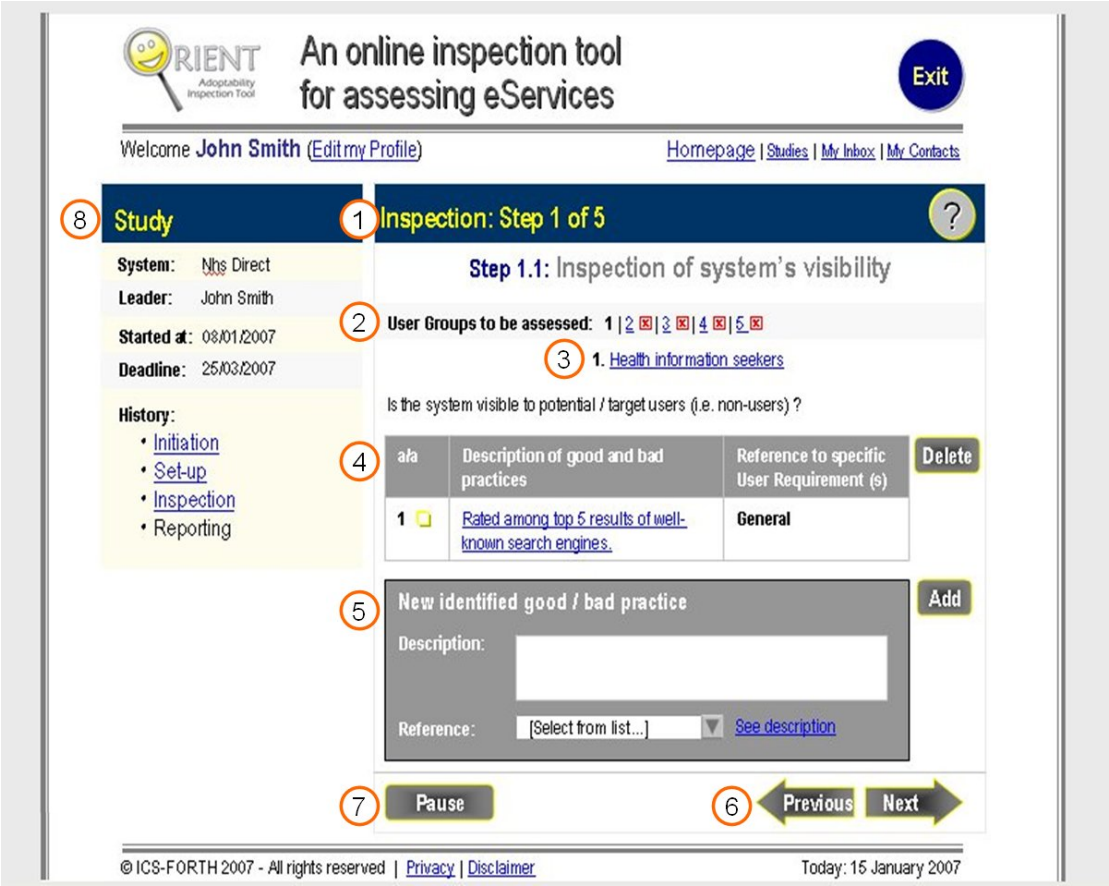

Figure 39: Mock-up of the layout for pages, used to document practices during the inspection phase.

The layout for the documentation of good / bad practices is displayed in figure 4.33. Each page's heading informs the user about the current sub-step, as well as the number of remaining sub-steps to complete the current stage **(1)**. Sub-steps of the inspection phase are iterated for all defined user groups (and functions). The user can navigate between user groups with the respective sub-menu located at the top of the main area of the window **(2)**. User groups (or functions) that the user has not yet assessed are marked with a special icon, in the shape of a red *x* in a square box.

As mentioned earlier, the user cycles through all user groups (and functions) to carry out the inspection. The current user group's (and function's) title is available as a hyperlink to the respective section of the study's documents that provide information about it **(3)**. All good / bad practices that the user has already documented are presented in a table **(4)** with all necessary information, which the user may edit or delete at any given time. The panel for documenting a new identified good / bad practice **(5)** is available right under the previous table. A hyperlink offers additional information about references to the context of use.

Navigating between sub-steps and stages is accomplished sequentially by the previous and next arrows **(6)** located at the bottom right part of the main area of the window. Furthermore, the user may quickly "jump" forward or backward to a stage using the respective links available in the "History" section **(8)** of the "Study" panel, located at the left side of the window. Important information about the study, such as the inspection leader and the time deadlines, is always visible in the "study" panel **(8)**. Help is always available by means of a help button in the header area **(1)**, providing the user with case sensitive help, as well as the ability to browse all help contents. Finally, the user may choose to "pause" the study **(7)** at any given study.

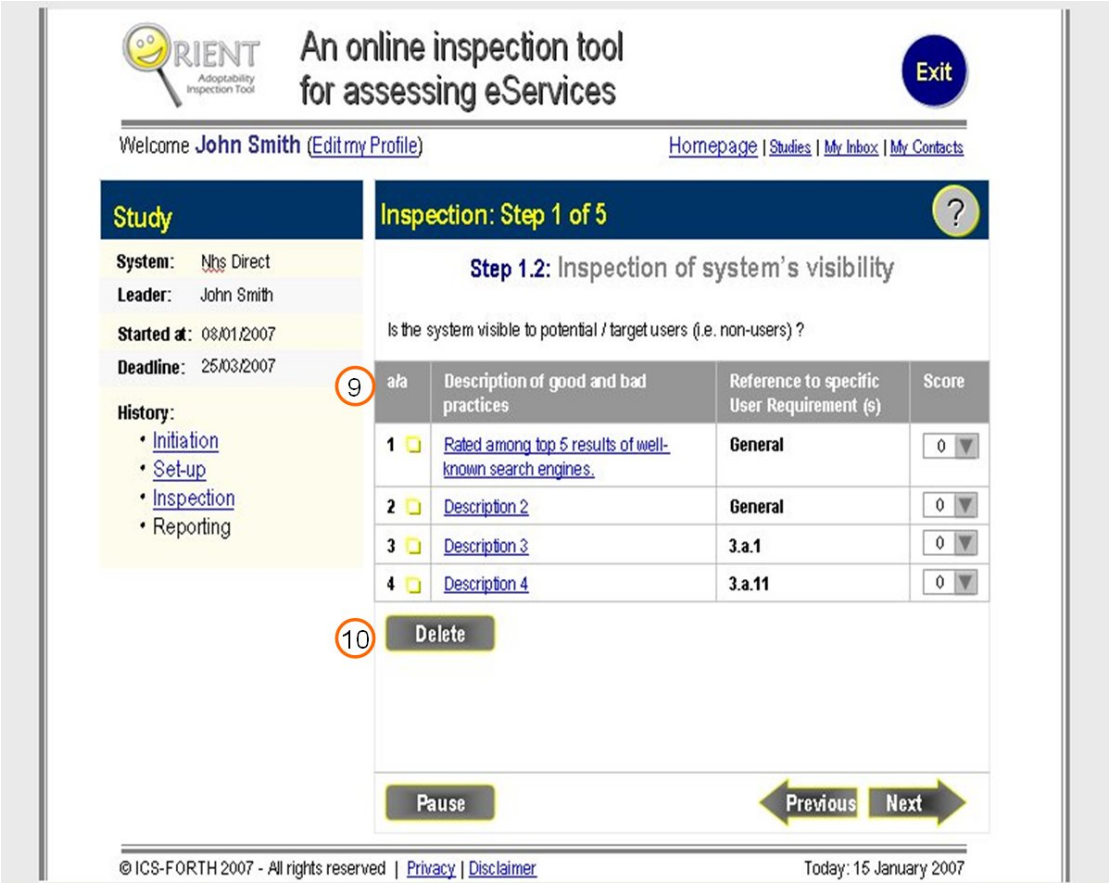

Figure 40: Mock-up of the layout for pages, used to score practices during the inspection phase.

The layout for the rating of good / bad practices is displayed in figure 4.34. All documented by the inspector practices for the current step are presented in a table **(9)** and the user may provide severity ratings for each of them or even delete any practices he / she finds redundant **(10)**.

Reporting phase of a Study

The final stage of a study is the reporting phase. All identified good / bad practices by individual inspectors are aggregated by the system and presented to the inspection leader for possible editing.

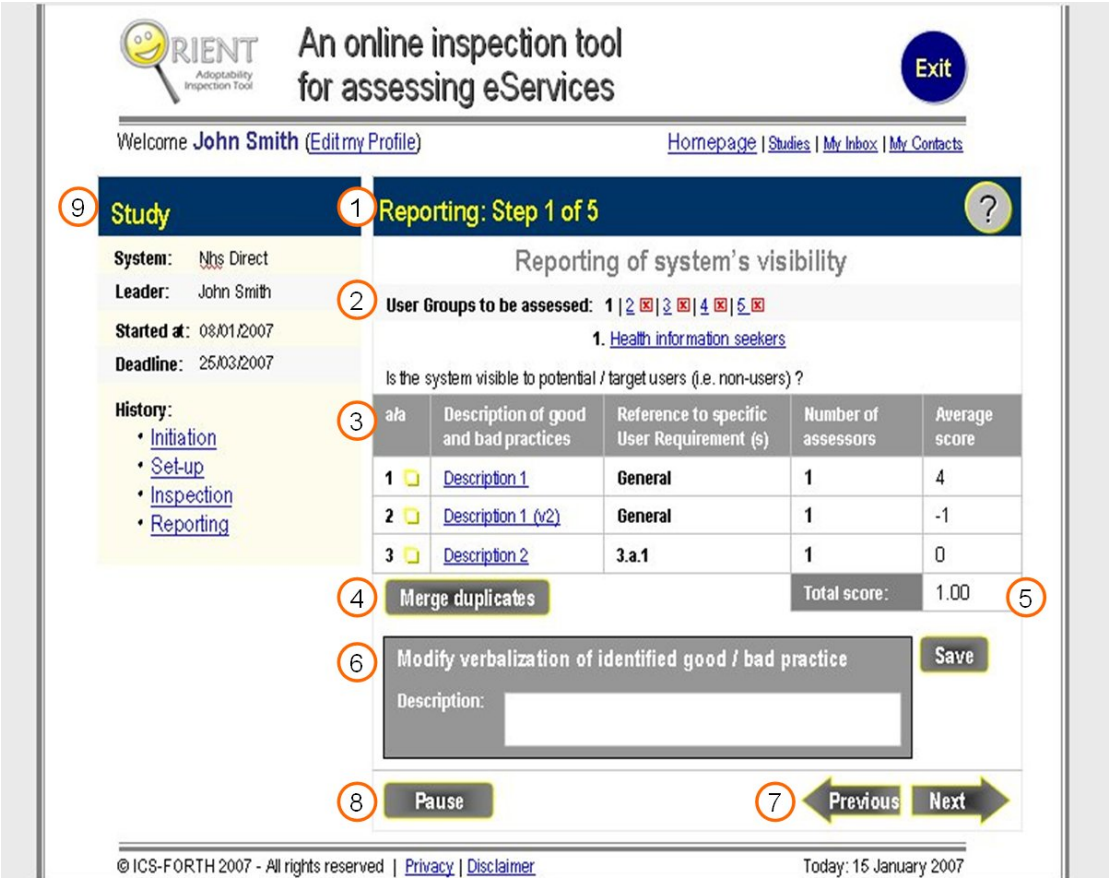

Figure 41: Mock-up the layout for pages during the reporting phase.

Each page's heading informs the user about the current sub-step, as well as the number of remaining sub-steps to complete the current stage **(1)**. The reporting process is iterated for all defined user groups (and functions). The user can navigate between them with the respective hyperlinks located at the top of the main area of the window **(2)**. User groups (or functions) that the user has not yet assessed are marked with a special icon, in the shape of a red *x* in a square box.

All good / bad practices documented by any inspector are presented in a table **(3)** with all necessary information. The inspection leader may review these practices and merge some of them in case they refer to the same good / bad design feature **(4)**. Total score for the current feature is automatically calculated by the ORIENT inspection tool **(5)**. The inspection leader may wish to modify the verbalization of a practice **(6)**.

Navigating between sub-steps and stages is accomplished sequentially by the previous and next arrows **(7)** located at the bottom right part of the main area of the window. Furthermore, the user may quickly access different stages using the respective links available in the "History" section **(9)** of the "Study" panel, located at the left side of the window. Important information about the study, such as the inspection leader and the time deadlines, is always visible in the "study" panel **(9)**. Help is always available by means of a help button in the header area **(1)**, providing the user with case sensitive help, as well as the ability to browse all help contents. Finally, the user may choose to "pause" the study **(8)** at any given study.

5 Implementation

5.1 Database implementation

5.1.1 Tools and materials

The database supporting the needs of the ORIENT inspection tool was implemented in MySQL (v. 5.0.27)[2] using SQLyog Community Edition (v. 6.11)[3], a popular MySQL GUI. MySQL runs over an Apache server (v. 2.2.6)[4].

5.1.2 Implementation of database's schema

The definitions of entities and relationships mentioned in chapter 4.2.2 are expressed as tables in order to fully describe the database model. Every table portrays an entity and presents its attributes and the type of value these attributes hold. Furthermore, every attribute is stated as an identifier (key) or as a descriptor (descriptive attribute).

Weak entities, such as forms, are identified by foreign keys and local partial identifiers. For example, an instance of a form 4a is identified by the id of the inspector who filled it in (foreign key from the entity **Inspection team member**), the id of the system which undergoes the inspection process (foreign key from the entity **Inspection**), the id of the user group it relates to (foreign key from the entity **Form 2 – Potential user groups**) and the description of the good / bad practice (local partial identifier of the entity **Form 4a**).

Relationships, similarly to entities, are represented as tables. Tables for relationships follow the same pattern as the tables for entities. However, taking into account the relationships' cardinalities, certain tables are not necessary. Relationships with a 1-to-N or 1-to-many cardinality are not displayed as separate tables. They are represented by inserting the identifier of one of the participating entities as a foreign key into the table of the other participating entity. Therefore, **EstablishInspectionBackgroundIn**, **FillIn** and **HasDocumentedResultsIn** relationships are already represented accurately by the previous tables for entities.

A full documentation of the tables that constitute the database supporting ORIENT is available in Appendix B.

---

[2] Distributed with WAMP5 at: http://www.wampserver.com/en/
[3] Available at: http://www.webyog.com/en/downloads.php
[4] Distributed with WAMP5 at: http://www.wampserver.com/en/

Studies

The main functionality of ORIENT is to aid its members inspect systems. The function of carrying out a study comprises three entities / relationships: the members of ORIENT, the study and the actual participation of the former in the study. Any given study can be identified with the aid of table "Inspection" (table B.1).

ORIENT members are represented in the database of the inspection tool by means of table "Inspection team member" (table B.2), which contains all the necessary information to identify a member. The same information acts as the member's profile.

Finally, the action of a member participating in a study is represented in the table "Participate" (table B.3). This table defines the specific characteristics of this relationship and also documents the composition of the inspection team.

Certain pieces of information documented by form 1a (e.g., *familiarity with the system in question*, *familiarity with languages supported by the system*, etc.) relate more closely to a member's actual involvement in a specific inspection. Thus, this information is best portrayed as attributes of the relationship **ParticipateIn**. Furthermore, the relationship needs additional attributes to specify the nature of participation of an inspection team member (i.e., as an individual inspector, as an inspection leader or as both). Finally, relationship **ParticipateIn** is the source of information for the overall presentation of the inspection team and its demographics, a process that takes places at the very beginning of the reporting phase.

Messages

In order to support the exchange and storing of messages between members of ORIENT, the following tables are necessary. Table "Message" (table B.4) holds information about the actual message and the sender of it. Table "Recipient" (table B.5) identifies the recipient(s) of a specific message.

The way these tables are used to provide ORIENT members with the feature of messages in an efficient and effective way is described in full detail in section 5.3.2.

Contacts

All connections between ORIENT members that formulate the concept of *Contact* are implemented with the aid of two relationships / tables. Each row in table "Contact" (table B.6) defines a couple of ORIENT members, where the first one acts as the "owner" of the relationship and the second one as the "recipient". Whenever, a member accepts another member's invitation two new rows are created in table B.6 (one for each of the participants in this contact-relationship), as a contact-relationship is reciprocative.

Table “Invited contact” (table B.7) states the couples of ORIENT members, where one member has extended an invitation to the other member, but the latter has yet to reply to that invitation.

Set-up phase forms

The set-up phase of the inspection, as described in 3.3, includes the description of the system under assessment, the declaration of user group(s) and system functions of importance for each user group respectively, and finally the definition of the context of use for each user group.

The system’s description is documented with the aid of table “System description”, described in table B.8. Potential user groups are described by means of the table “Potential user group(s)” (table B.9).

For each user group, several system functions (of importance to the specific user group) are declared and analyzed in table “System functions” (table B.10). Furthermore, each function is broken down into a series of actions, which themselves are documented as well in table “System actions” (table B.11).

The final sub-step of the set-up phase is the documentation of the context of use (table B.12) and the induced requirements (table B.13) for each user characteristic per user group.

In order to maintain context-of-use records that the user has chosen to “save” for later use, a set of new tables are created (tables B.14, B.15). These new tables will retain the information documented in a saved context-of-use record even if the study in which they originally appeared is deleted. Thus, a user may reuse a context-of-use record they have defined in the past (as long as it is suitable for the needs of the current study) and save considerable time and effort in the setting-up of a new study. In this context, reusability of information is achieved.

Inspection phase forms

The inspection of a system includes the assessment of five system characteristics: visibility, usefulness, availability, quality of interaction and relationship maintainability. Findings by inspectors concerning the visibility of the system under assessment are documented by means of table “Visibility” (table B.16).

Likewise, any good / bad practices (relating to the system’s perceived usefulness and ease of use) identified by inspectors are described with the aid of table “Perceived usefulness & ease of use” (table B.17).

In 4.2.2, information about the system’s availability and approachability was divided into three weak entities, according to the level of experience of the users to whom the good /

bad practices referred to. Table “Availability & approachability” (table B.18) merges these three entities into one with the introduction of a new attribute, “Experience”.

The inspection of a function’s user-experience contains 11 sub-steps in total, assessing the function’s visibility, usefulness, availability, quality of interaction and relationship maintainability for any of the three levels of users’ experience (novice, moderate, expert). These sub-steps are unified into one table (table B.19). Attribute experience receives values from 1 to 11. Each value reflects practices documented at the respective sub-step of the inspection of a function’s user experience. The user’s experience is extracted from the same data by formulating three groups of values (i.e. values 1 to 5 correspond to first-time and novice users, 6 – 8 to moderate users and 9 – 11 to expert users).

The three entities described in 4.2.2 concerning the relationship maintainability of the system under assessment are merged into one table (table B.20) in the same way as described for table B.18.

Reporting phase forms

The final stage of a study is the reporting phase, where findings by individual inspectors are aggregated in collective forms. Collective forms concerning the visibility and the usefulness of the system under assessment are documented by means of the tables “Visibility” and “Perceived usefulness & ease of use” respectively (tables B.21 and B.22).

Collective data for the system’s availability and relationship maintainability, as well as functions’ user-experience, are documented in a similar way using tables B.23, B.24 and B.25 respectively. The only difference between them and the two previous collective tables is the introduction of the attribute “Experience” to distinguish the group of users the practice refers to.

Forms 7b, 9 and 10, described in 4.2.2, need not be implemented as individual tables as they contain data that can easily be extracted from the previous tables.

5.2 Interface implementation

5.2.1 Tools and materials

The interface of the ORIENT inspection tool was implemented in PHP: Hypertext Preprocessor (PHP v. 5.2.0) [5]. PHP code is executed, upon request, and Hyper Text Markup Language (HTML v. DTD XHTML 1.0 Transitional[6]) web-pages are delivered to clients over an Apache server (v. 2.2.6) [7].

Web-pages layout is produced with use of structural HTML (i.e., use of tables for layout purposes). Presentation effects are accomplished by Cascading StyleSheets (CSS v.2[8]). Any necessary client-side scripting is implemented in Javascript[9].

5.3 Functionality

In the following sections, the code of the ORIENT inspection tool will be briefly presented, covering all important and interesting parts.

5.3.1 Libraries

Library "user.php"

This library contains functions related to the user of the ORIENT inspection tool. This includes actions of a generic nature that the user performs (e.g., sign-in, register, etc.), as well as actions that relate to the user's personal profile.

Table 5.4: Function register( ), included in library user.php

Function **register( )**		
Arguments	**Returned values**	**Description**
• **$firstname (String)** • **$lastname (String)** • **$username (String)** • **$password (String)** • **$email (String)** • **$question (String)**	None	Registers a new ORIENT user with the specified attributes.

[5] Distributed with WAMP5 at: http://www.wampserver.com/en/
[6] Specification available at: http://www.w3.org/TR/xhtml1/
[7] Distributed with WAMP5 at: http://www.wampserver.com/en/
[8] Specification available at: http://www.w3.org/TR/CSS21/
[9] Specification available at: http://www.planetpdf.com/codecuts/pdfs/tutorial/jsspec.pdf

• **$answer (String)**		

Table 5.5: Function update( ), included in library user.php

Function **update( )**		
Arguments	**Returned values**	**Description**
• **$what (String)** • **$value (String)** • **$username (String)**	None	Replaces the existing value of field *$what* of user-profile belonging to *$username* with value *$value*.

Table 5.6: Function show_profile( ), included in library user.php

Function **show_profile()**		
Arguments	**Returned values**	**Description**
• **$user (String)**	None	Retrieves and displays the user-profile belonging to *$user*.

Table 5.7: Function login( ), included in library user.php

Function **login( )**		
Arguments	**Returned values**	**Description**
• **$username (String)** • **$password (String)**	Integer value.	Logs user with *$username* and *$password* into ORIENT. If the process is successful, it returns 0. If the password is wrong, or the user doesn't exist, it returns another error value (1 or 2 respectively).

Table 5.8: Function new_password( ), included in library user.php

Function **new_password( )**		
Arguments	**Returned values**	**Description**
• **$usernane (String)**	String value.	Generates a new password for user *$username*. Returned value is the new password.

Table 5.9: Function email_password( ), included in library user.php

Function **email_password( )**		
Arguments	**Returned values**	**Description**
• **$usernane (String)** • **$password (String)**	Boolean value.	E-mails the *$password* to the e-mail address specified in *$username*'s profile. Returned value is a Boolean value (successful or not).

Table 5.10: Function get_Name( ), included in library user.php

Function **get_Name( )**		
Arguments	**Returned values**	**Description**
• **$username (String)**	String value.	Retrieves the full name of *$username*. Returned value is the full name.

Table 5.11: Function edit_Profile( ), included in library user.php

Function **edit_Profile( )**		
Arguments	**Returned values**	**Description**
• **$username (String)**	None	Retrieves and displays the user-profile belonging to *$username* for editing.

Library "messages.php"

This library's functions address issues related with the creation, management and deletion of messages.

Table 5.12: Function count_new_messages ( ), included in library messages.php

Function **count_new_messages( )**		
Arguments	**Returned values**	**Description**
• **$user1 (String)**	Integer value.	Accesses *$user1*'s inbox and counts new messages. Returned value is the total number of new messages found.

Table 5.13: Function count_messages( ), included in library messages.php

Function **count_messages( )**		
Arguments	**Returned values**	**Description**
• **$user1 (String)**	Integer value.	Accesses *$user1*'s inbox and counts all messages found in the folder. Returned value is the total number of messages found.

Table 5.14: Function count_sent_messages( ), included in library messages.php

Function **count_sent_messages( )**		
Arguments	**Returned values**	**Description**
• **$user1 (String)**	Integer value.	Accesses *$user1*'s "Sent" folder and counts all messages found in the folder. Returned value is the total number of messages found.

Table 5.15: Function show_inbox_messages( ), included in library messages.php

Function **show_inbox_messages( )**		
Arguments	**Returned values**	**Description**
• **$user1 (String)** • **$order (Integer)**	None	Retrieves and displays messages found in *$user1*'s inbox folder in the specified *$order*.

Table 5.16: Function show_sent_messages( ), included in library messages.php

Function **show_sent_messages( )**		
Arguments	**Returned values**	**Description**
• **$user1 (String)** • **$order (Integer)**	None	Retrieves and displays messages found in *$user1*'s "Sent" folder in the specified *$order*.

Table 5.17: Function show_new_messages( ), included in library messages.php

Function **show_new_messages( )**		
Arguments	**Returned values**	**Description**
• **$user1 (String)**	None	Retrieves and displays, in a brief list, messages found in *$user1*'s inbox folder.

Table 5.18: Function select_list_from_my_contacts( ), included in library messages.php

Function **select_list_from_my_contacts( )**		
Arguments	**Returned values**	**Description**
• **$user1 (String)** • **$match (String)**	None	Accesses *$user1*'s contacts and generates a select-list from which the user may select ORIENT members for different functions (e.g. recipients of a message, selection for an inspection team, etc.). If *$match* is supplied, the generated list has *$match* as preselected value.

Table 5.19: Function show_list( ), included in library messages.php

Function **count_contacts( )**		
Arguments	**Returned values**	**Description**
• **$list (String)**	None	Processes *$list*, (a string value comprising of usernames separated by space characters), and displays it as a list of full names that correspond to each username.

Table 5.20: Function count_list( ), included in library messages.php

Function **count_list( )**		
Arguments	**Returned values**	**Description**
• **$list (String)**	Integer value.	Counts the total number of usernames in *$list*. Returned value is the total number.

Table 5.21: Function send_message( ), included in library messages.php

Function **send_message( )**		
Arguments	**Returned values**	**Description**
• **$user (String)** • **$to_list (String)** • **$cc_list (String)** • **$subject (String)** • **$body (String)** • **$affiliaction (Character)**	None	Sends a message with *$subject*, *$body* and *$affiliation* from *$user* (sender) to *$to_list* and *$cc_list* (recipients).

Table 5.22: Function delete_message( ), included in library messages.php

Function **delete_message( )**		
Arguments	**Returned values**	**Description**
• **$sender (String)** • **$date (Date)** • **$recipient (String)**	None	Removes the message with sender *$sender* sent on *$date* from $recipient's inbox.

Table 5.23: Function delete_sent_message( ), included in library messages.php

Function **delete_sent_message( )**		
Arguments	**Returned values**	**Description**
• **$sender (String)** • **$date (Date)**	None	Removes a message sent on *$date* from the sender's *$sender* "Sent" messages folder.

Table 5.24: Function show_message( ), included in library messages.php

Function **show_message( )**		
Arguments	**Returned values**	**Description**
• **$user (String)** • **$from (String)** • **$date (Date)** • **$order (Integer)** • **$mode (Integer)**	None	Retrieves and displays a message from a message folder of *$user*, which was sent by *$from* on *$date*. *$mode* is used to specify which of the two message folders the user currently accesses. *$order* defines the ordering of the list of messages from which the user accessed the current message with the purpose of providing the user with the freedom to navigate through the list via "previous" and "next" message hyperlinks.

Library "contacts.php"

This library covers functions related to the management of a member's contacts, (e.g., removing a contact, searching for new contacts, counting contacts in a user's contacts list, etc.).

Table 5.25: Function count_contacts( ), included in library contacts.php

Function **count_contacts( )**		
Arguments	**Returned values**	**Description**
• **$username (String)**	Integer value.	Retrieves and counts the contacts of *$username*. Returned value is the total number of contacts.

Table 5.26: Function show_contacts( ), included in library contacts.php

Function **show_contacts( )**		
Arguments	**Returned values**	**Description**
• **$username (String)** • **$order (Integer)**	None	Retrieves and displays the contacts of *$username* in the specified *$order*.

Table 5.27: Function show_invited_contacts( ), included in library contacts.php

Function **show_invited_contacts( )**		
Arguments	**Returned values**	**Description**
• **$username (String)** • **$order (Integer)**	None	Retrieves and displays the members *$username* has extended an invitation to, in the specified *$order*.

Table 5.28: Function search( ), included in library contacts.php

Function **search( )**		
Arguments	**Returned values**	**Description**
• **$language (String)** • **$familiarity (String)** • **$expertise (String)** • **$fluency (String)** • **$order (Integer)** • **$username (String)**	MySQL result set	Performs a search among all registered members of ORIENT according to search criteria *$language*, *$familiarity*, *$expertise* and *$fluency* (if any of them have been defined) and orders the results according to *$order*. *$username* is used to exlude the user performing the search from the results.

Table 5.29: Function invite_contact( ), included in library contacts.php

Function **invite_contact( )**		
Arguments	**Returned values**	**Description**
• **$user1 (String)** • **$user2 (String)**	None	Lists *$user2* as one of *$user1*'s invited contacts and extends a message to *$user2* informing them about the

		invitation and offering them the choice to accept or decline the invitation.

Table 5.30: Function remove_contact( ), included in library contacts.php

Function **remove_contact( )**		
Arguments	**Returned values**	**Description**
• **$user1 (String)** • **$user2 (String)**	None	Validates that each user has the other one in their contact. If the validation yields positive results, the function removes *$user2* from *$user1*'s contacts and vice versa.

Table 5.31: Function remove_invcontact( ), included in library contacts.php

Function **remove_invcontact( )**		
Arguments	**Returned values**	**Description**
• **$user1 (String)** • **$user2 (String)**	None	Validates that *$user1* has indeed extended an invitation to *$user2* and afterwards it cancels / deletes the invitation.

Table 5.32: Function check_contact( ), included in library contacts.php

Function **check_contact( )**		
Arguments	**Returned values**	**Description**
• **$user1 (String)** • **$user2 (String)**	Boolean value.	Checks to see whether *$user2* (who is about to be invited by *$user1* to join their contacts) already is included in *$user1*'s contacts.

Table 5.33: Function valid_invitation( ), included in library contacts.php

Function **valid_invitation( )**		
Arguments	**Returned values**	**Description**
• **$from (String)** • **$date (Date)** • **$user (String)**	Boolean value.	Checks that invitation made to *$user* by $from on *$date* is still valid (i.e. *$from* has not cancelled it).

Library "studies.php"

This library contains functions related to the presentation and the management of studies (e.g., displaying running studies, displaying a published study, calculating the next step in the study for each member of the inspection team, etc.).

Table 5.34: Function show_Studies( )**, included in library** studies.php

Function **show_Studies( )**		
Arguments	**Returned values**	**Description**
• **$user (String)** • **$category (Integer)** • **$order (Integer)**	None	Retrieves and displays studies of *$category* in the specified *$order*. Studies are divided in 5 categories: running studies presented on the **homepage**, running studies presented in the **Studies** section, archived studies presented in the **Studies** section, published studies presented in the **Studies** section and published studies presented in the available to public **Published Studies** section. *$user* is used to sort out studies that relate to a specific user (necessary to present e.g. a user's running studies table, but not necessary for published

		studies in the public section of the tool).

Table 5.35: Function next_step( ), included in library studies.php

Function **next_step( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user2 (String)**	String value.	Finds out what the next step for *$user* in the study about *$system* is. This depends on the progress indicator of the study, the role of the specific inspection team member and the "documents" this inspection team member has already produced during the current phase of the study.

Table 5.36: Function show_published_study( ), included in library studies.php

Function **show_published_study( )**		
Arguments	**Returned values**	**Description**
• **$system (String)**	None	Retrieves all documents from the reporting phase of *$system*'s study and presents them in a simplified list.

Table 5.37: Function check_contact( ), included in library studies.php

Function **show_section_of_published_study( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$section (Integer)** • **$step (Integer)** • **$ugid (String)**	None	Retrieves and displays the content of "document" (which is specified by *$section*) from the reporting phase of *$system*'s study in view mode (i.e. no interaction with the data allowed besides browsing).

		$step and *$ugid* are used to retrieve sub-sections of the aforementioned "documents".

Table 5.38: Function assign_color( ), included in library studies.php

Function **assign_color( )**		
Arguments	**Returned values**	**Description**
• **$value (Integer)**	String value.	Assesses *$value* and matches it with a RGB colour value, as specified by the framework[ ].

Library "workspace.php"

The "workspace.php" library includes functions that surround the notion of a workspace for every study and supports the overall presentation and management of issues involving the study's "documents" and the inspection team.

Table 5.39: Function show_Study_panel( ), included in library workspace.php

Function **show_Study_panel( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$phase (Integer)**	None	Retrieves information about *$system*'s study and displays it in panel situated at the left part of the interface. The aforementioned information includes the name of the study, the inspection leader, the time period when the study takes place, the composition of the inspection team and a way of contacting them. *$phase* indicates the current phase of the study.

Table 5.40: Function show_inspection_team( ), included in library workspace.php

Function **show_inspection_team( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$order (Integer)**	None	Retrieves and presents the composition of the inspection team for *$system*'s study in the specified *$order*.

Table 5.41: Function show_history_stages( ), included in library workspace.php

Function **show_history_stages( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$stage (Integer)** • **$order (Integer)** • **$user (String)**	None	Retrieves and displays all existing "documents" for the specified *$stage* of *$system*'s study in the specified *$order*. Furthermore, it assesses the next step that the specific *$user* should take in order to complete their participation in the study.

Library "initiation_phase.php"

This library contains functions that facilitate an inspection leader in initiating a new study (i.e., establish a background for the study and form an inspection team).

Table 5.42: Function new_inspection( ), included in library initiation_phase.php

Function **new_inspection( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user (String)**	None	Displays a data form allowing the *$user* to document information concerning *$system*'s study, retrieves and displays for editing by *$user* the stored data concerning *$system*'s study or retrieves and displays in read-only mode

		stored information concerning *$system*'s study.

Table 5.43: Function select_Study_team( ), included in library initiation_phase.php

Function **select_study_team( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$inspectors (Integer)** • **$designers (Integer)** • **$user (String)** • **$error (Boolean)**	None	Displays a data form allowing the *$user* to document information concerning the composition of the inspection team for *$system*'s study, retrieves and displays for editing by *$user* the stored data concerning the composition of the inspection team for *$system*'s study or retrieves and displays in read-only mode stored information concerning the composition of the inspection team for *$system*'s study. *$inspectors* and *$designers* dictate the number of inspectors and designers participating in the inspection team. *$error* informs the system that an error has occurred.

Library "set_up_phase.php"

This library contains functions related to the steps an inspection leader has to take during the set-up phase of a study (i.e., describe the system under assessment, define user groups and functions of importance to them that will be assessed, and finally document a context of use for each user group).

Table 5.44: Function setup_description( ), included in library set_up_phase.php

Function **setup_description( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user (String)**	None	Displays a data form allowing the *$user* to describe *$system*, retrieves and displays for editing by *$user* the stored data concerning *$system*'s description or retrieves and displays in read-only mode stored information concerning *$system*'s description.

Table 5.45: Function show_user_groups( ), included in library set_up_phase.php

Function **show_user_groups( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user (String)**	None	Retrieves and displays all user groups that have already been defined for *$system*'s study.

Table 5.46: Function new_user_group( ), included in library set_up_phase.php

Function **new_user_group( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user (String)** • **$ugid (String)** • **$functions (Integer)**	None	Displays a data form allowing the *$user* to document a new user group for *$system*'s study, retrieves and displays stored information about user group *$ugid* for editing or retrieves and displays in read-only mode stored information concerning user group *$ugid*'s study.

		$functions represents the number of system functions defined for each user group.

Table 5.47: Function show_functions_per_user_group( ), included in library set_up_phase.php

Function **show_functions_per_user_group( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user (String)** • **$ugid (String)**	None	Retrieves and displays information concerning the system functions that have been defined for user group *$ugid*. Users may edit some of the functions attributes depending on their privileges.

Table 5.48: Function show_context_of_use( ), included in library set_up_phase.php

Function **show_context_of_use( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user (String)** • **$ugid (String)** • **$category (Integer)** • **$mode (Integer)** • **$more (Integer)** • **$sno (Integer)** • **$error (Boolean)**	None	Retrieves and displays all information about the context of use for user group *$ugid* sorted by categories. More over, it allows users to define new or edit requirements (identified by *$category* and *$sno*) depending on their privileges. *$mode* determines if the user has selected to save or load a context-of-use record. *$more* informs the function that the user needs more requirements slots.

		$error is a Boolean value indicating when an error has occurred.

Library "inspection_phase.php"

This library contains functions that have to do with the steps an inspection team member has to take during the inspection phase of a study (i.e., assess the system in terms of visibility, usefulness, availability, quality of interaction and relationship maintainability).

Table 5.49: Function inspection_profile( ), included in library inspection_phase.php

Function **inspection_profile( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user (String)** • **$access (String)** • **$error (Boolean)**	None	Allows the *$user* to create or edit their personal inspector profile concerning the evaluation of *$system*. Retrieves and displays a *$user*'s individual inspector profile in read-only mode for user *$access*. *$error* is a Boolean value indicating when an error has occurred.

Table 5.50: Function inspect_visibility( ), included in library inspection_phase.php

Function **inspect_visibility( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user (String)** • **$ugid (String)** • **$sno (Integer)** • **$step (Integer)** • **$access (String)** • **$error (Boolean)**	None	Allows the *$user* to document new or edit old (identification of old practices is done by means of a user group id *$ugid* and a practice's serial number *$sno*) identified good / bad practices concerning the visibility of *$system*.

		Retrieves and displays a *$user*'s identified practices in read-only mode for user *$access*. *$step* determines the current sub-step of the process of assessing *$system*'s visibility. *$error* is a Boolean value indicating when an error has occurred.

Table 5.51: Function inspect_usefulness( ), included in library inspection_phase.php

Function **inspect_usefulness( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user (String)** • **$ugid (String)** • **$sno (Integer)** • **$step (Integer)** • **$access (String)** • **$error (Boolean)**	None	Allows the *$user* to document new or edit old (identification of old practices is done by means of a user group id *$ugid* and a practice's serial number *$sno*) identified good / bad practices concerning the perceived usefulness and ease of use of *$system*. Retrieves and displays a *$user*'s identified practices in read-only mode for user *$access*. *$step* determines the current sub-step of the process of assessing *$system*'s usefulness. *$error* is a Boolean value indicating when an error has occurred.

Table 5.52: Function inspect_availability( ), included in library inspection_phase.php

Function **inspect_availability( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user (String)** • **$ugid (String)** • **$sno (Integer)** • **$step (Integer)** • **$access (String)** • **$error (Boolean)**	None	Allows the *$user* to document new or edit old (identification of old practices is done by means of a user group id *$ugid* and a practice's serial number *$sno*) identified good / bad practices concerning the availability and approachability of *$system*. Retrieves and displays a *$user*'s identified practices in read-only mode for user *$access*. *$step* determines the current sub-step of the process of assessing *$system*'s availability. Furthermore, *$step* represents the level of experience of users to whom the specific practices refer. *$error* is a Boolean value indicating when an error has occurred.

Table 5.53: Function inspect_quality( ), included in library inspection_phase.php

Function **inspect_quality( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user (String)** • **$ugid (String)** • **$func_id (String)** • **$sno (Integer)**	None	Allows the *$user* to document new or edit old (identification of old practices is done by means of a user group id *$ugid*, a function id *$func_id* and a

• **$step (Integer)** • **$access (String)** • **$error (Boolean)**		practice's serial number *$sno*) identified good / bad practices concerning the quality of interaction of *$system*. Retrieves and displays a *$user*'s identified practices in read-only mode for user *$access*. *$step* determines the current sub-step of the process of assessing *$system*'s quality of interaction. Furthermore, *$step* represents the level of experience of users to whom the specific practices refer. *$error* is a Boolean value indicating when an error has occurred.

Table 5.54: Function inspect_maintainability( ), included in library inspection_phase.php

Function **inspect_maintainability( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user (String)** • **$ugid (String)** • **$sno (Integer)** • **$step (Integer)** • **$access (String)** • **$error (Boolean)**	None	Allows the *$user* to document new or edit old (identification of old practices is done by means of a user group id *$ugid* and a practice's serial number *$sno*) identified good / bad practices concerning the relationship maintainability of *$system*. Retrieves and displays a *$user*'s identified practices in read-only mode for user

		$access. *$step* determines the current sub-step of the process of assessing *$system*'s relationship maintainability. Furthermore, *$step* represents the level of experience of users to whom the specific practices refer. *$error* is a Boolean value indicating when an error has occurred.

Library "reporting_phase.php"

This library includes functions related to the steps an inspection leader has to take during the reporting phase of a study (i.e., produce summative forms that report findings concerning the system's visibility, usefulness, availability, quality of interaction and relationship maintainability).

Table 5.55: Function report_team( ), included in library reporting_phase.php

Function **report_team( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$access (String)**	None	Retrieves and displays information about the inspection team using as sources the individual inspector profiles by each member. Average values and aggregation of text values is done automatically, where this is needed. *$user* may edit some of these values, depending on their privileges.

Table 5.56: Function report_visibility( ), included in library reporting_phase.php

Function **report_visibility( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user (String)** • **$ugid (String)** • **$sno (Integer)** • **$error (Boolean)**	None	Retrieves and displays good / bad practices identified by all inspectors concerning the *$system*'s visibility. Average values are calculated and aggregation of text values is done by the system, where this is needed. *$user* may merge practices or edit (in order to identify a practice a user group id *$ugid* and a serial number *$sno* are needed) some of their values, depending on their privileges. *$error* is a Boolean value indicating when an error has occurred.

Table 5.57: Function report_usefulness( ), included in library reporting_phase.php

Function **report_usefulness( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user (String)** • **$ugid (String)** • **$sno (Integer)** • **$error (Boolean)**	None	Retrieves and displays good / bad practices identified by all inspectors concerning the *$system*'s perceived usefulness and ease of use. Average values are calculated and aggregation of text values is done by the system, where this is needed. *$user* may merge practices or edit (in order to identify a practice a user group id *$ugid*

		and a serial number *$sno* are needed) some of their values, depending on their privileges. *$error* is a Boolean value indicating when an error has occurred.

Table 5.58: Function report_availability( ), included in library reporting_phase.php

Function **report_availability( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user (String)** • **$ugid (String)** • **$sno (Integer)** • **$step (Integer)** • **$error (Boolean)**	None	Retrieves and displays good / bad practices identified by all inspectors concerning the *$system*'s availability and approachability. Average values are calculated and aggregation of text values is done by the system, where this is needed. *$user* may merge practices or edit (in order to identify a practice a user group id *$ugid* and a serial number *$sno* are needed) some of their values, depending on their privileges. *$step* determines the current sub-step of the process of reporting the *$system*'s availability. Furthermore, *$step* represents the level of experience of users to whom the specific practices refer. *$error* is a Boolean value indicating when an error has

		occurred.

Table 5.59: Function report_quality( ), included in library reporting_phase.php

Function **report_quality( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user (String)** • **$ugid (String)** • **$func_id (String)** • **$sno (Integer)** • **$step (Integer)** • **$access (String)** • **$error (Boolean)**	None	Retrieves and displays good / bad practices identified by all inspectors concerning the *$system*'s quality of interaction. Average values are calculated and aggregation of text values is done by the system, where this is needed. *$user* may merge practices or edit (in order to identify a practice a user group id *$ugid*, a function id *$func_id* and a serial number *$sno* are needed) some of their values, depending on their privileges. *$step* determines the current sub-step of the process of reporting the *$system*'s quality of interaction. Furthermore, *$step* represents the level of experience of users to whom the specific practices refer. *$error* is a Boolean value indicating when an error has occurred.

Table 5.60: Function report_maintainability( ), included in library reporting_phase.php

Function **report_maintainability( )**		
Arguments	**Returned values**	**Description**
• **$system (String)** • **$user (String)** • **$ugid (String)** • **$sno (Integer)** • **$step (Integer)** • **$access (String)** • **$error (Boolean)**	None	Retrieves and displays good / bad practices identified by all inspectors concerning the *$system*'s relationship maintainability. Average values are calculated and aggregation of text values is done by the system, where this is needed. *$user* may merge practices or edit (in order to identify a practice a user group id *$ugid* and a serial number *$sno* are needed) some of their values, depending on their privileges. *$step* determines the current sub-step of the process of reporting the *$system*'s relationship maintainability. Furthermore, *$step* represents the level of experience of users to whom the specific practices refer. *$error* is a Boolean value indicating when an error has occurred.

Library "functions.php"

This library includes functions that perform validation on their arguments and are used throughout the system (e.g., validation of an e-mail address, validation of a password, etc.).

Table 5.61: Function valid_email( ), included in library functions.php

Function **valid_email( )**		
Arguments	**Returned values**	**Description**
• **$email (String)**	Boolean value.	Validates the specified *$email* address against a set of syntax rules (i.e., a regular expression). Returned value is a Boolean value representing the result of the aforementioned validation.

Table 5.62: Function valid_userName( ), included in library functions.php

Function **valid_userName( )**		
Arguments	**Returned values**	**Description**
• **$name (String)**	Boolean value.	Validates the specified *$name* against a set of syntax rules (i.e., a regular expression). Returned value is a Boolean value representing the result of the aforementioned validation.

Table 5.63: Function valid_password( ), included in library functions.php

Function **valid_password( )**		
Arguments	**Returned values**	**Description**
• **$pwd (String)**	Boolean value.	Validates the specified *$pwd* against a set of syntax rules (i.e., a regular expression). Returned value is a Boolean value representing the result of the aforementioned validation.

Table 5.64: Function check_password( ), included in library functions.php

Function **check_password( )**		
Arguments	**Returned values**	**Description**
• **$password (String)** • **$username (String)**	Boolean value.	Validates that a *$username* with the specified *$password* is stored in the tool's database. Returned value is a Boolean value representing the result of the aforementioned validation.

Table 5.65: Function check_username( ), included in library functions.php

Function **report_maintainability( )**		
Arguments	**Returned values**	**Description**
• **$username (String)**	Boolean value.	Checks whether *$username* is already in use by another registered member. Returned value is a Boolean value representing the result of the aforementioned check.

Table 5.66: Function check_mail( ), included in library functions.php

Function **check_mail( )**		
Arguments	**Returned values**	**Description**
• **$mail (String)**	Boolean value.	Checks whether *$mail* is already in use by another registered member. Returned value is a Boolean value representing the result of the aforementioned check.

Table 5.67: Function precede( ), included in library functions.php

Function **precede( )**		
Arguments	**Returned values**	**Description**
• **$d1 (Date)** • **$d2 (Date)**	Boolean value.	Assesses the chronological order between $d1 and $d2 (i.e., which date precedes the other one). Returned value is a Boolean value representing the result of the aforementioned check.

Library "general.php"

This library includes functions that shape the content of web-pages for the public view (i.e., sections that the user can access without having to be logged in) of the ORIENT inspection tool.

Table 5.68: Function introduction( ), included in library general.php

Function **introduction( )**		
Arguments	**Returned values**	**Description**
• **$section (Integer)**	None	Retrieves and displays the proper content for the **Introduction** section of the public view of ORIENT according to the specified sub-section *$section*.

Table 5.69: Function about( ), included in library general.php

Function **about( )**		
Arguments	**Returned values**	**Description**
• **$section (Integer)**	None	Retrieves and displays the proper content for the **About ORIENT** section of the public view of ORIENT according to the specified sub-section *$section*.

Table 5.70: Function background( ), included in library general.php

Function **background( )**		
Arguments	**Returned values**	**Description**
• **$section (Integer)**	None	Retrieves and displays the proper content for the **Theoretical background** section of the public view of ORIENT according to the specified sub-section *$section*.

5.3.2 Messages implementation

Messages, as mentioned in previous sections, are employed as a mechanism to facilitate the communication between members of ORIENT. Their implementation for ORIENT resembles the way Java's garbage collector functions [41].

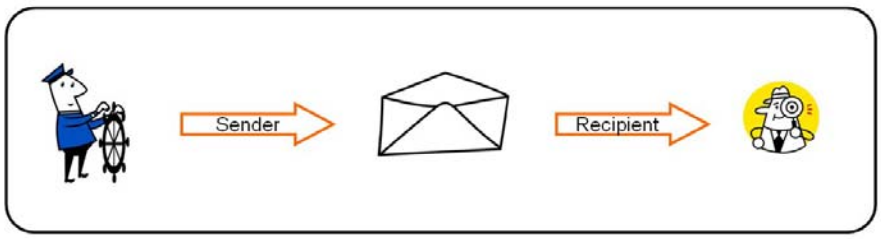

Figure 5.42: Two members referencing the same message.

Messages do not "belong" to any member. Instead they are created as individual entities. Any member related to a message, either as the sender or the recipient of it, acts as a reference to the message (figure 5.1). When a member decides that they no longer need the message and delete it, then this relationship between the member and the message is deleted as well. However, the message itself is (usually) only "virtually" deleted (figure 5.2).

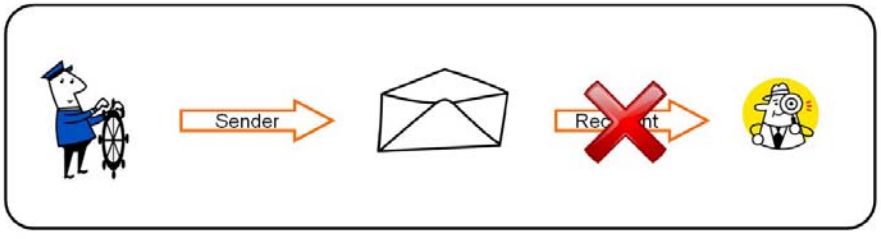

Figure 5.43: Virtual deletion of a message (i.e., there still exists at least one member referencing the message).

When a relationship between a member and a message is severed, the system checks whether the message is referenced by other members as well. If no members still reference the message, then it is actually deleted from the system (figure 5.3).

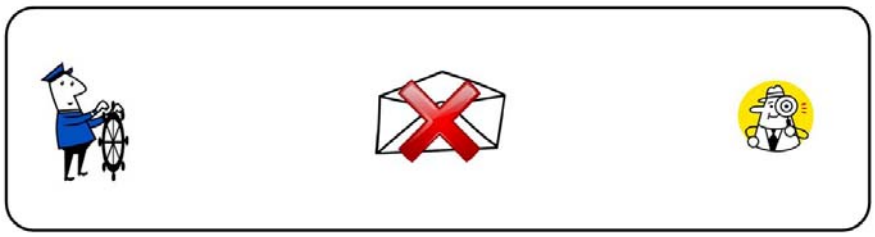

Figure 5.44: Deletion of a message (i.e., no members are referencing the message anymore).

This implementation saves disk space required for the storage of messages, as all users "related" to a message view the same message instead of a "personalized" copy of the original message.

5.3.3 Access rights to pages

Each member of ORIENT should be granted a different level of access to sections of the tool. For example, it would not be appropriate (or wise) to grant the right to alter the content of a study to a third party who was not included in the inspection team. Therefore, it is important to create a mechanism that authorizes access to sections of ORIENT according to user's privileges.

This mechanism is incorporated into every function that retrieves and presents information. In general, each function takes into consideration the user that requests access to information and their privileges (i.e., is this user a member of the inspection team? If so, is this user an inspection leader, an inspector or a designer?) and presents information in a way suitable for the specific level of access. Groups of authorized actions are listed as follows:

- Non-members of the inspection team can access a study only if it has been published (and even then, only in read-only mode).
- All members of the inspection team have access to any section of a study they participate (or have participated) in. This access is granted in read-only mode, however.
- Only the inspection leader of a study is authorized to perform actions that affect the study as a whole (e.g., abort a study, publish a study, etc.).
- "Documents" produced during the initiation, set-up and reporting phase of a study are available for editing only by the study's inspection leader.

- Evaluation "documents" can only be edited by the inspector that produced them in the first place.

5.3.4 Error messages

Error messages, generated during the use of the ORIENT tool, are presented in red colour, as it is the colour used more commonly for conveying to the user that an error has occurred [42]. More over, they are presented at the top part of each screen and always at the same location making it easier for the user to notice them even if they suffer from a colour-deficiency (figure 5.4).

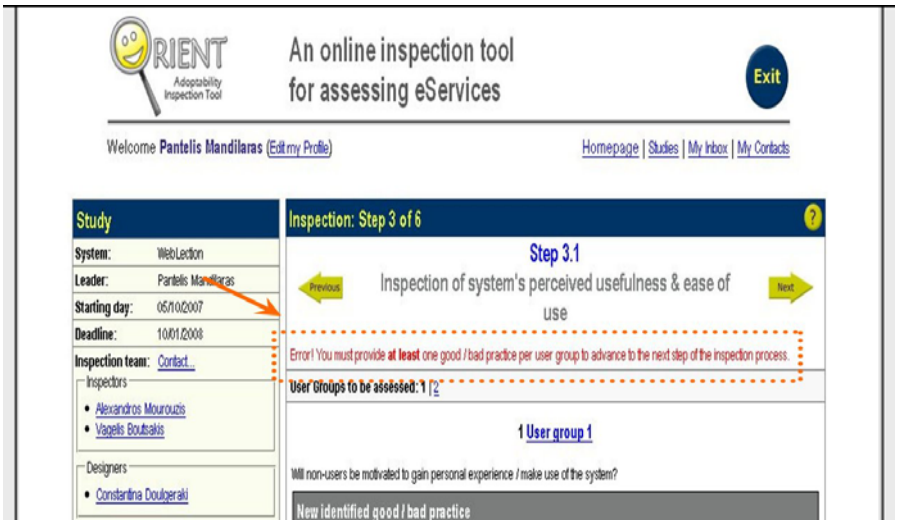

Figure 5.45: Error messages are always presented underneath the main content's heading in red colour.

Error messages occur in general in two situations, when the user has neglected to fill-in a required field and when the user attempts to move on to the next step without having completed the current step first. In the first case, errors are discovered after the user has submitted the form and it has been processed by the system. Typically, the page that triggered the error is reloaded, the user is informed of the "mistake" they made and a short recommendation for fixing it is supplied.

In the second case, errors are "foreseen" before they occur. For example, when the section of the inspection tool related to the assessment of the system's visibility is formed by the respective function, the system checks to see whether the user has already identified at least one practice for each user group concerning the system's visibility. If they have not, hyperlinks (which would under other circumstances transfer the user to the next step) are loaded with a target URL that would trigger the appearance of an error.

5.3.5 Virtual calendar

In order to eliminate the chance of errors concerning date values, they are defined by means of a virtual calendar. The tool used for this purpose is the freeware program CodeThatCalendar JavaScript Calendar 3.2.1[10]. The only changes made to the above tool involved CSS formatting to ensure a harmonious appearance among the different elements of ORIENT.

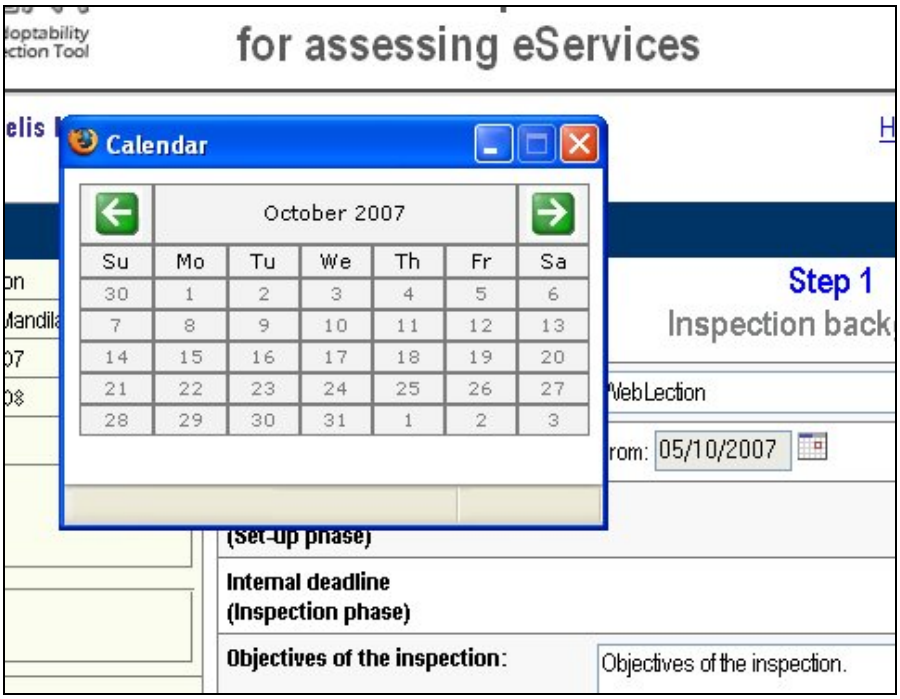

Figure 5.46: The virtual calendar.

[10] Available at: http://www.softizer.com/show_product/web_authoring/javascript/codethatcalendar_javascript_calendar/

6 Evaluation

6.1 Heuristic evaluation

6.1.1 Process and method

The first phase of evaluation for the ORIENT online inspection tool took place as soon as a working prototype of the system was available. A team of three usability experts was formed in order to conduct a heuristic evaluation on the prototype version of the system. Each individual evaluator inspected the interface alone, documenting any usability issues that according to his/her judgement needed to be dealt with.

All problems, located during this evaluation, were documented, as well as what general principle for user interface design (*heuristic*) they refer to. A summative list of all the problems that were identified was formed at the end of the inspection and afterwards each individual evaluator provided a severity rating for every problem on the list, taking into consideration the frequency, the impact and the persistence of the problem [24]. The ten heuristics used for this evaluation, as defined by Jakob Nielsen [25], are:

(1) Visibility of system status,
(2) Match between system and the real world,
(3) User control and freedom,
(4) Consistency and standards,
(5) Error prevention,
(6) Recognition rather than recall,
(7) Flexibility and efficiency of use,
(8) Aesthetic and minimalist design,
(9) Help users recognize, diagnose and recover from errors and
(10) Help and documentation.

The final list of problems, identified during this phase of the evaluation of the ORIENT inspection tool, is recorded in table 6.1. Besides the documentation of the problems, heuristics that each problem is related to, as well as their average severity rating, are presented.

Table 71: The final list of problems, as documented through the heuristic evaluation of the ORIENT inspection tool.

Description of problem	Heuristic principle (s)	Average rating
There are links in grey colour and with no underlining.	(1) (4) (8)	4
The label "history" at the bottom of the "Study" panel is neither clear, nor informative.	(2)	2
The four links at the bottom of the "Study" panel, used as a quick access to the four different phases of a study, are not presented in a way that clearly indicates their order within the sequence of steps for a study.	(2)	2
In a study's workspace, the first thing a user sees regardless of which phase they may select to view is the inspection team. This may confuse / disorient the user.	(1)	2
During each phase of a study, there are certain buttons located at the bottom left of each screen. These remain inactive during almost every step of the study and become active **only** during the final stage of each phase. Their purpose is to demonstrate that the user cannot advance to the next phase of a study, unless they have completed all steps of the current phase. However, having a control element constantly visible and rarely usable may confuse the user.	(2) (5)	2
The function of the "pause" button is not clear to the user. Sometimes pressing it saves data and transfers the user back to the study's workspace. Most of the times, it simply transfers the user back to the study's workspace.	(5)	3
Buttons used for previous and next steps are presented side by side, which is moderately clear to the user. A more natural mapping would be to situate them at the left and right (for previous and next respectively) corners of each screen.	(2)	1
Sometimes users have not gone though the whole page and still need to advance to the next step. It is not a good practice to "force" the user to scroll down to the bottom	(3) (7)	2

of the page to locate the previous / next arrows.		
Not all buttons have the same look and feel.	(4) (8)	1
Buttons do not demonstrate the same "behaviour" for different browsers. For example, in Mozilla Firefox, their inactive status is barely visible.	(4) (8)	2
Links of the nature "Add more…" do not behave as the user would expect them to. They simply increase the available slots, whereas they should increase the available slots and retain the information the user has filled in so far.	(2) (7)	3
Tips and suggestions in the system would be more visible if the whole text was presented in colour instead of just their label.	(1)	0
Links made up of ORIENT member's full names act as a quick access to a member's profile. However, when these appear close to other links with completely different functionality (e.g. study panel, messages' inbox, etc.), they confuse the user and may cause them to follow wrong courses of action.	(5)	1
The constitution of the inspection team is an important piece of information that the user should have access to all the time. However, a more suitable location for it would be the study's panel, as its purpose is exactly a permanent short presentation of the most important information of a study.	(1) (6)	2
At a study's workspace, for every phase of the study, the most important pieces of information a user could get is the documentation of the steps that inspection team members have already completed. As such, this documentation should be presented closer to the top of the screen and **not** at the bottom.	(2) (7)	3
The "continue study" button is not easily noticed by the user, which may lead to the user not knowing how to continue with the study. A good solution would be to duplicate the button's functionality and assign it to the	(1) (7)	4

links currently leading to the study's workspace, such as the running studies table presented on the homepage and in the "Studies" section.		
On the system description step of the set-up phase, the label "access information" is not particularly informative about the nature of the input that the user must supply in the respective text field.	(2)	0
Generally speaking, functions "add" and "delete" should not co-exist on the same page. Even more when they are presented one on top of the other.	(5)	3
As the user adds new user groups, the list of added user groups increases in size. As a result the "add new user group" panel is soon pushed off the visible area of the screen. The user if forced to remember where it is and scroll down to it every time they need it.	(1) (6) (7) (8)	3
During the set-up phase, it is logical to assume that every new entry (user group, function, reference) is one that interests the user or else they wouldn't go into the trouble of making one in the first place. Consequently, it is also safe to assume that the field "to assess: yes or no" should be default be set to "yes".	(7)	2
The presentation of information on the system functions stage is too "crowded", confusing.	(7) (8)	3
Though the user is asked to sort functions according to priority, the respective buttons reside at the bottom of the page. Thus, they are not easily noticed by the user and even if the user locates them, they still have to scroll up/down significantly in order to perform the selected task.	(1) (7)	4
Section "individual inspector's profile" is not clearly marked as a sub-part of the inspection phase. Users may wonder why they ended up there when all they wanted was to continue the inspection process.	(1)	2
Typically, form buttons are placed at the bottom part of the form and not on its right. Therefore, buttons on pages regarding the inspection and reporting phases should be	(4) (5) (7)	2

moved. Their current location may confuse / disorient users.		
During the inspection phase, adding a new good/bad practice is the most common task a user may carry out. This is not reflected in its order in the layout (its location is at the bottom of the screen).	(2)	3
The way system functions of a user group are presented during the inspection of the quality of usage experience does not clearly demonstrate that these functions are "related" to the specific highlighted user group. Users may think that navigation among functions and user groups is arbitrary.	(1) (2) (5)	3
When the user re-visits stages of the evaluation process that contain sub-steps, there should be a way to quickly jump to a specific sub-step.	(7)	2
To avoid errors during the reporting phase, the "edit verbalization of practice" panel should be presented separately from the summative presentation of documented practices.	(5)	2

6.1.2 Problems and solutions

In order to address the problems documented in table 6.1, several changes in the design of ORIENT's interface had to be made. The majority of changes that were made affected several parts of the interface. Most of these changes will be presented in this section selecting at random some of the aforementioned parts of the interface. In order to distinguish the changes in the layout more easily, old and new design mock-ups will be presented together.

Section: "Study's workspace"

The original layout of the section of a study's workspace is presented in figure 6.1.

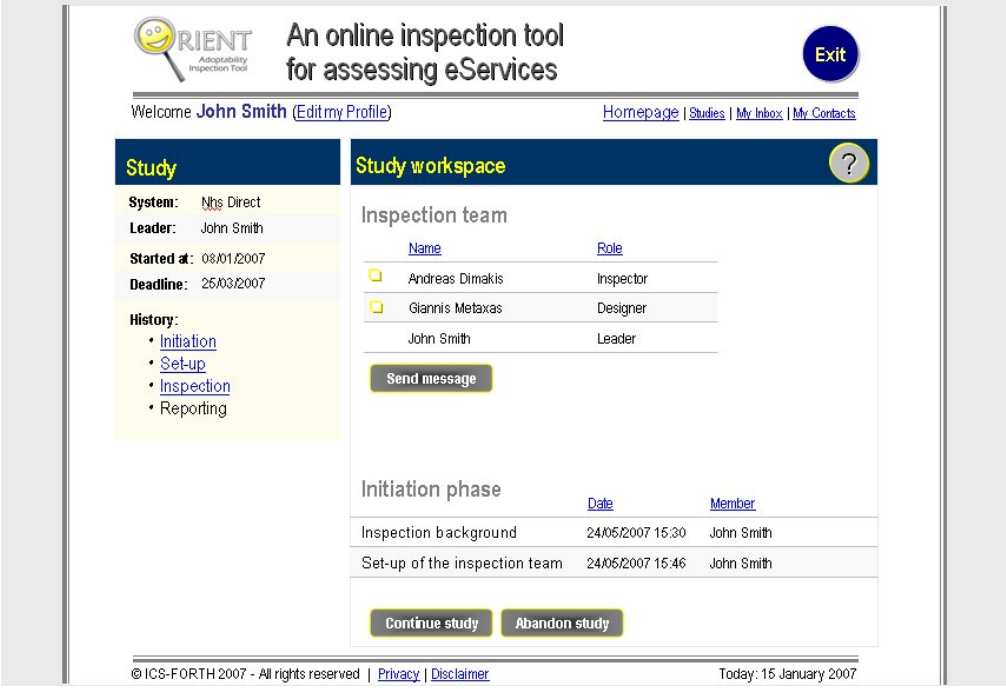

Figure 6.47: A study's workspace (original layout).

Taking into consideration the list of problems that were identified during the heuristic evaluation of the system, a new layout was designed (figure 6.2).

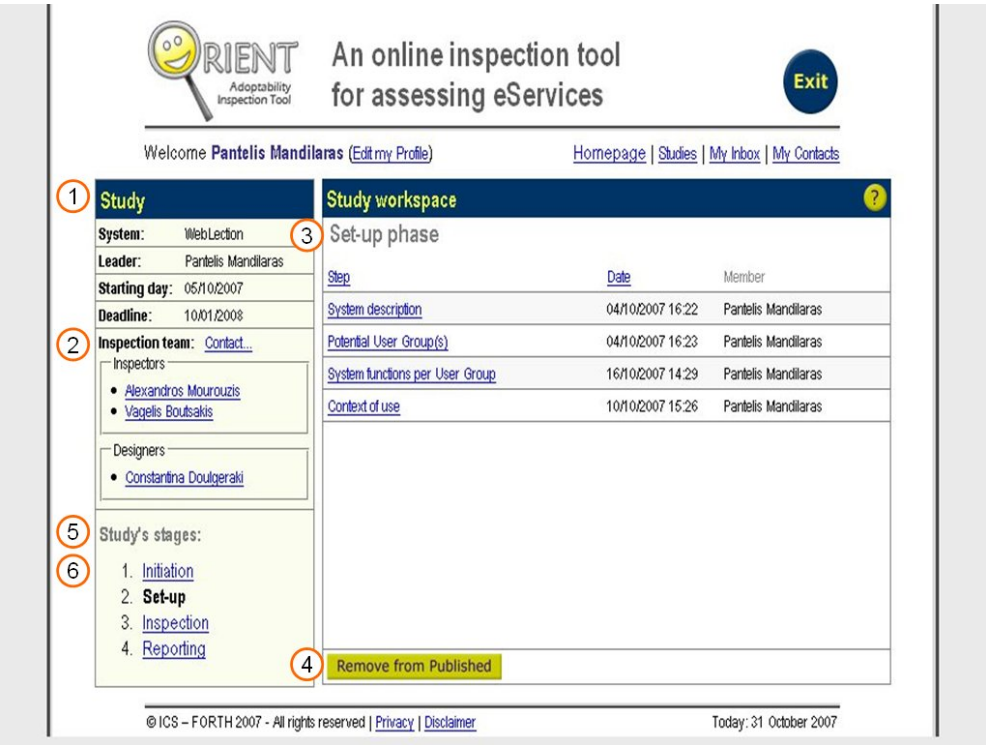

Figure 6.48: A study's workspace (redesigned layout).

The study's panel offers more information in the new layout, but in a more condensed way **(1)**. The inspection team's constitution has been added to the study's panel **(2)**, allowing for the documentation of steps for each study stage to be presented at the beginning of the main content section **(3)**. A new uniform set of buttons has been used throughout the system **(4)**. The label "history" has been changed to "Study's stages" **(5)**, which is more informative and finally the list containing the stages of the study has been numbered **(6)**, thus providing the user with a logical hint that the sequence stages appear in the list is defined by a logical sequence of succession within the progress of a study.

Section: "Initiation of a new study (step2)"

The original layout of this section is presented in figure 6.3.

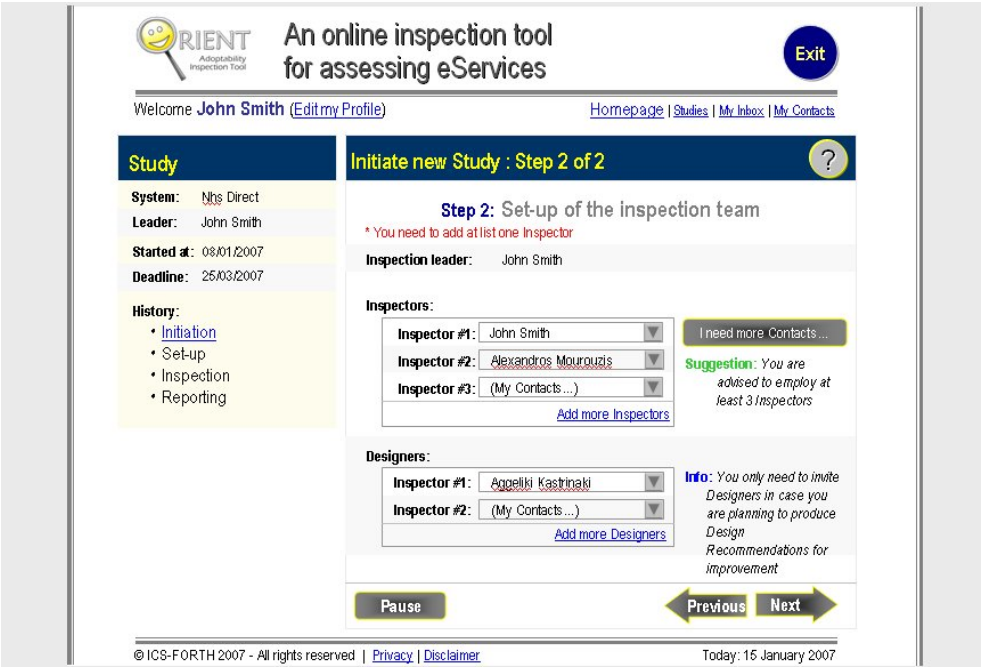

Figure 6.49: Set-up of the inspection team (original layout).

The new layout for this section, according to the findings of the heuristic evaluation of the system, is presented in figure 6.4.

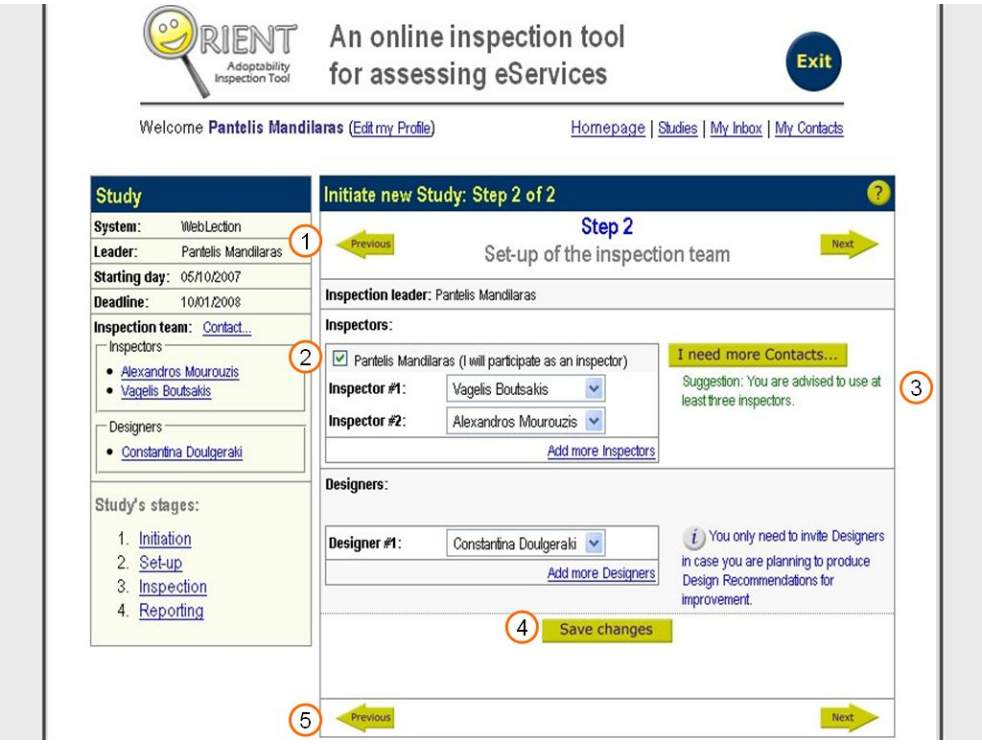

Figure 6.50: Set-up of the inspection team (redesigned layout).

Navigation arrows have been added at the top of the content in addition to the ones at the bottom of the content **(1)**. Moreover, their appearance has changed to be in accordance with the new look & feel of the buttons and finally "previous" and "next" buttons were placed at a distance to enhance the metaphor of moving backwards / forwards respectively **(1)(5)**. The "pause" button has been omitted as there was a high possibility that users would get confused about its purpose / functionality **(5)**. The inspection leader's participation appears in the inspectors' or designers' panel according to the aptitude that the inspection leader has declared in his / her profile (i.e. "more experienced as" field in a user's profile) **(2)**. The full text of suggestions / hints and helpful information pieces / tips is displayed in the respective colour (green for hints and blue for tips), instead of just their label **(3)**. The "save changes" button was moved out of the content's footer area and placed at the end of every form, because it refers to the content of the page, while the previous / next arrows refer to the navigation **(4)**.

Section: "Potential user group(s)"

The original layout of this section is presented in figure 6.5.

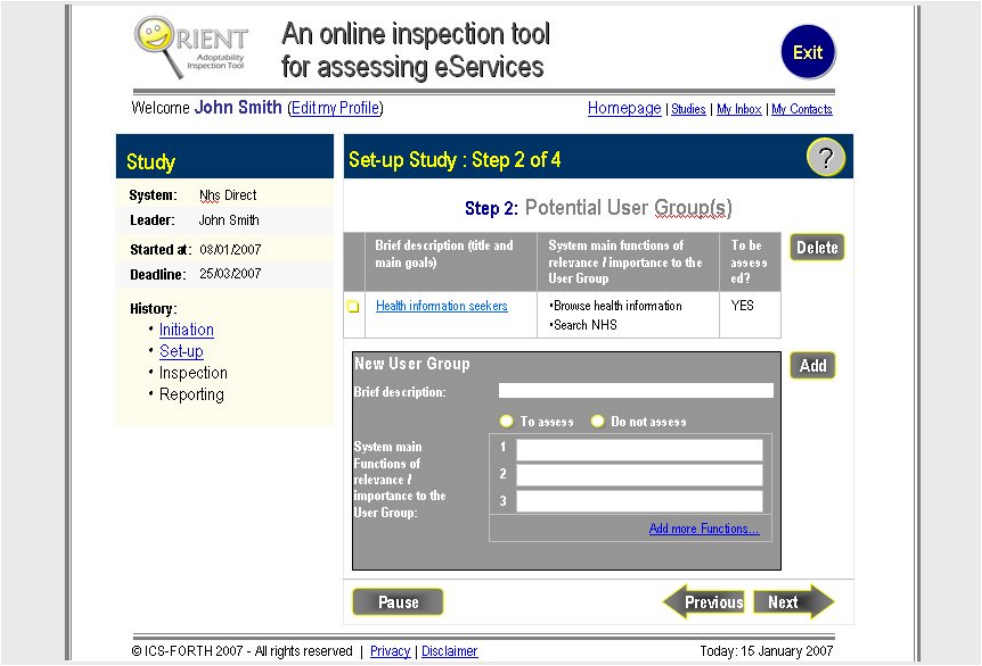

Figure 6.51: Documentation of potential user group(s) (original layout).

The new layout for this section (and for similar ones, which used to contain functions of the nature of "add" and "delete" in one screen) consists of two screens instead of one. The new layout, according to this change and other remarks made by the evaluators during the heuristic evaluation of the system, is presented in figures 6.6 and 6.6.

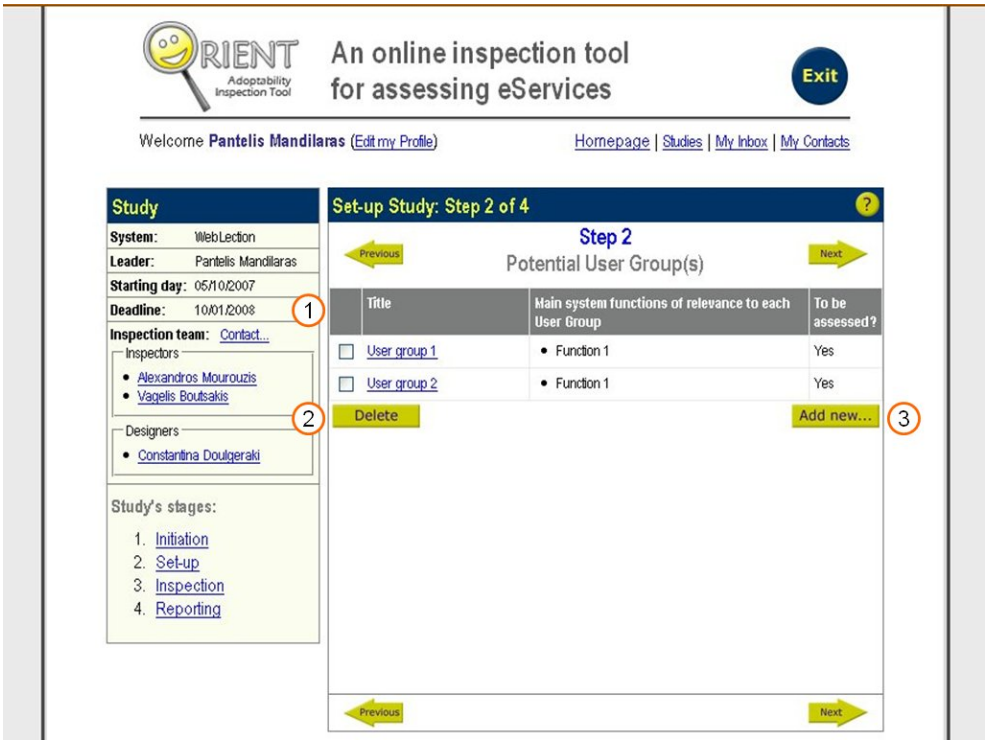

Figure 6.52: Documentation of potential user group(s) (part a) (redesigned layout).

Potential user groups that have already been defined by the user are presented in a table at the top of the content section **(1)**. The layout of the table itself has been slightly altered (change in dimensions and percentages each column covers of the table's total width), in order to take advantage of the space that the "delete" button used to be. The "delete" button has been moved to a more suitable location, which helps the user realize the relationship between the button and the checkboxes next to each user group **(2)**. Finally, a new button "add new…" has been added as a way of accessing the "new user group" panel that in this redesigned layout has been moved to a different screen. This new button is placed as far as possible from the "delete" button, to ensure that users will not accidentally press one instead of the other and to also highlight the difference in the nature of the functions they perform **(3)**.

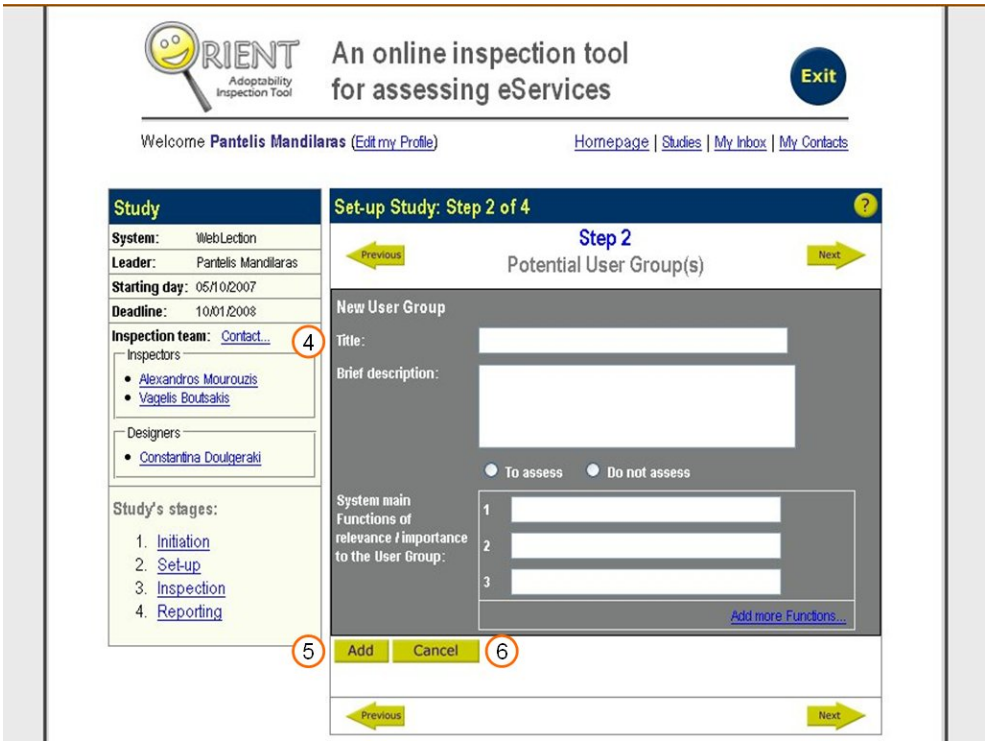

Figure 6.53: Documentation of potential user group(s) (part b) (redesigned layout).

In the redesigned version of the "new user group" panel, there is a clear distinction between the title and the brief description of the user group **(4)**. The "add" button has been moved at the bottom of the panel, which fits the natural movement of the user better (i.e., follows the direction in which the user scrolls down the panel and fills-in information) **(5)**. Finally, a new "cancel" button has been added which acts as a way to leave this screen and return to the previous one **(6)**.

Section: "System functions per user group"

The original layout of this section is presented in figure 6.8.

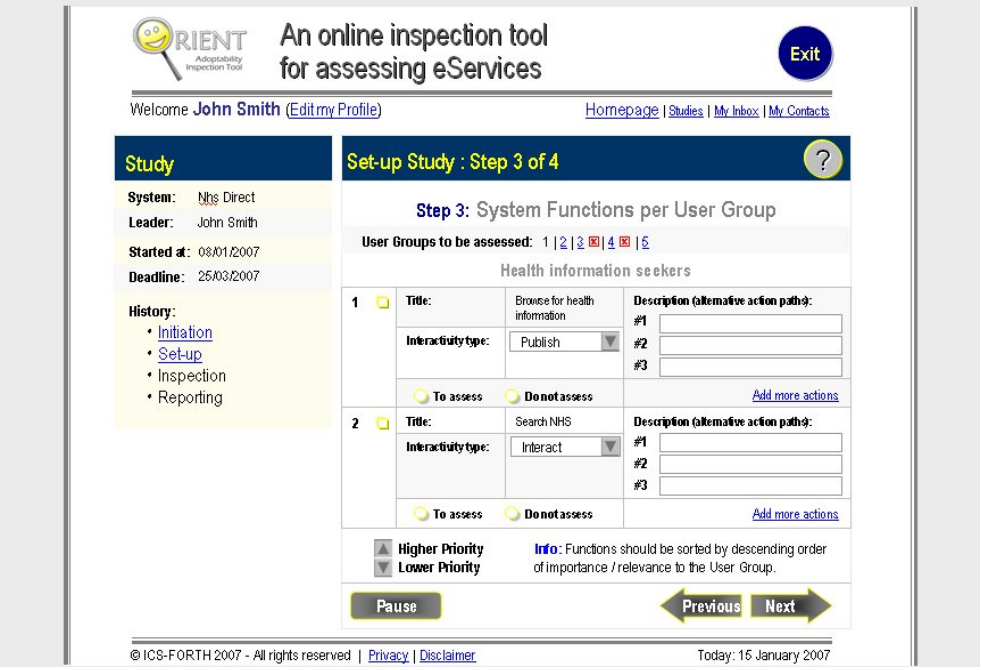

Figure 6.54: Documentation of system functions per user group (original layout).

As the heuristic evaluation findings demonstrated that the original layout was overcrowded, the new layout for this section takes up more space and groups information in a different way, as shown in figure 6.9.

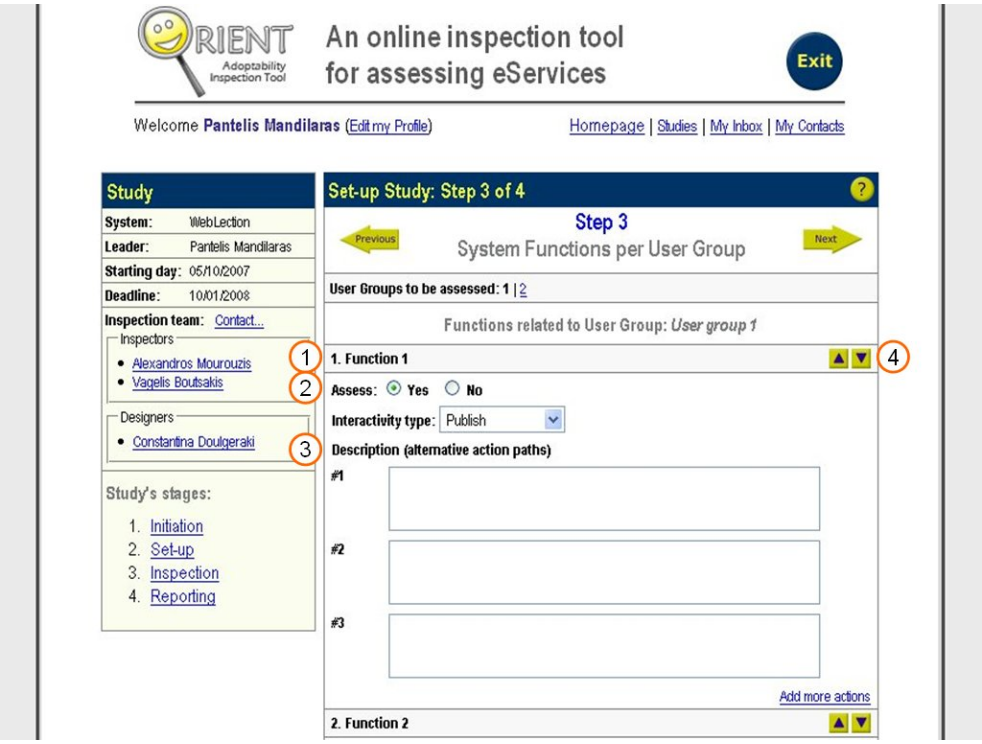

Figure 6.55: Documentation of system functions per user group (redesigned layout).

A function's title is presented as the title of a box that holds all information related to the specific function **(1)**. Labels for the "assess" field have been changed for a more concise presentation **(2)** and the alternative action paths' fields have been increased in size to allow the user to enter more text. This way, users may be more descriptive without having to move back and forth within the field in order to have an overview of the information they have already filled in **(3)**. Finally, the buttons for adjusting each function's priority have been "embedded" into the presentation of each function instead of residing elsewhere. Thus, it is more clear which function they are related to, and the user has a better overview of the function they wish to move up / down the ordered presentation of functions **(4)**.

Section: "Inspection of system's visibility"

The original layout of this section is presented in figure 6.10.

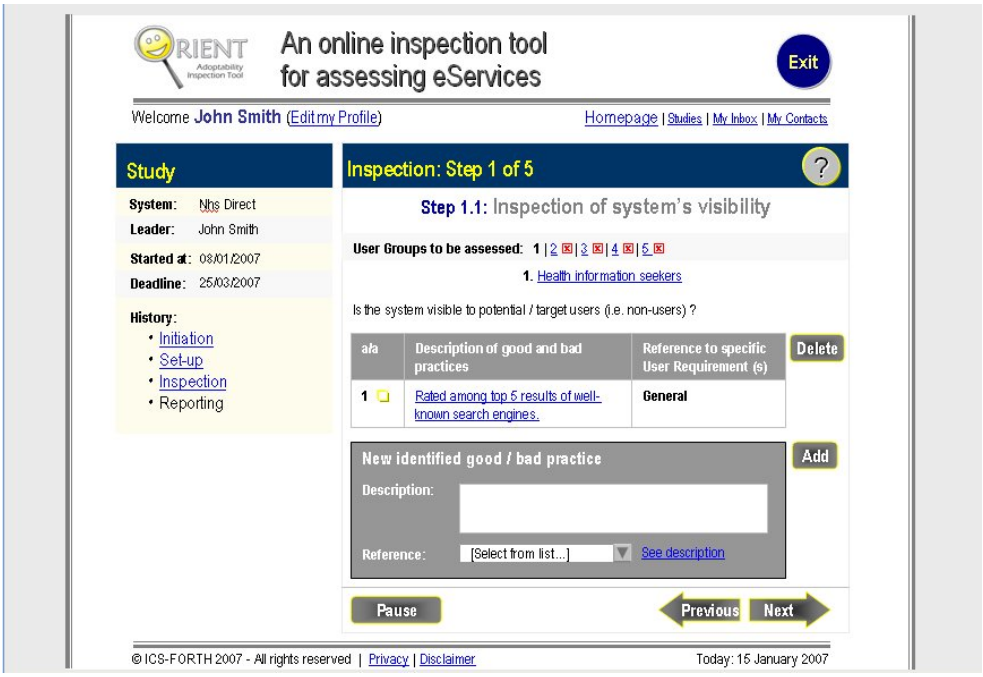

Figure 6.56: Inspection of system's visibility (original layout).

According to findings from the heuristic evaluation, the layout of this section (as well as of sections with similar layout, i.e., almost all steps of the inspection process) was shaped as depicted in figure 6.11.

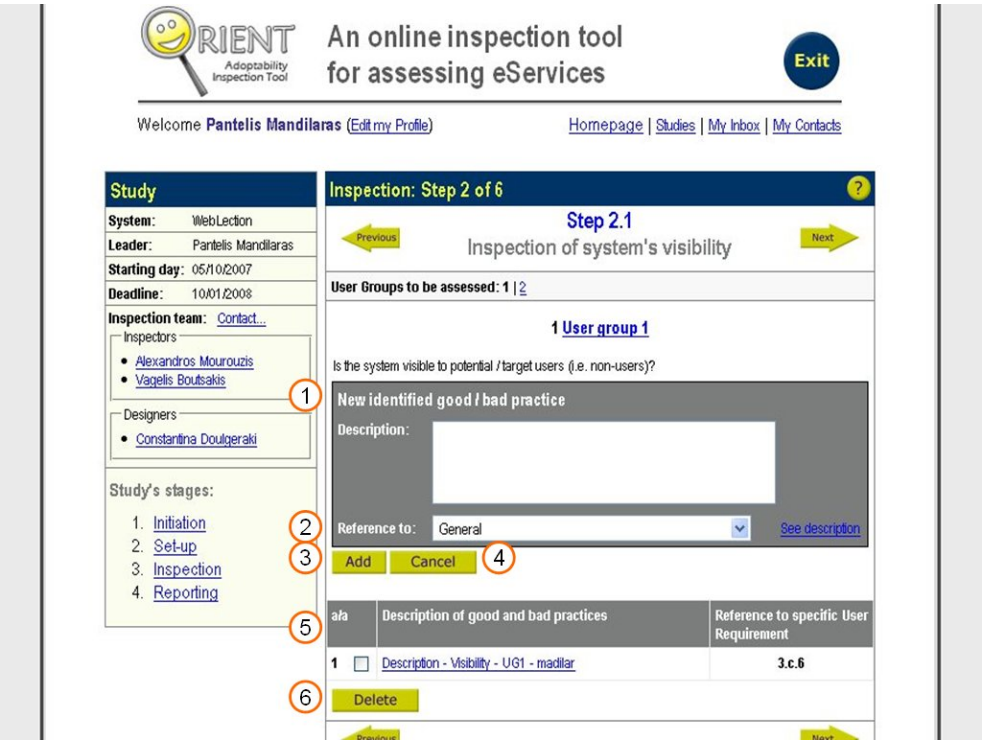

Figure 6.57: Inspection of system's visibility (redesigned layout).

The panel for documenting a new good / bad practice was moved to the top of the content section, as it represents the most usual function a user may perform in the specific page **(1)**. The label "reference" was changed to "reference to" which is more informative. The "reference" drop-down list was increased in size, offering more information to users about each requirement listed in the drop-down list. More over, the old default value of the drop-down list (which was "[Select from the list…]") was omitted as the guideline stated an obvious fact. The default value for the list in the redesigned version is "General" **(2)**.

The "add" button was moved at the bottom of the "new identified good / bad practice" panel with the intention to follow the course of actions of the user when interacting with the panel (i.e. the user uses the "description" field first, then scrolls down to the "reference" field and eventually finds the "add" button, instead of searching for it at the top right corner of the panel, which was its prior location in the original layout) **(3)**. When the user has chosen to edit a documented practice or has began describing a new one and wants to abort the process, he / she is in need of a button that performs that function. That is the function of the "cancel" button that was added in the redesigned version of such screens **(4)**.

The table of previously documented good / bad practices is presented at the bottom of the "new identified good / bad practice" **(5)**. The reason for this is twofold: first its significance is less, as the user will most likely add several practices at every stage of the inspection process and may never need to review them. The second reason is that the "new identified good / bad practice" panel remains constant in size and will never make the table of practices "move" off the visible area of the screen.

Finally, the "delete" button was moved under the checkboxes in the table, thus providing the user with a visual clue that the two are related **(6)**.

Section: "Inspection of functions' user-experience"

The original layout of this section is presented in figure 6.12.

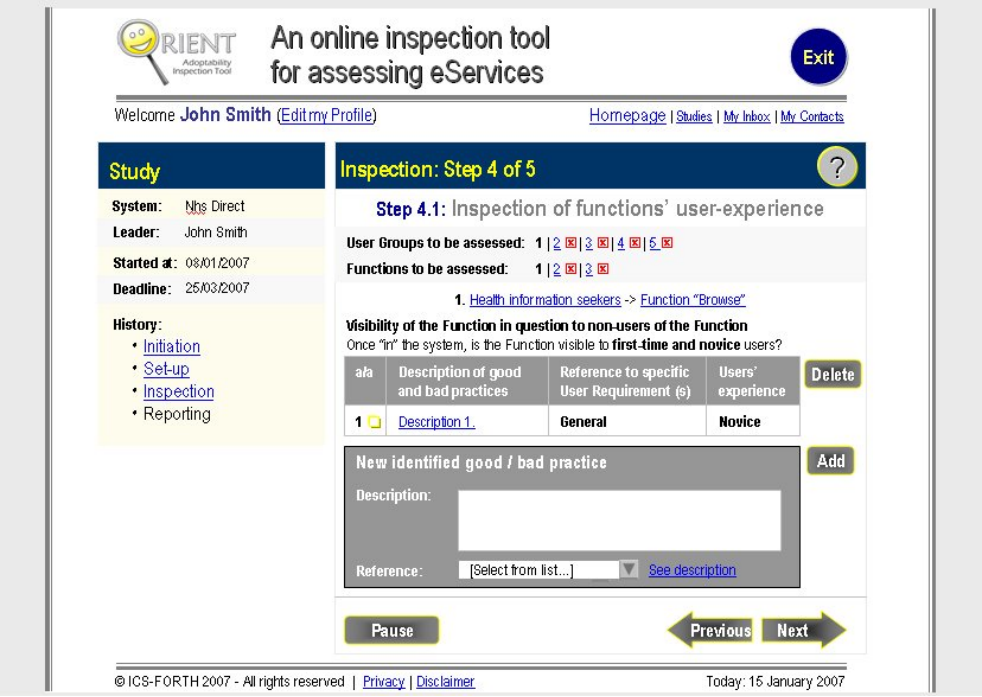

Figure 6.58: Inspection of functions' user-experience (original layout).

As evident from the heuristic evaluation, it is very likely that users will find it difficult to navigate between the different functions of a user group during the inspection of the functions' user-experience. This difficulty does not reside in the actual use of the hyperlinks, but rather in the way these are presented to the user, which according to findings is ambiguous and misleading. The proposed solution for this problem is presented in figure 6.13.

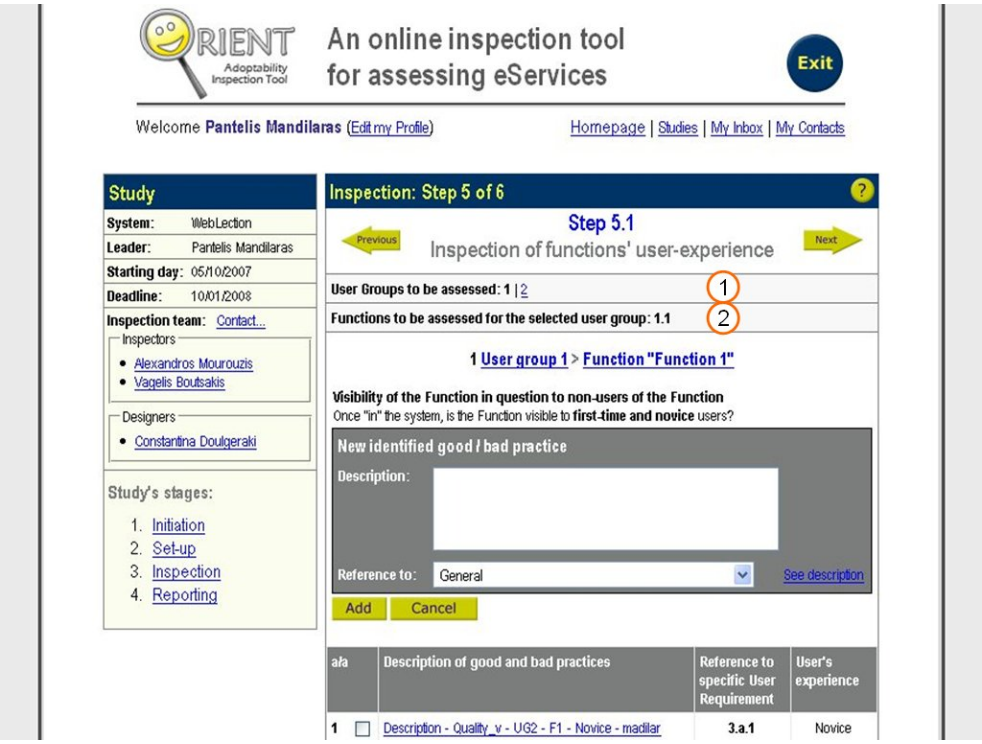

Figure 6.59: Inspection of functions' user-experience (redesigned layout).

Functions for a specific user group are no longer listed with the use of a simple number. Instead, they are listed with the use of a set of numbers in the form of 1.3 (number dot number) **(2)**. The first digit represents the user group to which the specific function is related **(1)**. The visual clue is enhanced by the fact that the selected user group is highlighted, thus it is very likely that the user will make the connection between the functions and the selected user group.

Section: "Reporting of system's visibility"

The original layout of this section is presented in figure 6.14.

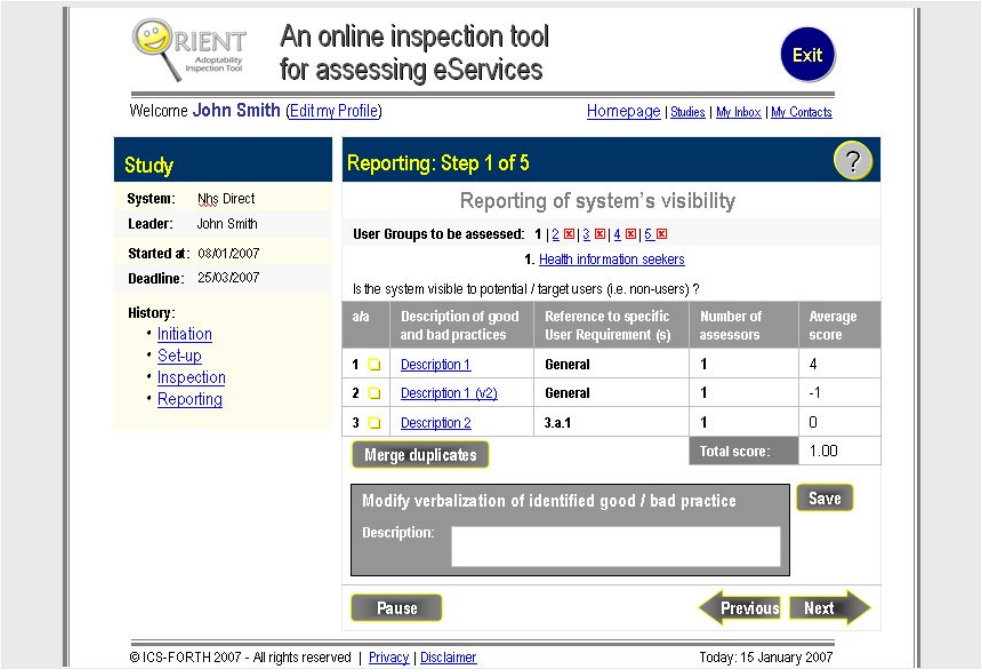

Figure 6.60: Reporting of system's visibility (original layout).

Offering the user the choice to "merge" and "modify" practices at the same time, according to the findings of the heuristic evaluation, increases the possibility of an error occurring. The new layout for this section, as well as other sections of the reporting phase, separates the two distinct functions into different screens and is presented in figures 6.15 and 6.16.

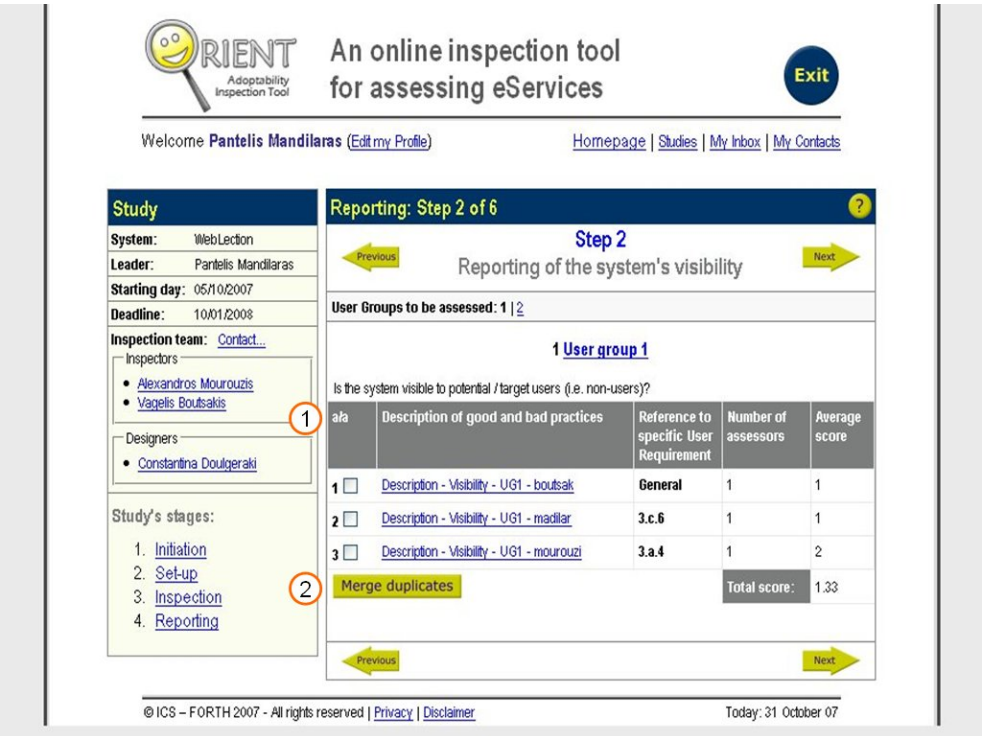

Figure 6.61: Reporting of system's visibility (part a) (redesigned layout).

The summative presentation of all documented practices by the inspection team is presented in a table by itself **(1)**. The "merge duplicates" button is situated directly below the checkboxes, thus providing a visual clue for the relationship between the two **(2)**.

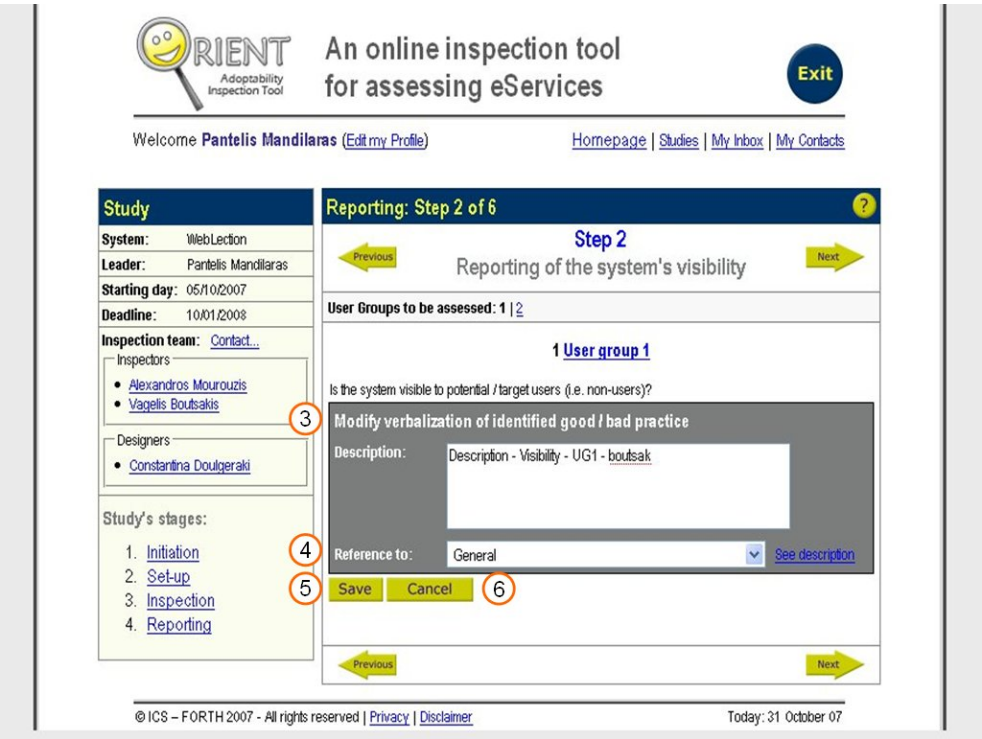

Figure 6.62: Reporting of system's visibility (part b) (redesigned layout).

When the user selects a practice from the table shown in figure 6.15, they are transferred to the screen shown in figure 6.16, where they may modify the verbalization of the practice, with no risk of getting side-tracked by other functions present in the same screen **(3)**. Besides the ability to modify the description of the practice, the ability to modify the reference of the practice to a specific requirement was also added, as practices that are merged may not always have common references and the user may need to re-define a suitable reference for the merged practice **(4)**. The "save" button was moved at the bottom of the "modify" panel to match the movement of the user while interacting with this function (i.e. the user scrolls down the panel and fills-in information **(5)**. Therefore, it is more logical to place the "save" button at the bottom of the panel, instead of the top right corner of the panel where it was originally). Finally, a "cancel" button was added to enable the user to abort the modifying process and return to the previous screen **(6)**.

6.2 User testing

6.2.1 Preparation of the evaluation

For the user testing part of the evaluation of the ORIENT inspection tool, a prototype version of the EDeAN web portal (figure 6.17) was used [9]. Four users were selected to act as the inspection team to whom the task of inspecting the EDeAN portal was assigned.

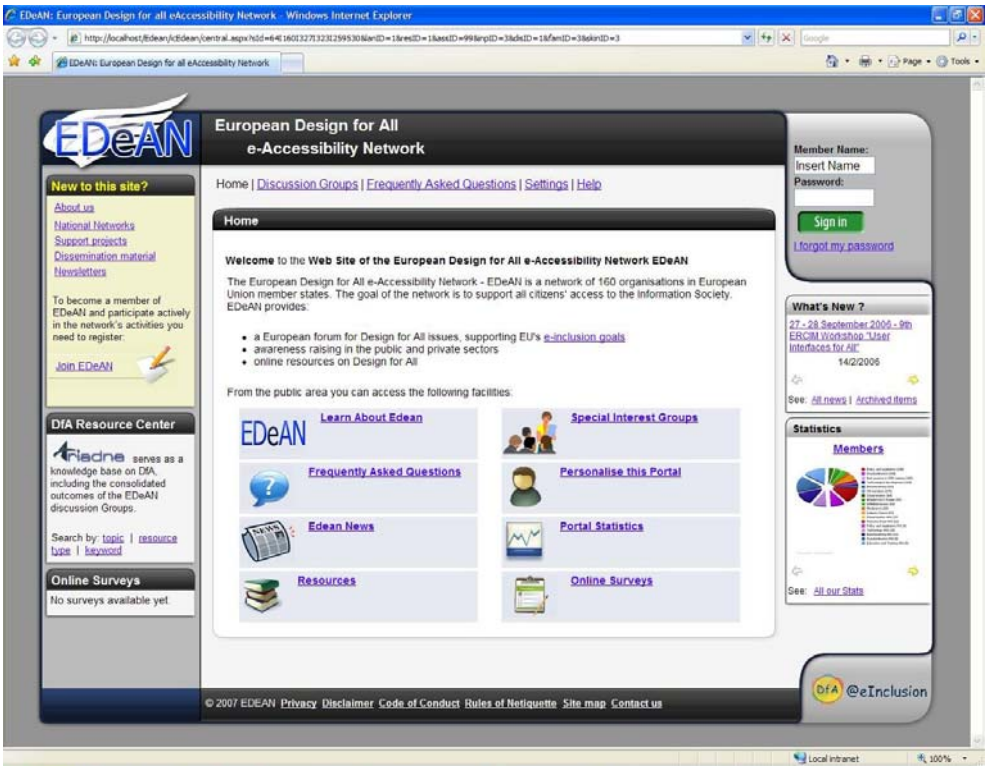

Figure 6.63: Prototype version of the EDeAN web portal that served as the system under assessment in the user testing of ORIENT.

Users had varying experience with the portal, as well as with the ORIENT inspection tool, as shown in table 6.2. However, all users had experience in the field of evaluations and an advanced understanding of all the terms and concepts involved.

Table 72: The level of experience of the four test users with a. the EDEAN portal and b. the ORIENT inspection tool.

User reference number	Relation to EDEAN	Relation to ORIENT
1	Moderately experienced with the portal.	Very experienced with the inspection tool.
2	Very experienced with the portal.	Novice with the inspection tool.

3	Novice with the portal.	Novice with the inspection tool.
4	Moderately experienced with the portal.	Moderately experienced with the inspection tool.

The most experienced user with the ORIENT inspection tool was selected to act as the inspection leader of the inspection team. In collaboration with the EDEAN portal's provider, he established the background of the study. In more details, he provided information about the study, such as the period of assessment, the objectives of the study and the type of expected results. Furthermore, he documented the inspection team's composition, assigning roles to each of the users involved in the user-testing. The EDEAN portal's provider was assigned the role of designer, as it involves no actual responsibilities regarding the inspection and enables that person to observe the inspection process for the entire duration of the study.

An informative leaflet was prepared for every user / member of the inspection team with instructions on how to reach / access the EDEAN portal, as well as extensive descriptions of the user profiles (blind, colour blind and motor impaired users with expertise varying from novice to expert) they would be asked to use and the functions they would perform during the evaluation. A representative sample of three functions was selected to act as the test scenario for the evaluation. These included:

- The user to visit the Resource Center of the EDeAn Portal and view a resource's information.
- The user to post a message with an attachment in a discussion group of his / her choice.
- The user to change several of his profile settings.

Finally, an adaptation (Appendix: User-evaluation questionnaire for the ORIENT online inspection tool) of the proposed questionnaire from [28] was prepared with the intent to be handed out to users at the end of the evaluation in order to record their experience with using the ORIENT inspection tool. The questionnaire contained ten groups of questions, which were visual clarity, consistency, compatibility, informative feedback, explicitness, appropriate functionality, flexibility and control, error prevention and correction, user guidance and support and system usability problems. Users were instructed to answer each question on a scale from 1 to 4, with 1 being *never*, 2: *some of the time*, 3: *most of the time* and 4: *always*. Users were also encouraged to write down comments, clarifying their answers whenever they felt it was necessary.

6.2.2 Evaluation

Prior to the actual inspection of the EDEAN portal for the purpose of evaluating the ORIENT inspection tool, users received a brief introduction course to the user-experience evaluation framework, which ORIENT implements.

Each user went through the test scenario, completing all steps. Whenever a user would require help, questions were addressed at the portal's developer. After the users had completed the test scenario, they began the actual evaluation of the portal by means of the ORIENT inspection tool. Using ORIENT, users documented practices of good / bad design that they identified while using the EDEAN portal, following the step-by-step process dictated by the tool. At the end of the inspection phase of the study, the user who was assigned the role of inspection leader proceeded with the reporting phase of the study. The entire duration of the study was four days.

When the study ended, all four participant users were debriefed by the inspection tool's developer and handed a copy of the questionnaire to fill out.

6.2.3 Results of the evaluation

After all users had completed their questionnaires, average values for every question were calculated and documented. In this section, average scores for each section, as well as the comments that users included in their answers, will be presented.

Section 1: Visual clarity

Visual clarity represents whether information is presented in a clear and well-organized way on the screen. Average scores concerning the visual clarity of the ORIENT inspection tool are presented in table 6.3.

Table 73: Average scores concerning the visual clarity of the ORIENT inspection tool.

Question	Rating
1. Is each screen clearly identified with an informative title or description?	3.75
2. When the user enters information on the screen is it clear where and in what format the information should be entered?	4
3. Does information appear to be organized logically on the screen? (e.g. menus organized by probable sequence of selection, or alphabetically)	3.75
4. Are different types of information clearly	4

separated from each other on the screen? (e.g. instructions, control options, data displays)	
5. Where a large amount of information is displayed on the screen, is it clearly separated into sections on the screen?	4
6. Are bright or light colours displayed on a dark background and vice versa?	4
6. Is the information on the screen easy to see and read?	3.5
8. Is it easy to find the required information on a screen?	3.5

Users were pleased with the visual clarity of the ORIENT inspection tool. Each screen of the system is identified by an informative title, which appears both in the title bar of the browser and in the main content area. Extensive use of separators, placeholders and lines with alternating background colours help visually separating different pieces of information.

On the other hand, font size is marginally large enough for users to read. User no2 believed that the presentation of studies in tabs should affect the general heading "Studies" of the section in accordance with the active tab. In certain parts of the tool, hyperlinks (which are coloured in blue) are presented onto a dark grey background, which does not provide sufficient contrast for readability. Finally, user no4 felt that the menu hierarchy and layout would have been more efficient if it appeared constantly on the left of the content of each page and was organized in a tree hierarchy with each level expanding only when selected.

Section 2: Consistency

Evaluating consistency aims to ensure that the way the inspection tool looks and functions should be the same at all times. Average scores concerning the consistency of the ORIENT inspection tool are presented in table 6.4

Table 74: Average scores concerning the consistency of the ORIENT inspection tool.

Question	Rating
1. Are icons, symbols, graphical representations and other pictorial information used consistently throughout the system?	4
2. Is the same type of information (e.g. instructions, menus, messages, titles, etc.)	3.75

displayed in the same location and layout on the screen?	
3. Is the same item of information displayed in the same format, wherever it appears?	4
4. Is the format in which the user should enter particular types of information on the screen consistent throughout the system?	3.75
5. Is the method of entering information consistent throughout the system?	4
6. Are there standard procedures for carrying out similar, related operations? (e.g. updating and deleting information)	4

Users were very satisfied with how consistent the inspection tool is. There is a common way of presentation of information throughout the tool and functions behave the same in similar sections.

Section 3: Compatibility

Table 6.5 illustrates the users' take on ORIENT's compatibility, meaning whether the way the system looks and works is compatible with user conventions and expectations.

Table 75: Average scores concerning the compatibility of the ORIENT inspection tool.

Question	Rating
1. Where icons, symbols, graphical representations and other pictorial information are displayed are they easy to recognize and understand and do they follow conventions where these exist?	3.75
2. Are established conventions followed by the format in which particular types of information are displayed? (e.g. layout of dates and telephone numbers)	3.75
3. Are control actions compatible with those used in other systems with which the user may need to interact?	3.75
4. Is information presented in a way which fits	3.5

the user's view of the task?	
5. Are graphical displays compatible with the user's view of what they are representing?	4
6. Does the organization and structure of the system fit the user's perception of the task?	3.5
6. Does the sequence of activities required to complete a task follow what the user would expect?	3.5
8. Does the system work in the way the user thinks it should work?	3.5

On the whole, users found the icons and symbols used in ORIENT easy to recognize and understand. Sole exception was the small red *x* icon, used to mark user groups and functions that the user has not yet assessed. Most of the users thought that it would be more natural to be able to provide grades to each practice when they document the practice in the first place and not in a separate screen. The explanation of the significance of the colour coding of cells used in the overview tables could have been presented in a compact and organized form, according to user no2. User no3 felt that the difference in meaning between the words "initiation" and "set-up" is slight and that users could get confused over it. Finally, user no4 feels that the sequence of activities required to complete a task is pretty unclear and states that he got confused.

Section 4: Informative feedback

Table 6.6 reflects on the clarity and how much informative is the system's feedback to users concerning their current location, what they have done so far and if these actions were successful and where they should go from that point.

Table 76: Average scores regarding how informative is the feedback provided to users by the ORIENT inspection tool.

Question	Rating
1. Are instructions and messages displayed by the system concise and positive?	4
2. Do instructions and prompts clearly indicate what to do?	4
3. Is it clear what actions the user can take at any stage?	4
4. Is it clear what the user needs to do in order to	3.5

take a particular action? (e.g. which options to select, which keys to press, etc.)	
5. When the user enters information on the screen, is it made clear what this information should be?	4
6. Do error messages explain clearly where and what the errors are and why they have occurred?	4
7. Is it clear to the user what should be done to correct an error?	4

Users felt that the inspection tool provided sufficient feedback about their current location. They also believe that the system adequately informed them about actions they had already taken and what should be the next steps for each stage.

Section 5: Explicitness

Average scores concerning the inspection tool's explicitness, measuring if the way the system works and is structured is clear to the user, is presented in table 6.7.

Table 77: Average scores regarding the explicitness of the ORIENT inspection tool.

Question	Rating
1. Is it clear what stage the system has reached in a task?	3.5
2. Is it clear what the user needs to do in order to complete a task?	3.5
3. Where the user is presented with a list of options (e.g. in a menu), is it clear what each option means?	3.75
4. Is it clear what part of the system the user is in?	3.25
5. Is it clear how, where and why changes in one part of the system affect other parts of the system?	3.5
6. Is it clear why a series of screens are sequenced as they are?	4
7. Is the system well-organized from the user's point of view?	3.75

As apparent from the average scores, users deemed that the inspection tool's explicitness was lacking at certain points. For instance, a study's workspace, though helpful as a concept and a way to organize a study and provide access to all parts of it, does not belong in any of the navigation hierarchies stemming from the options in the main navigation menu of the inspection tool. User no4 felt that on the whole the system was not structured in a clear way and found that often it was not clear to him what stage he currently was in.

Section 6: Appropriate functionality

Table 6.8 reflects on ORIENT's appropriate functionality (i.e., whether the system meets the needs and requirements of users when carrying out tasks).

Table 78: Average scores regarding the appropriate functionality of the ORIENT inspection tool.

Question	Rating
1. Is the way in which information is presented appropriate for the tasks?	3.5
2. Does each screen contain all the information which the user feels is relevant to the task?	3.75
3. Can users access all the information which they feel they need for their current task?	3.75
4. Do the contents of help and tutorial facilities make use of realistic task data and problems?	4
5. Where task sequences are particularly long, are the broken into appropriate subsequences? (e.g. separating a lengthy editing procedure into its consistent parts)	4

The majority of users felt that the way users groups and functions are presented as navigation aids during the inspection process is not very clear. The presentation of the user group's title as a hyperlink would be clearer, as they suggested. User no2 remarked that the information a user may need while carrying out a task is not always available on the respective screen. However, as he stated, it is available on other parts of the inspection tool, accessible to the user through hyperlinks present on those very screens.

Section 7: Flexibility and control

Evaluating flexibility and control aims to ensure that the interface is sufficiently flexible in structure and in the way information is presented to the user, thus allowing them to feel in

control of the system. Average scores on flexibility and control of the ORIENT inspection tool are presented in table 6.9.

Table 79: Average scores regarding the flexibility and control of the ORIENT inspection tool.

Question	Rating
1. Is there an easy way for the user to 'undo' an action and step back to a previous stage or screen? (e.g. if the user makes a wrong choice)	3.75
2. Can the user look through a sequence of actions in either direction?	4
3. Can the user access a particular screen in a sequence of screens directly? (e.g. where a list or table covers several screens)	2.75
4. In menu-based systems, is it easy to return to the main menu from any part of the system?	4
5. Can the user move to different parts of the system as required?	3.75
6. Does the system prefill repeated information on the screen, where possible? (e.g. to save the user having to enter the same information several times)	2.75
7. Can the user override computer-generated (e.g. default) information, if appropriate?	3.75

The majority of users reported that although the ability to 'undo' an action is not presented to the user, similar results can be achieved by 'stepping back'. Besides this, users remarked that there is no direct way of accessing sub-steps of the different stages of the inspection process (e.g., when the user is in the inspection of the functions' user-experience step, they have no way of moving directly from sub-step 1 to sub-step 12). User no2 suggested that the "reference to" field used in the documentation of practices could remain the same as the one used the previous time. However, as references to requirements are most likely to change from practice to practice, this suggestion may not prove so helpful. Finally, user no1 stated that users are not able to change average scores that the system calculates in the reporting phase from individual scores provided by the members of the inspection team. However, since average scores are produced based on a mathematical equation for the calculation of an average from individual values, there is no point in allowing the user to tamper with these results.

Section 8: Error prevention and correction

Error prevention and correction examines if the possibility of user error is minimum. Average scores for this section are presented in table 6.10.

Table 80: Average scores regarding the error prevention and correction of the ORIENT inspection tool.

Question	Rating
1. Does the system validate user inputs before processing, wherever possible?	3.75
2. Does the system clearly and promptly inform the user when it detects an error?	3.75
3. Are users able to check what they have entered before it is processed?	4
4. It the system protected against common trivial errors?	3.75
5. Does the system prevent users from taking actions which they are not authorized to take? (e.g. by requiring passwords, hiding functions which some users are not authorized to use from them, etc.)	4

User no2 found the use of a virtual calendar for date inputting a very good addition, as it minimizes the possibility of user error.

Section 9: User guidance and support

Table 6.11 illustrates how informative, easy-to-use and relevant the provided guidance and support was according to users of the ORIENT inspection tool.

Table 81: Average scores regarding the user guidance and support of the ORIENT inspection tool.

Question	Rating
If there is some form of help facility (or guidance) on the computer to help the user when using the system then:	
1. Can the user request this easily from any point in the system?	3.5
2. Is it clear how to get in and out of the help facility?	4
3. Is the help information presented clearly, without	

interfering with the user's current activity?	4
4. When the user requests help, does the system clearly explain the possible actions which can be taken, in the context of what the user is currently doing?	4
5. When using the help facility, can the user find relevant information directly, without having to look through unnecessary information?	4
6. Does the help facility allow the user to browse through information about other parts of the system?	4
7. Is the organization of all forms of user guidance and support related to the tasks which the user can carry out?	3.75

Users found the help function very well organized. As they stated, help was provided in a case-sensitive as well on a more generic context. Screenshots with numbered steps were very helpful and helped users quickly and easily find what they were looking for. Finally, the presentation of help on a separate new window provided the necessary assistance without interfering with the user's main activity at the time.

Section 10: System usability problems

Table 6.12 summarizes the occurrence of usability problems by the users while interacting with the ORIENT inspection tool.

Table 82: Average scores regarding system usability problems the users encountered while interacting with the ORIENT inspection tool.

Question	Rating
1. Working out how to use the system	1.75
2. Lack of guidance on how to use the system	1.25
3. Understanding how to carry out the tasks	1.75
4. Knowing what to do next	1.5
5. Understanding how the information on the screen relates to what you are doing	1.5
6. Finding the information you want	1
6. Colours which are difficult to look at for any length of time	1
8. An inflexible HELP (guidance) facility	1
9. Losing track of where you are in the system or of what you are doing or have done	1.5

10. Having to remember too much information while carrying out a task	1
11. System response times are too slow	1
12. Knowing where or how to input information	1
13. Having to be very careful in order to avoid errors	1

Users encountered a few problems working out how to use the system, due to lack of knowledge of the user-experience evaluation framework. Besides this, as previously mentioned, users found that the presentation of user groups and functions as a navigation aid during the inspection process is not very clear.

A summative presentation of the quantitative results of the user-testing evaluation of the ORIENT inspection tool is presented in figure 6.18.

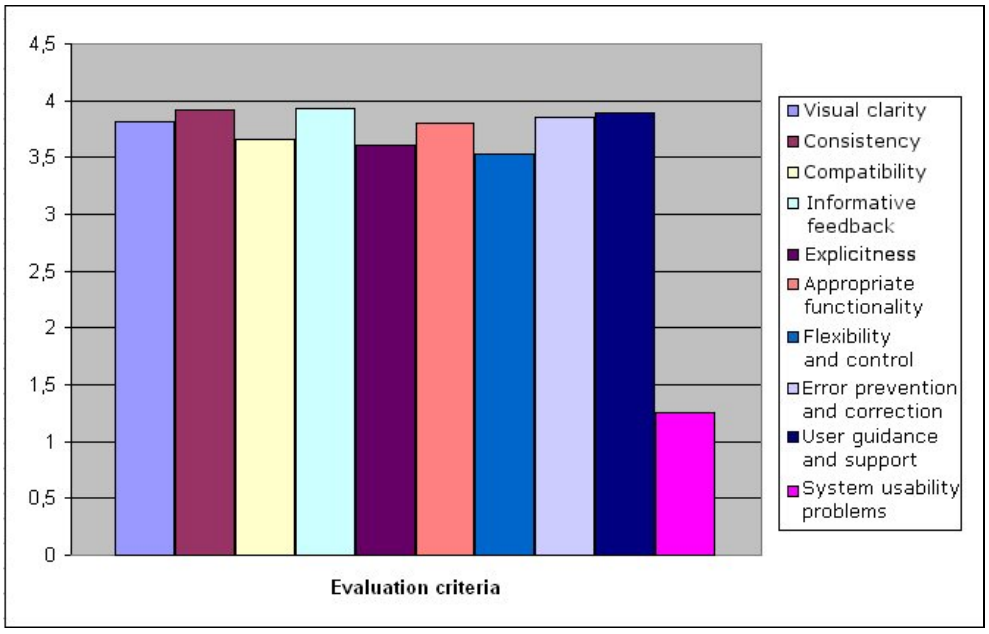

Figure 6.64: Overall quantitative results of the user-testing evaluation of ORIENT.

6.3 Conclusions

The first part of the evaluation process for ORIENT, namely the heuristic evaluation, identified a large number of problems that would affect greatly the usability of the inspection tool. Findings from the heuristic evaluation served as input in a new design session for the entire inspection tool, which resulted in important changes in the tool's structure and presentation, as well as the way users interact with it.

The second part of the evaluation process (user testing) proved the new design approach right, as ORIENT received significantly higher scores in comparison to those from

the heuristic evaluation. Although users identified a few extra problems that the expert evaluators had overlooked, on the whole they were satisfied with the inspection tool.

The evaluation process of ORIENT (as a whole) resulted in a more usable implementation of the inspection tool. Although the majority of users involved in the user testing phase of the evaluation had little or no experience with using the inspection tool, they were able to figure out quickly and easily how to use it (i.e., the tool demonstrates an ease of learning).

7 Conclusions and future work

The aim of this thesis was to present the design, implementation and evaluation of the web-based version of ORIENT, an inspection tool for the evaluation of User-Experience of online services.

The web-based version of the tool improves significantly the ease of use of the inspection tool, as well as its visibility and availability. Several parts of the inspection process are automated or permit the repeated use of older information (reusability) greatly enhancing the speed of the assessment. The inspection tool was redesigned and re-implemented according to findings from the evaluation process making it more usable. Finally, the tool demonstrates a satisfying ease of learning, as it was observed that users with little or no experience with using the tool were able to figure out quickly and easily how to use it.

The web-based version of the ORIENT inspection tool should make it more available and appealing to system providers in search of a holistic approach to usability and accessibility evaluation. Additionally, the web-based version of ORIENT includes communication features supporting collaboration in the context of evaluation cases, and evaluation practice in general.

Future work may include the redesign of the inspection tool to make it accessible to people with disabilities. Furthermore, findings of a comparative study between ORIENT and other inspection tools would highlight weaknesses of ORIENT (so, that they could be remedied in future versions of the tool) and validate its strengths, through hard evidence.

The current implemented version will be made available online in order to test the inspection tool in real conditions of use. Moreover, when the inspection tool has been used to perform a number of assessments, it will be possible to collect and process findings from all the studies with the intention of creating a set of guidelines for the development of usable and accessible systems.

Appendix A: User-evaluation questionnaire for the ORIENT online inspection tool

This questionnaire is designed to reflect your experience with using the ORIENT online inspection tool for the purposes of evaluating the EDEAN portal.

Brief instructions
The questions of this questionnaire are divided into 10 sections, each reflecting upon a specific attribute of the ORIENT tool. Answer each question with a number between 1 and 4 (4: *always*, 3: *most of the time*, 2: *some of the time*, 1: *never*).
If you feel some of your answers require further clarifying, please write your comments in the respective column and be as extensive in your commenting as you feel is necessary.
Remember, these answers reflect your personal views on the usability of the system and, as such, there are no right or wrong answers.

Section 1: Visual clarity

Information displayed on the screen should be clear, well-organized, unambiguous and easy to read.

Question	Rating	Comments
1. Is each screen clearly identified with an informative title or description?		
2. When the user enters information on the screen is it clear where and in what format the information should be entered?		
3. Does information appear to be organized logically on the screen? (e.g. menus organized by probable sequence of selection, or alphabetically)		
4. Are different types of information clearly separated from each other on the screen? (e.g. instructions, control options, data displays)		
5. Where a large amount of information is displayed on the screen, is it clearly separated into sections on the screen?		
6. Are bright or light colours displayed on a dark background and vice versa?		
7. Is the information on the screen easy to see and read?		
8. Is it easy to find the required information on a screen?		

Section 2: Consistency

The way the system looks and works should be consistent at all times.

Question	Rating	Comments
1. Are icons, symbols, graphical representations and other pictorial information used consistently throughout the system?		
2. Is the same type of information (e.g. instructions, menus, messages, titles, etc.) displayed in the same location and layout on the screen?		
3. Is the same item of information displayed in the same format, wherever it appears?		
4. Is the format in which the user should enter particular types of information on the screen consistent throughout the system?		
5. Is the method of entering information consistent throughout the system?		
6. Are there standard procedures for carrying out similar, related operations? (e.g. updating and deleting information)		

Section 3: Compatibility

The way the system looks and works should be compatible with user conventions and expectations.

Question	Rating	Comments
1. Where icons, symbols, graphical representations and other pictorial information are displayed are they easy to recognize and understand and do they follow conventions where these exist?		
2. Are established conventions followed by the format in which particular types of information are displayed? (e.g. layout of dates and telephone numbers)		
3. Are control actions compatible with those used in other systems with which the user may need to interact?		
4. Is information presented in a way which fits the user's view of the task?		
5. Are graphical displays compatible with the user's view of what they are representing?		
6. Does the organization and structure of the system fit the user's perception of the task?		

7. Does the sequence of activities required to complete a task follow what the user would expect?		
8. Does the system work in the way the user thinks it should work?		

Section 4: Informative feedback

Users should be given clear, informative feedback on where they are in the system, what actions they have taken, whether these actions have been successful and what actions should be taken next.

Question	Rating	Comments
1. Are instructions and messages displayed by the system concise and positive?		
2. Do instructions and prompts clearly indicate what to do?		
3. Is it clear what actions the user can take at any stage?		
4. Is it clear what the user needs to do in order to take a particular action? (e.g. which options to select, which keys to press, etc.)		
5. When the user enters information on the screen, is it made clear what this information should be?		
6. Do error messages explain clearly where and what the errors are and why they have occurred?		
7. Is it clear to the user what should be done to correct an error?		

Section 5: Explicitness

The way the system works and is structured should be clear to the user.

Question	Rating	Comments
1. Is it clear what stage the system has reached in a task?		
2. Is it clear what the user needs to do in order to complete a task?		
3. Where the user is presented with a list of options (e.g. in a menu), is it clear what each option means?		
4. Is it clear what part of the system the user is in?		
5. Is it clear how, where and why changes in one part of the system affect other parts of		

the system?		
6. Is it clear why a series of screens are sequenced as they are?		
7. Is the system well-organized from the user's point of view?		

Section 6: Appropriate functionality

The system should meet the needs and requirements of users when carrying out tasks.

Question	Rating	Comments
1. Is the way in which information is presented appropriate for the tasks?		
2. Does each screen contain all the information which the user feels is relevant to the task?		
3. Can users access all the information which they feel they need for their current task?		
4. Do the contents of help and tutorial facilities make use of realistic task data and problems?		
5. Where task sequences are particularly long, are the broken into appropriate subsequences? (e.g. separating a lengthy editing procedure into its consistent parts)		

Section 7: Flexibility and control

The interface should be sufficiently flexible in structure, in the way information is presented and in terms of what the user can do, to suit the needs and requirements of all users, and to allow them to feel in control of the system.

Question	Rating	Comments
1. Is there an easy way for the user to 'undo' an action and step back to a previous stage or screen? (e.g. if the user makes a wrong choice)		
2. Can the user look through a sequence of actions in either direction?		
3. Can the user access a particular screen in a sequence of screens directly? (e.g. where a list or table covers several screens)		
4. In menu-based systems, is it easy to return to the main menu from any part of the system?		
5. Can the user move to different parts of the system as required?		

6. Does the system prefill repeated information on the screen, where possible? (e.g. to save the user having to enter the same information several times)		
7. Can the user override computer-generated (e.g. default) information, if appropriate?		

Section 8: Error prevention and correction

The system should be designed to minimize the possibility of user error, with inbuilt facilities for detecting and handling those which do occur; users should be able to check their inputs and to correct errors or potential error situations before the input is processed.

Question	Rating	Comments
1. Does the system validate user inputs before processing, wherever possible?		
2. Does the system clearly and promptly inform the user when it detects an error?		
3. Are users able to check what they have entered before it is processed?		
4. It the system protected against common trivial errors?		
5. Does the system prevent users from taking actions which they are not authorized to take? (e.g. by requiring passwords, hiding functions which some users are not authorized to use from them, etc.)		

Section 9: User guidance and support

Informative, easy-to-use and relevant guidance and support should be provided to help the user understand and use the system.

Question	Rating	Comments
If there is some form of help facility (or guidance) on the computer to help the user when using the system then: **7.** Can the user request this easily from any point in the system? **8.** Is it clear how to get in and out of the help facility? **9.** Is the help information presented clearly, without interfering with the user's current activity? **10.** When the user requests help, does		

the system clearly explain the possible actions which can be taken, in the context of what the user is currently doing? **11.** When using the help facility, can the user find relevant information directly, without having to look through unnecessary information? **12.** Does the help facility allow the user to browse through information about other parts of the system?		
7. Is the organization of all forms of user guidance and support related to the tasks which the user can carry out?		

Section 10: System usability problems

When using the system, did you experience problems with any of the following:

Question	Rating	Comments
1. Working out how to use the system		
2. Lack of guidance on how to use the system		
3. Understanding how to carry out the tasks		
4. Knowing what to do next		
5. Understanding how the information on the screen relates to what you are doing		
6. Finding the information you want		
7. Colours which are difficult to look at for any length of time		
8. An inflexible HELP (guidance) facility		
9. Losing track of where you are in the system or of what you are doing or have done		
10. Having to remember too much information while carrying out a task		
11. System response times are too slow		
12. Knowing where or how to input information		
13. Having to be very careful in order to avoid errors		

Appendix B: Definition of ORIENT's database tables

Table B.1: Table for the entity Inspection

Inspection			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	The name of the system under assessment.
Date_from	**Datetime**	**Descriptor**	The starting date of the study.
Date_to	**Datetime**	**Descriptor**	The deadline of the study.
Objectives	**Alphanumeric (255)**	**Descriptor**	The objectives of the study.
Type	**Alphanumeric (255)**	**Descriptor**	The type of expected results for the study.
Recipients	**Alphanumeric (255)**	**Descriptor**	The recipients of the final report of the study.
Progress	**Integer**	**Descriptor**	The current stage of the study.
Leader	**Alphanumeric (15)**	**Descriptor**	The inspection leader.
Date_to_setup	**Datetime**	**Descriptor**	Internal deadline for the end of the set-up phase.
Date_to_inspection	**Datetime**	**Descriptor**	Internal deadline for the end of the inspection phase.
Status	**Boolean**	**Descriptor**	Indication about the publication status of the study.
Date_created	**Datetime**	**Descriptor**	Date the specific record was created.

Table B.2: Table for the entity Inspection team member

Inspection team member			
Field	**Type (size)**	**Key**	**Description**
username	**Alphanumeric (15)**	**Primary**	Username of the member.

userPassword	**Alphanumeric (32)**	**Descriptor**	Password of the member.
userMail	**Alphanumeric (30)**	**Descriptor**	E-mail of the member.
firstName	**Alphanumeric (15)**	**Descriptor**	First name of the member.
lastName	**Alphanumeric (20)**	**Descriptor**	Last name of the member.
Language	**Alphanumeric (20)**	**Descriptor**	Native language of the member.
Sex	**Character**	**Descriptor**	Sex of the member.
Background	**Alphanumeric (255)**	**Descriptor**	Background and expertise of the member.
Familiarity	**Integer**	**Descriptor**	Familiarity of the member with the ORIENT inspection tool.
Expertise	**Integer**	**Descriptor**	Familiarity of the member with inspection methods and tools in general.
Fluency	**Integer**	**Descriptor**	Level of fluency with the English language.
Type	**Character**	**Descriptor**	Indication about the member being more experienced as an inspector or as a designer.

Table B.3: Table for the relationship Participate between an ORIENT member and an Inspection

Participate			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	Name of the system under assessment.
userName	**Alphanumeric (15)**	**Primary**	Username of the member.
Relation_system	**Alphanumeric (255)**	**Descriptor**	Relation of the member to the system under assessment.

Relation_provider	**Alphanumeric (255)**	**Descriptor**	Relation of the member to the provider of the system under assessment.
Familiarity_system	**Integer**	**Descriptor**	Familiarity of the member with the system under assessment.
Familiarity_similar	**Integer**	**Descriptor**	Familiarity of the member with similar to the system under assessment systems.
Familiarity_language	**Integer**	**Descriptor**	Familiarity of the member with the language supported by the system under assessment.
Other	**Alphanumeric (255)**	**Descriptor**	Other relevant information.
Role	**Integer**	**Primary**	Indication of the role assigned to the member.
Done	**Boolean**	**Descriptor**	Indication that the member has completed their assigned tasks.
Date_created	**Datetime**	**Descriptor**	Date the member was added to the inspection team.

Table B.4: Table for the entity Message

Message			
Field	**Type (size)**	**Key**	**Description**
Sender	**Alphanumeric (15)**	**Primary**	Sender of the message.
Subject	**Alphanumeric (50)**	**Descriptor**	Subject of the message.
Date	**Datetime**	**Primary**	Date the message was sent.
Message_text	**Alphanumeric (255)**	**Descriptor**	Body of the message.
Affiliation	**Character**	**Descriptor**	Affiliation of the sender to the receiver of the

			message.
Multiple_recipients	**Boolean**	**Descriptor**	Indication that the message was delivered to multiple recipients.
Sender_delete	**Boolean**	**Descriptor**	Indication that the sender of the message has deleted it from his/her "sent messages" folder.

Table B.5: Table for the relationship Recipient between an ORIENT member and a message

Message			
Field	**Type (size)**	**Key**	**Description**
Sender	**Alphanumeric (15)**	**Primary**	Sender of the message.
Date	**Datetime**	**Primary**	Date the message was sent.
Recipient	**Alphanumeric (15)**	**Primary**	Recipient of the message.
Type	**Character**	**Descriptor**	Indication whether the specific recipient is included in the "To:" or the "Cc:" list of the message.
Recipient_delete	**Boolean**	**Descriptor**	Indication that the specific recipient has deleted the specific message from their inbox.
Old	**Boolean**	**Descriptor**	Indication that the message has been opened by the specific recipient.

Table B.6: Table for the relationship Contact between two ORIENT members

Contact			
Field	**Type (size)**	**Key**	**Description**
member_ID	**Alphanumeric (15)**	**Primary**	Member – "owner" of the contacts' list.

contact_ID	**Alphanumeric (15)**	**Primary**	Member / contact in previous member's list.

Table B.7: Table for the relationship invitedContact between two ORIENT members

Contact			
Field	**Type (size)**	**Key**	**Description**
member_ID	**Alphanumeric (15)**	**Primary**	Member extending the invitation.
contact_ID	**Alphanumeric (15)**	**Primary**	Invited member.

Table B.8: Table for form 2 – System description

Form 2 – System description			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	Name of the system under assessment.
Access	**Alphanumeric (255)**	**Descriptor**	Location of the system under assessment (usually a url).
Provider	**Alphanumeric (50)**	**Descriptor**	Provider of the system under assessment.
Developer	**Alphanumeric (50)**	**Descriptor**	Developer of the system under assessment.
Platform	**Alphanumeric (255)**	**Descriptor**	Platform of the system under assessment.
Lifecycle	**Integer**	**Descriptor**	Current lifecycle stage of the system under assessment.
Objectives	**Text**	**Descriptor**	Objectives of the study.
Target	**Alphanumeric (255)**	**Descriptor**	Target user population of the system under assessment.
Application	**Alphanumeric (255)**	**Descriptor**	Application field(s) of the system under assessment.
Navigation	**Alphanumeric (255)**	**Descriptor**	Navigation styles supported by the system under assessment.

Networking	**Alphanumeric (255)**	**Descriptor**	Networking and communications supported by the system under assessment.
Mail	**Boolean**	**Descriptor**	Indication that e-mail is supported as a communication media.
Telephone	**Boolean**	**Descriptor**	Indication that telephone is supported as a communication media.
Postal	**Boolean**	**Descriptor**	Indication that postal correspondence is supported as a communication media.
Fax	**Boolean**	**Descriptor**	Indication that fax is supported as a communication media.
Other	**Alphanumeric (50)**	**Descriptor**	Other supported communication media.
Resources	**Alphanumeric (255)**	**Descriptor**	Resources of the system in terms of hardware, software and personnel of the provider.
Date_created	**Datetime**	**Descriptor**	Date the form was created.

Table B.9: Table for form 2 – Potential user group(s)

Form 2 – Potential user group(s)			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	Name of the system under assessment.
UG_ID	**Alphanumeric (50)**	**Primary**	Title of the user group.
Desc	**Text**	**Descriptor**	Description of the user group.
Assess	**Boolean**	**Descriptor**	Indication whether the specific user group will

			be assessed.
Date_created	**Datetime**	**Descriptor**	Date the entry was created.

Table B.10: Table for form 2a – System functions per user group

Form 2a – System functions per user group			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	Name of the system under assessment.
UG_ID	**Alphanumeric (50)**	**Primary**	Title of the user group.
Func_id	**Alphanumeric (50)**	**Primary**	Title of the function.
Type	**Integer**	**Descriptor**	Interactivity type of the function.
Assess	**Boolean**	**Descriptor**	Indication whether the specific function will be assessed.
Priority	**Integer**	**Descriptor**	Priority ordering of the function in relation to other functions of the same user group.
Actions	**Integer**	**Descriptor**	Number of actions describing the function.
Date_created	**Datetime**	**Descriptor**	Date the entry was created.

Table B.11: Table for form 2a – System actions for each function

Form 2 – System actions for each function			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	Name of the system under assessment.
UG_ID	**Alphanumeric (50)**	**Primary**	Title of the user group.
Func_id	**Alphanumeric (50)**	**Primary**	Title of the function.
Action	**Alphanumeric (255)**	**Primary**	Description of the action.
Sno	**Integer**	**Primary**	Serial number of the action.

Table B.12: Table for form 3 – Context of use per user group

Form 3 – Context of use per user group			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	Name of the system under assessment.
UG_ID	**Alphanumeric (50)**	**Primary**	Title of the user group.
Description	**Text**	**Primary**	Title of the context of use entry.
Assess	**Boolean**	**Descriptor**	Indication whether the specific context of use entry will be assessed.
Category	**Integer**	**Primary**	Category of the context of use entry.
Sno	**Integer**	**Descriptor**	Serial number of the context of use entry.
Date_created	**Datetime**	**Descriptor**	Date the entry was created.

Table B.13: Table for form 3 – Induced requirement(s) for context-of-use entries

Form 3 – Induced requirement(s) for context-of-use entries			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	Name of the system under assessment.
UG_ID	**Alphanumeric (50)**	**Primary**	Title of the user group.
Description	**Integer**	**Primary**	Context-of-use entry's serial number.
Requirement	**Text**	**Descriptor**	Description of the requirement.
Sno	**Integer**	**Primary**	Serial number of the requirement.

Table B.14: Table for saved context-of-use records

Saved context-of-use records			
Field	**Type (size)**	**Key**	**Description**
Title	**Alphanumeric (50)**	**Primary**	Name of the saved record.
Leader	**Alphanumeric (15)**	**Primary**	Member / owner of the

			record.
Description	**Text**	**Primary**	Title of the context of use entry.
Assess	**Boolean**	**Descriptor**	Indication whether the specific context of use entry will be assessed.
Category	**Integer**	**Primary**	Category of the context of use entry.
Sno	**Integer**	**Descriptor**	Serial number of the context of use entry.

Table B.15: Table for saved induced requirement(s) for saved context-of-use records

Saved induced requirement(s) for saved context-of-use records			
Field	**Type (size)**	**Key**	**Description**
Title	**Alphanumeric (50)**	**Primary**	Name of the saved record.
Leader	**Alphanumeric (15)**	**Primary**	Member / owner of the record.
Description	**Integer**	**Primary**	Context-of-use entry's serial number.
Requirement	**Text**	**Descriptor**	Description of the requirement.
Sno	**Integer**	**Primary**	Serial number of the requirement.

Table B.16: Table for form 4a – Visibility (per inspector)

Form 4a – Visibility (per inspector)			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	Name of the system under assessment.
username	**Alphanumeric (15)**	**Primary**	Member/inspector documenting the practice.
UG_ID	**Alphanumeric (50)**	**Primary**	Title of the user group.
Sno	**Integer**	**Descriptor**	Serial number of the documented practice.
Description	**Alphanumeric (255)**	**Primary**	Description of the

			documented practice.
Reference	**Alphanumeric (50)**	**Descriptor**	Reference to user requirement.
Score	**Integer**	**Descriptor**	Severity rating of the practice.
Date_created	**Datetime**	**Descriptor**	Date the entry was created.

Table B.17: Table for form 5a – Perceived usefulness & ease of use (per inspector)

Form 5a – Perceived usefulness & ease of use (per inspector)			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	Name of the system under assessment.
username	**Alphanumeric (15)**	**Primary**	Member/inspector documenting the practice.
UG_ID	**Alphanumeric (50)**	**Primary**	Title of the user group.
Sno	**Integer**	**Descriptor**	Serial number of the documented practice.
Description	**Alphanumeric (255)**	**Primary**	Description of the documented practice.
Reference	**Alphanumeric (50)**	**Descriptor**	Reference to user requirement.
Score	**Integer**	**Descriptor**	Severity rating of the practice.
Date_created	**Datetime**	**Descriptor**	Date the entry was created.

Table B.18: Table for form 6a – Availability & approachability (per inspector)

Form 6a – Availability & approachability (per inspector)			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	Name of the system under assessment.
Username	**Alphanumeric (15)**	**Primary**	Member/inspector documenting the practice.

UG_ID	**Alphanumeric (50)**	**Primary**	Title of the user group.
Sno	**Integer**	**Descriptor**	Serial number of the documented practice.
Description	**Alphanumeric (255)**	**Primary**	Description of the documented practice.
Reference	**Alphanumeric (50)**	**Descriptor**	Reference to user requirement.
Score	**Integer**	**Descriptor**	Severity rating of the practice.
Experience	**Integer**	**Primary**	Indication of the experience of users to whom the specific practice refers to.
Date_created	**Datetime**	**Descriptor**	Date the entry was created.

Table B.19: Table for form 7a – Function's user-experience (per inspector)

Form 7a – Function's user-experience (per inspector)			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	Name of the system under assessment.
username	**Alphanumeric (15)**	**Primary**	Member/inspector documenting the practice.
UG_ID	**Alphanumeric (50)**	**Primary**	Title of the user group.
Sno	**Integer**	**Descriptor**	Serial number of the documented practice.
Description	**Alphanumeric (255)**	**Primary**	Description of the documented practice.
Reference	**Alphanumeric (50)**	**Descriptor**	Reference to user requirement.
Score	**Integer**	**Descriptor**	Severity rating of the practice.
Experience	**Integer**	**Primary**	Indication of the experience of users to whom the specific

			practice refers to.
Func_id	**Alphanumeric (50)**	**Primary**	Title of the function.
Date_created	**Datetime**	**Descriptor**	Date the entry was created.

Table B.20: Table for form 8a – Relationship maintainability (per inspector)

Form 8a – Relationship maintainability (per inspector)			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	Name of the system under assessment.
username	**Alphanumeric (15)**	**Primary**	Member/inspector documenting the practice.
UG_ID	**Alphanumeric (50)**	**Primary**	Title of the user group.
Sno	**Integer**	**Descriptor**	Serial number of the documented practice.
Description	**Alphanumeric (255)**	**Primary**	Description of the documented practice.
Reference	**Alphanumeric (50)**	**Descriptor**	Reference to user requirement.
Score	**Integer**	**Descriptor**	Severity rating of the practice.
Experience	**Integer**	**Primary**	Indication of the experience of users to whom the specific practice refers to.
Date_created	**Datetime**	**Descriptor**	Date the entry was created.

Table B.21: Table for form 4 – Visibility (all inspectors)

Form 4 – Visibility (all inspectors)			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	Name of the system under assessment.
UG_ID	**Alphanumeric (50)**	**Primary**	Title of the user group.
Sno	**Integer**	**Primary**	Serial number of the

			documented practice.
Description	**Alphanumeric (255)**	**Descriptor**	Description of the documented practice.
Reference	**Alphanumeric (50)**	**Descriptor**	Reference to user requirement.
Assessors	**Integer**	**Descriptor**	Number of inspectors who identified the specific practice.
Score	**Integer**	**Descriptor**	(Average) Severity rating of the practice.
Date_created	**Datetime**	**Descriptor**	Date the entry was created.

Table B.22: Table for form 5 – Perceived usefulness & ease of use (all inspectors)

Form 5 – Perceived usefulness & ease of use (all inspectors)			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	Name of the system under assessment.
UG_ID	**Alphanumeric (50)**	**Primary**	Title of the user group.
Sno	**Integer**	**Primary**	Serial number of the documented practice.
Description	**Alphanumeric (255)**	**Descriptor**	Description of the documented practice.
Reference	**Alphanumeric (50)**	**Descriptor**	Reference to user requirement.
Assessors	**Integer**	**Descriptor**	Number of inspectors who identified the specific practice.
Score	**Integer**	**Descriptor**	(Average) Severity rating of the practice.
Date_created	**Datetime**	**Descriptor**	Date the entry was created.

Table B.23: Table for form 6 – Availability & approachability (all inspectors)

Form 6 – Availability & approachability (all inspectors)			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	Name of the system

			under assessment.
UG_ID	**Alphanumeric (50)**	**Primary**	Title of the user group.
Experience	**Integer**	**Primary**	Indication of the experience of users to whom the specific practice refers to.
Sno	**Integer**	**Primary**	Serial number of the documented practice.
Description	**Alphanumeric (255)**	**Descriptor**	Description of the documented practice.
Reference	**Alphanumeric (50)**	**Descriptor**	Reference to user requirement.
Assessors	**Integer**	**Descriptor**	Number of inspectors who identified the specific practice.
Score	**Integer**	**Descriptor**	Severity rating of the practice.
Date_created	**Datetime**	**Descriptor**	Date the entry was created.

Table B.24: Table for form 7 – Function's user-experience (all inspectors)

Form 7 – Function's user-experience (all inspectors)			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	Name of the system under assessment.
UG_ID	**Alphanumeric (50)**	**Primary**	Title of the user group.
Func_id	**Alphanumeric (50)**	**Primary**	Title of the function.
Experience	**Integer**	**Primary**	Indication of the experience of users to whom the specific practice refers to.
Sno	**Integer**	**Primary**	Serial number of the documented practice.
Description	**Alphanumeric (255)**	**Descriptor**	Description of the documented practice.
Reference	**Alphanumeric (50)**	**Descriptor**	Reference to user

			requirement.
Assessors	**Integer**	**Descriptor**	Number of inspectors who identified the specific practice.
Score	**Integer**	**Descriptor**	Severity rating of the practice.
Date_created	**Datetime**	**Descriptor**	Date the entry was created.

Table B.25: Table for form 8 – Relationship maintainability (all inspectors)

Form 8 – Relationship maintainability (all inspectors)			
Field	**Type (size)**	**Key**	**Description**
System_name	**Alphanumeric (50)**	**Primary**	Name of the system under assessment.
UG_ID	**Alphanumeric (50)**	**Primary**	Title of the user group.
Experience	**Integer**	**Primary**	Indication of the experience of users to whom the specific practice refers to.
Sno	**Integer**	**Primary**	Serial number of the documented practice.
Description	**Alphanumeric (255)**	**Descriptor**	Description of the documented practice.
Reference	**Alphanumeric (50)**	**Descriptor**	Reference to user requirement.
Assessors	**Integer**	**Descriptor**	Number of inspectors who identified the specific practice.
Score	**Integer**	**Descriptor**	Severity rating of the practice.
Date_created	**Datetime**	**Descriptor**	Date the entry was created.

Bibliography

1. Andre, T. S. (2000). Determining the effectiveness of the usability problem inspector: a theory-based model and tool for finding usability problems. Unpublished dissertation, Virginia Tech, Blacksburg, VA. Retrieved on 15/9/2007 from: http://scholar.lib.vt.edu/theses/available/etd-04122000-09440030/.
2. Antona, M., Kastrinaki, A., Mourouzis, A., Boutsakis, E., & Stephanidis, C. (2006). User-orientation inspection of ten European eServices: Results and lessons learned. FORTH – ICS / TR 373. Retrieved on 15/6/2007 from: http://www.ics.forth.gr/ftp/tech-reports/2006/2006.TR373_User-orientation_inspection_European_eServices.pdf.
3. Antona, M., Mourouzis, A., Kartakis, G., & Stephanidis, C. (2005). User Requirements and Usage Life-Cycle for Digital Libraries. In J. Jacko & V. Kathlene Leonard (Eds), Emergent Application Domains in HCI – Volume 5 of the Proceedings of the 11th International Conference on Human-Computer Interaction (HCI International 2005), Las Vegas, Nevada, USA, 22-27 July. Mahwah, New Jersey: Lawrence Erlbaum Associates. [CD-ROM].
4. Ascad Networks, (2007). Study: Screen Resolution Conclusions. Retrieved 25/6/2007 from: http://www.ascadnetworks.com/news.php?article=10&type=site.
5. Becker, S., & Lundman, D. (1998). Improving Access to Computers for Blind and Visually-Impaired – The Development of the Test-Method for Usability. In Proc. of the 3rd TIDE Congress, 23-25 June 1998, Helsinki, Finland.
6. Brooke, J. (1996). SUS: a "quick and dirty" usability scale. In P. W. Jordan, B. Thomas, B. A. Weerdmeester & A. L. McClelland (eds.) Usability Evaluation in Industry. London: Taylor and Francis.
7. Centeno, V. L., Kloos, C. D., Fisteus, J. A., & Alvarez, L. A. (2006). Web Accessibility Evaluation Tools: A Survey and Some Improvements. Electronic notes in theoretical computer science, 157(2), pp. 87-100.
8. Dillon, A. (2003). User Interface Design. MacMillan Encyclopaedia of Cognitive Science, Vol. 4, London: MacMillan, pp. 453-458.
9. Doulgeraki, C., Partarakis, N., Mourouzis, A., Antona, M., & Stephanidis, C. (2007). Towards Unified Web-based User Interfaces. FORTH-ICS Technical Report, TR-394. Retrieved on 19/10/2007 from: http://www.ics.forth.gr/ftp/tech-reports/2007/2007.TR394_Towards_Unified_Web-based_UI.pdf.
10. EMMUS (1999). Cello: Evaluation by Inspection. Retrieved on 23/10/2007 from: http://www.ucc.ie/hfrg/emmus/methods/cello.html.

11. eUSER (2004) eUSER Conceptual and Analytical Framework (first version). Kevin Cullen (Ed), eUSER Deliverable D1.1, Part A.
12. Gould, J. D., Boies, S. J., & Lewis, C. (1991). Making Usable, Useful, Productivity – Enhancing Computer Applications. Communications of the ACM, 34 (1), pp.74-85.
13. Gray, W. D., & Salzman, M. C. (1998). Damaged Merchandise? A Review of Experiments that Compare Usability Evaluation Methods. Human-Computer Interaction, 13(3), pp. 203-261.
14. Henry, S. L. Just Ask: Integrating Accessibility Throughout Design. Retrieved on 11/10/2007 from: http://www.uiaccess.com/accessucd/evaluate.html.
15. Human-Computer Interaction Lab, University of Maryland. QUIS: The Questionnaire for User Interaction Satisfaction. Retrieved on 23/10/2007 from: http://www.cs.umd.edu/hcil/quis/.
16. International Organization for Standardization (1998). ISO 9241-11:1998: Ergonomic requirements for office work with visual display terminals (VDTs) -- Part 11: Guidance on usability. Retrieved on 10/10/2007 from: http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=16883.
17. IT Accessibility & Workforce Division (ITAW), Office of Government-wide Policy, U.S. General Services Administration, (2001). Section 508 Standards. Retrieved on 3/11/2007 from: http://www.section508.gov/index.cfm?FuseAction=content&ID=12.
18. Matera, M., Costabile, M.F., Garzotto, F., & Paolini, P. (2002). SUE inspection: an Effective Method for Systematic Usability Evaluation of Hypermedia, IEEE Transactions on Systems, Man and Cybernetics – Part A, 32(1), pp. 93-103.
19. Microsoft.com. Colour. Retrieved on 15/11/2007 from: http://msdn2.microsoft.com/en-us/library/aa511283.aspx.
20. Mourouzis, A., Antona, M., Boutsakis, E., & Stephanidis, C. (2005). An Evaluation Framework Incorporating User Interface Accessibility. In C. Stephanidis, (Ed.), Universal Access in HCI: Exploring New Dimensions of Diversity – Volume 8 of the Proceedings of the 11th International Conference on Human-Computer Interaction (HCI International 2005), Las Vegas, Nevada, USA, 22-27 July. Mahwah, New Jersey: Lawrence Erlbaum Associates. [CD-ROM].
21. Mourouzis, A., Antona, M., Boutsakis, E., Kastrinaki, A., & Stephanidis, C. (2006). User-orientation Evaluation Framework for eServices: Inspection tool and usage guidelines, FORTH – ICS / TR 372. Retrieved on 13/6/2007 from: http://www.ics.forth.gr/ftp/tech-reports/2006/2006.TR372_User-orientation_Evaluation_Framework.pdf.

22. National Institute of Standards and Technology (NIST) (2002). WebSAT Static Analyzer. Retrieved on 24/10/2007from: http://zing.ncsl.nist.gov/WebTools/WebSAT/overview.html.
23. Nielsen, J. (2000). Designing Web Usability: the practice of simplicity. New Riders Publishing, Indianapolis, ISBN 1-56205-810-X.
24. Nielsen, J. How to Conduct a Heuristic Evaluation. Retrieved on 24/9/2007 from: http://www.useit.com/papers/heuristic/heuristic_evaluation.html.
25. Nielsen, J. Ten Usability Heuristics. Retrieved on 4/11/2007 from: http://www.useit.com/papers/heuristic/heuristic_list.html.
26. Nielsen, J. (1994). Usability Engineering. Morgan Kaufmann, San Francisco.
27. Nielsen, J., & Mack, R. L. (Eds.) (1994). Usability Inspection Methods. John Wiley & Sons, New York, NY, ISBN 0-471-01877-5.
28. Ravden, S. J. & Johnson, GI, (1989). Evaluating usability of Human Computer Interfaces: a practical method. Ellis Horwood, Chichester.
29. Redish, J., & Dumas, J. (1999). A Practical Guide to Usability Testing. Intellect, Ltd (UK).
30. Root, R. W., & Draper, S. (1983). Questionnaires as a software evaluation tool. Proceedings of CHI 83, 83-87. New York: NY: ACM.
31. Serco Usability Services. SUMI Questionnaire. Retrieved on 23/10/2007 from: http://www.usability.serco.com/trump/methods/satisfaction.htm.
32. Silius, K., Tervakari, A. M., & Pohjolainen, S. (2003). A Multidisciplinary Tool for the Evaluation of Usability, Pedagogical Usability, Accessibility and Informational Quality of Web-based Courses. PEG2003 - The Eleventh International PEG Conference: Powerful ICT for Teaching and Learning, 28 June - 1 July 2003 in St. Petersburg, Russia. Proceedings of PEG2003, [CD-rom].
33. Slatin, J., & Rush, S. (2003). Maximum Accessibility: Making Your Web Site More Usable for Everyone. Addison – Wesley.
34. TeLaRs, The University of Melbourne, (2001). Evaluating Technology-Enhanced Teaching and Learning. Retrieved on 23/9/2007 from: http://www.infodiv.unimelb.edu.au/telars/re/ete.html.
35. Thatcher, J., Waddell, C., Henry, S., Swierenga, S., Urban, M., Burks, M., Regan, B., & Bohman, P. (2002). Constructing Accessible Web Sites. Glasshouse.
36. Venners, B. (1996). Java's garbage-collected heap. JavaWorld.com. Retrieved on 15/11/2007 from: http://www.javaworld.com/javaworld/jw-08-1996/jw-08-gc.html.
37. W3C (2005). Involving Users in Web Accessibility Evaluation. Retrieved on 25/10/2007 from: http://www.w3.org/WAI/eval/users.

38. W3C, (1999). Web Content Accessibility Guidelines. Retrieved on 19/9/2007 from: http://www.w3.org/TR/WAI-WEBCONTENT/.
39. WebAIM. A Review of Free, Online Accessibility Tools. Retrieved on 3/9/2007 from: http://www.webaim.org/articles/freetools/.
40. WebAIM. Using the Cynthia Says Accessibility Validation Service. Retrieved on 24/10/2007 from: http://www.webaim.org/resources/cynthiasays/.
41. WebAIM. WAVE 3.0 – Web Accessibility Versatile Evaluator. Online at: http://www.wave.webaim.org/index.jsp.
42. Wharton, C., Rieman, J., Lewis, C., & Polson, P. (1994). The cognitive walkthrough method: A practitioner's guide. In Nielsen, J., & Mack, R. L. (Eds.), Usability inspection methods. New York, NY: John Wiley & Sons, pp. 105-140.
43. Wikipedia, the free encyclopaedia. Usability. Retrieved on 1/11/2007 from: http://en.wikipedia.org/wiki/Usability.
44. WebXACT. Watchfire Bobby. Online at: http://webxact.watchfire.com/.
45. Weinman, L. The Browser-Safe Web Palette. Retrieved 23/6/2007 from: http://www.lynda.com/hex.asp.
46. Wikipedia, the free encyclopaedia. Entity-relationship model. Retrieved on 2/11/2007 from: http://en.wikipedia.org/wiki/Entity-relationship_model.