Abstract

From the software engineering perspective, testing is a process of finding errors in the program code. From the user perspective, it may also be a process of modifying the application. This paper discusses how classifications embedded in forms may influence action and how people may use them as politically charged mediators in the testing negotiations. An ethnographically inspired study of handling a suggestion by a user demonstrates how two testing forms were used as mediators to include or exclude a perspective. These inclusions and exclusions were supported by classifications embedded in the forms. Classifications in one software testing form were sufficiently weak to be used by a user to include the user perspective. Classifications in another form appeared at first sufficiently strongly structured to be used by the designers to exclude this perspective. However, these classifications were also then challenged by one person who occupied an informal ‘user advocate’ position. Finally, some insights into how to augment the user perspective are outlined.

Keywords

Software testing, classifications, forms, negotiation, user perspective, ethnography

INTRODUCTION

Software applications are tested before they go into production. The general purpose of testing is to ensure the quality of applications. From the software engineering perspective, testing a piece of software is to execute its program code in order to find errors (Myers 1979). From the user perspective, however, testing may have other purposes as well. One such purpose could be the modification of the usability of applications. From the software engineering perspective, however, usability issues should have been handled earlier during information systems development. From the user perspective, however, this might not be possible before the application can be used and tested as a whole.

In negotiations during testing, artifacts such as forms can be used not only as neutral intermediaries, but also as politically charged mediators. Forms are a special category of artifacts since they may include strong classifications. Each box in a form involves at least one classification, which includes and excludes perspectives. What distinguishes forms from many other human-made artifacts is their relative inflexibility. Due to the built-in classifications, forms may serve one party better than another. Moreover, as we will see in the case studied here, one of the forms allowed a much wider range of uses than the other one. What is, however, even more significant is how these artifacts are used during testing.

In the present case, a group of users conducted the black box testing to find if the program does not work according to the specifications. This was supposed to demand no prior knowledge of the internal workings of the program. What the users, instead, were supposed to know were the inputs to be entered, the expected outputs and the intended goals. When errors in the outputs occurred, the users were supposed to connect them to the particular inputs. After this, the designers conducted the white box testing to find the exact sources of the errors in the program code. The designers tested the program code in detail with the test data generated based on their knowledge of the internal workings of the program. In testing, such salient artifacts as the test log form and the error report, were used to help recognize errors in the program code, to convey information about them from one party to another, and to prioritize them.

To illustrate the differences between how people perceive technologies, Orlikowski and Gash (1994) introduced the concept of ‘technological frames’. Because software applications are introduced as tools for certain work practices, their users are likely to assess their quality against their personal work experiences, and they may take an instrumental view of technologies, expecting “immediate, local, and task-specific benefits” (ibid., p.180). Thus, for user representatives, software testing gives an opportunity to study the realization of these benefits. Unlike the black and white box testing, the usability inspection is supposed to allow this (Nielsen & Mack 1994). According to them,
“[A] usability inspection is aimed at finding usability problems in an existing user interface design, and then using these problems to make recommendations for fixing the problems and improving the usability of the design” (ibid., p.3).

During these inspections, users’ future actions are “structured and defined in relation to the machine” (Woolgar 1991, p.89). Interestingly, Woolgar (1991) shows that even during these inspections, or trials as he calls them, designers may configure users. That is, they may define, enable or even constrain users. Obviously, this is even more likely to happen during the software testing activities, where error-free software is the main goal. Software testing sessions are thus potentially political even in their goals and methods.

Artifacts are built upon certain value orientations. Informed by them, artifacts classify the world in some particular ways. Value-bound artifacts have been discussed especially in the Participatory Design context (e.g. Bodker & Grønhaak 1991, Trigg et al. 1991 and Karasti 2001). The user perspective has been emphasized in those artifacts. They have been created to allow users to become equal design partners to designers during the multi-cultural requirements determination interaction. They have also been built to give users a stronger voice than normally may be the case. As an example, Karasti (2001) has studied video as a common communication channel for multi-cultural groups involving also users. In addition, Torvinen and Jalonen (2000) have introduced an innovative method, the Labour Game, which includes two sets of playing cards and the board as representations shifting focus of discussions from technical aspects to “power-related elements of change in work structures” (p.16). Also artifacts such as software testing forms may have political functions in addition to their other functions, for example, as coordination mechanisms (Carstensen et al 1995).

The objective of the present study is to discuss the roles that forms may take and how they may be used in the attempts to include or exclude a perspective. The forms discussed here supported the program code testing, but, at the same time, the user perspective was at least partly excluded by the designers. Finally, some strategies to strengthen the user perspective are outlined. Decreasing the structure of the classifications embedded in artifacts could be one potential way of enlisting users during negotiations.

CONSCRIPTION DEVICES/CLASSIFICATIONS IN NEGOTIATING

Representations are not the primary entities in the world, but people are. Hence, what turns out to be more significant than representations per se is how people use them or represent things for some purposes, and how meanings are co-constructed by those people. In other words (Hall 1997, p. 25), meanings are not only seen as imposed by a single individual or a group of people on the world we live in, as intentionalists would argue, but also as (co-)constructed by people within this world.

Henderson (1991) calls conscription devices those representations that are used by people who are engaged in putting together a joint product, such as negotiating an agreement. Conscription devices “enlist and constrain participation in creating and maintaining the object” (ibid, p.XXX). The meanings embodied in them are subject to be interpreted, re-interpreted and modified numerous times. Conscription devices, so to speak, function in a read-and-write mode. In this regard, they differ from immutable mobiles (Latour 1987) or intermediaries (Latour 1993) designed for conveying information from one party to another. A conscription device acts as a mediator, as “[o]ne who mediates; especially, one who interposes between parties at variance for the purpose of reconciling them; hence, an intercessor” (Webster’s Revised Unabridged Dictionary 1913). The “one” in this definition is interpreted here to include also material objects. A mediator is a resource used by intentional persons geared towards reconciling possibly conflicting concerns. Karsten et al. (2001) point out that conscription devices are mediators whose “[s]tructure forms a grammar for constructing the object [itself]” (ibid., p.90). In other words, they present a set of actual or presumed prescriptive notions about their correct use and thereby guide meaning co-construction. With conscription devices, therefore, closure signifies agreement.

Conscription devices reflect their owners’ classifications of the world. Bowker and Star (1999) define a classification as “a spatial, temporal, or spatio-temporal segmentation of the world” (p.10). For them, a classification denotes a metaphorical/conceptual or literal/material box into which an entity in the world may be put. Classifications organize our world by including and excluding things; that is, “[e]very standard and each category valorizes some point of view and silences another” (Bowker & Star 2000, p.156). For example, characterizing a usability suggestion only from one perspective by saying that it is insignificant is to exclude, consciously or not, that another person from a different social world could regard it as very relevant. Bowker and Star (2000) argue that classifications frame how we represent the past and sequence events in the present. They may even order our understandings of the past and present at particular moments.

If used in negotiating, forms may be regarded as conscription devices. This view contrasts with Star and Griesemer’s (1989) perspective on forms. For them, forms are not meant to be used in negotiating but they have been designed to convey immutable information and remove local uncertainties within diverse worlds. In either face-to-face or artifact-mediated local negotiations, classifications of one social world embodied in conscription
During the design/implementation stage, which took place between July 2000 and May 2001, there were only a few sessions in which the client was involved. In these sessions, the design documents were tested. The users were also invited to these sessions, but only a marginal role was given to them during this stage. Even if they were consulted especially about the user interface screens, the users were found to have trouble commenting on them. The new system was designed and implemented in three distinct episodes during this stage. After each episode, the implemented system was tested. The third testing episode that I attended took place in the turn of May and June 2001. The application was finally installed in RAY in the beginning of fall 2001.

In this article, the third testing episode and one of the days is discussed, and the focus is laid on one issue registered by a user. This issue was then classified by the Incognito designers, and finally it was prioritized by the joint testing team.

THE RESEARCH SETTING

This study examines a project that was established to determine, design, and implement an information system for Finland’s Slot Machine Association (RAY). RAY professes to be “a significant, widely-known organization with a legal monopoly to function as a gaming operator. RAY’s gaming revenue is utilized efficiently and reliably to benefit people needing assistance and support” (http://www.ray.fi) via charitable associations providing information, services and different kinds of care. RAY hired the Incognito software house to build the information system whose primary purpose was to manage the financial transactions between the RAY funding department and the charitable associations. The focus here is on managing the payment plans. At the beginning of each year, the organization sends RAY its payment plan, a proposal for the amount and schedule for funding. This may also have been negotiated during a meeting. First, this payment plan is scanned and attached to the particular organization. Then, the relevant data is stored and the plan assessed. All active payment plans can be seen in a list and each of them can be modified.

The project took place over a period of one and a half years between January 2000 and June 2001. The requirements determination stage started in January 2000 and ended in June 2000. There were fifteen one-day requirements determination sessions and one two-day session during the project. User participation varied depending on the session. Sometimes it was more engaging, but in some sessions it was more like “pseudo-involvement” (Newman & Noble 1990) during which users were given only an information-provider position. Especially during one of the sessions, as their manager was absent, the users were allowed to be notably active. During the design/implementation stage, which took place between July 2000 and May 2001, there were only a few sessions in which the client was involved. In these sessions, the design documents were tested. The users were also invited to these sessions, but only a marginal role was given to them during this stage. Even if they were consulted especially about the user interface screens, the users were found to have trouble commenting on them. The new system was designed and implemented in three distinct episodes during this stage. After each episode, the implemented system was tested. The third testing episode that I attended took place in the turn of May and June 2001. The application was finally installed in RAY in the beginning of fall 2001.

It is, however, worth remembering that “no such tool can be defined once and for all” (Bowker & Star 1999, p.158). Different social worlds have their own classifications of the world. Thus an interesting question is who controls what happens when people from different social worlds and, by implication, different classifications, conflict. For Brown (2001), it would be too simplistic to argue that there is a linear relationship between formalisations and power. As he reminds us, “the relationship between activity and formalisation is something to be investigated in particular situations, we cannot say that rules are simply “ignored”, “blindly followed”…in each case there is some form of use and adjustment” (ibid., p.308). For me, which classification ‘wins’ is a matter of co-constructing and negotiating. Based upon Hall’s approaches, meanings are not only imposed on different social worlds but also co-constructed by their members, as they encounter them in a negotiation situation.

There are different sorts of conscription devices, and they differ in terms of their grammar or classifications which this grammar stems from. Let us take two examples, one without a prescriptive grammar or a set of classifications of one social world and another with a grammar. Some conscription devices such as a whiteboard do not bring with them a set of classifications. Instead, people from diverse social worlds classify the world differently. What becomes primary is the interaction situation and the present actors themselves. What is the outcome of negotiating around a whiteboard depends on what happens in the interaction. Not even a whiteboard could allow an equal communication pattern to exist but, in the end, what is talked about and how and what is emphasized in the first instance depends on people who may or may not allow ‘user-centered’ classifications. Another example is a form, such as a tax return form that has a structural grammar including a set of classifications of the social world, which has created it. This sort of form frames how its owners would represent the past and sequence events in the present. Its classifications come in a specified form, and they may have an impact on other persons from different social worlds. Despite this, “[t]he particular negative or positive effects of a formalisation will depend in part on the “play” which is involved in their use” (Brown 2001, p.309), or on the negotiation situation, how people influence the interaction. What then matters most is for what purposes the form is used and by whom and how meanings are co-constructed around it.
The testing team met on the day in question, June 7. There were six participants from RAY, all acting as software testers. The users were represented by three Supervisors and the company/sponsor by one Manager and two RAY IS Project Managers. In addition, there were two designers, Jouni and Petri, from Incognito who supported the testers. The role of these designers was to support the testers by providing guidance and the ‘right’ working methods. There were also two researchers from the University, myself and another person, observing the sessions. Much of the data in this description is taken from my field notes and Designer Jouni’s notes.

Software testing sessions were seen to be focal for RAY as the performance of the system could be ensured, the users could familiarize themselves with the system and learn to use it and, finally, possible ideas for future development could be expressed [Jouni’s notes, May 30]. The designers, in turn, would benefit from the sessions since they could find errors before the system would go into production, learn from errors, evaluate methods and conventions in use, and develop their customer relations. The purposes of the testing sessions were thus in potential contradiction. The designers emphasized that software testing was not a matter of suggesting new ideas or re-developing systems. New ideas, they said, should be handled only as ideas for future development. Designer Petri, for example, paid special attention to certain types of errors: for example, he gave “more weight to getting the database flawless” [my notes, June 6]. The potential contradictions became real with borderline cases. According to Jouni’s notes, it was one of the responsibilities of the designers to assess whether a certain issue was an error or a new idea. However, the users did not necessarily have the same interpretation.

Each testing episode comprised three stages. These are here called error registration, classification and prioritization. These stages were part of the standard testing practices of the Incognito software house. The first stage, error registration, took place in the morning and afternoon on June 7. Before this, there had already been four days for error registration. Everybody was present but the configuration of the group changed occasionally during the day. During this stage, the software testers were supposed to find errors in the program code and to register them in the test log forms they were given. There was a text field in this form for freely describing errors that were found.

The software testers sat in their own corner (ring A, Figure 1) and the designers wandered around the room, or “the Holy of Holies” as one designer called it. Generally speaking, the atmosphere was emotionally charged. The excerpts from my notes, however, show that the designers found the session sensitive and charged whereas the users were more relaxed:

*The atmosphere there in “the Holy of Holies” has changed after the first days; the users are now joking, telling stories, laughing, eating cookies and even belching as they are testing the program.*

*Designer Jouni lets me know that testing is mentally tormenting and that the customer relations are vulnerable. Many times he comes over to me and is wiping sweat off his face… He says that you should never get mad; you have to tolerate anything and be as neutral as possible.*

The second stage, classification, took place in the afternoon in this same day. The designers assembled for a meeting held in their own corner B in Figure 1. The software testers from RAY were absent. During this stage, the designers moved the registered errors from the test log forms into the error management system using the error report form, which was also used to classify these errors. The designers used three classification tools (“status”, “priority” and “status history” fields). There were four status attributes that may be used: “corrected”, “pending”, “ignored” and “expect information”. The first of these four attributes, “corrected”, emphasizes that an error has been fixed. The second attribute “pending” tells that an error is currently being fixed. The third one “expect information” is obvious. Finally, if a thing was labelled by the designers as “ignored”, it was typically and mistakenly regarded by someone as an error even though it was not (e.g. an error had been fixed but it was
still regarded as an error). The priority of an error may be set to “minor”, “major” or “fatal”. Finally, the “status history” field may be used to communicate information on the errors among the designers.

The third stage, prioritization, took place in the same afternoon. The users, the three supervisors, were absent. Figure 2 presents the seating arrangement in the meeting. The symbol “A” refers to the three RAY representatives. The symbols “B” and “C” refer to the Incognito designers, Jouni and Petri. Finally, the symbols “D” and “E” refer to the researchers from the University. The participants discussed the errors one by one and prioritized them either as “minor”, “major” or “fatal” in priority. The purpose was to decide which of these errors could be repaired immediately and which ones were collected as ideas for future development.

![Figure 2: The seating arrangement during the prioritization meeting](image)

**METHOD**

For data collection purposes, I and my fellow researcher used ethnography as “[its] enterprise is to describe the world as perceived by those within that world” (Harper 2000, p. 245). In the analysis, however, I decided to employ external concepts more than an absolutely faithful ethnographer would do. I studied the observed social world and its social and material cultures as “anthropologically strange” (Hammersley & Atkinson 1995, p.XXX) since “usually, the setting is not previously known in an intimate way” (Emerson et al. 1995, p.1) to the researcher(s). Even though I was a participant observer, I did not participate in all phases of the project. For me, during the project, there were also some situations intended for ‘real’ insiders only. I felt I should not intrude on them. Ethnography seemed to increase my closeness to the cultures being studied and enabled me to penetrate what “experiences and activities mean to them” (ibid., p.12, italics in the original), and I studied information as “marshalled”, “worked up”, “reviewed”, “circulated”, “used”, “stored”, and finally, “forgotten about”, and, in general, inscribed on artifacts (Harper 1998; 2000, p.246).

I had been present in the project meetings already during the earlier stages of the project. This was beneficial for me since I had already succeeded in establishing some sort of rapport with most of the members of the project. I also found that those members were more frank with me than before. Especially Designer Jouni had learned to trust me more and more during the project. He frequently came to me and opened up during moments of trouble. In this way, the primary data of this study included both the project members’ and my own interpretations.

During the registration stage, I was a complete observer almost all the time. I tape-recorded the interactions. Only occasionally did I interrupt the other members of the project and ask in-situ questions. I wrote notes on what happened and collected all the materials, notes and documents that were used. During the classification stage, the designers moved to their own corner. I did not join them as I found being there intrusive. Thus, I did not tape-record these conversations. Later, however, I was offered all the materials they were working on there. During the prioritization stage, I was present again and tape-recorded the conversations. I was, however, a passive observer only, writing notes.

I started analyzing the data by reading my field notes, listening to the recordings and familiarizing myself with the forms that were used. Gradually I began to ask myself questions (as suggested by Emerson et al. 1995, p.146) as to what people were doing, what specific means or strategies they were using, how members were talking about what was going on and what I had seen going on there. Some themes began to emerge from this learning process. I decided to employ the strategy of “portraying a rare moment” which “persuasively reveal[s] members’ concerns” (ibid., p.175). In this particular case, the ‘rare moments’ were the usability issues which were more exceptional than the errors in the program code. These moments enabled me to understand how the interpretations of the different cultures deviated from each other. I decided to choose one of the usability issues not only because its handling demonstrated how classifications embedded in the forms may be used strategically to influence others, but also because there was an interesting negotiation around it during the testing stage.
REGISTGERING THE SUGGESTION

The chain of events began when Supervisor Erkki challenged the usability of one part of the user interface and registered what was found as “not an error but a suggestion”. He found that the user interface did not allow him to open the scanned documents of a particular due item, for which there was a payment plan, on the same screen where the numerical data of the payment plan was shown. The user interface, instead, forced him to open also the “Choose Payment Plan” screen if he wanted to open the original documents in addition to seeing the numerical data stored on the “Payment Plan – Payment by RAY” screen. It also appears that for Supervisor Erkki, usability problems were errors. By contrast, for the designers, only errors in the program code were considered valid.

Immediately after he had voiced his suggestion, Supervisor Erkki had a chat with Designer Jouni, who was quite unresponsive as if he was skirting around the problematic issues. He responded evasively to Erkki’s opinions and questions. After talking with him for some time, Designer Jouni turned to his colleague, Petri. They stood aside as if having something to hide from the others, especially Erkki. Jouni then secretly asked whether this need had been discussed earlier. Petri recalled it had been but he could not remember the outcome. They seemed to have doubts about what the user had just proposed. They also compared the suggestion with what had been specified in the determination documentation. This may be interpreted as their tendency to prioritize what is stated there. However, Designer Petri suddenly remembered that: “There were some discussions about this earlier but it was preferred that [the document viewer] wouldn’t be opened” [Audiotape]. This can be interpreted as downplaying the suggestion. Soon, Designer Jouni turned also to Manager Timo who could not help either: “Nor can I remember why it would be” [Audiotape]. The utterance “why it would be” implies that there was no acute need for this usability improvement in his opinion. After this, Designer Jouni aired his opinion: “It has been implemented like this, it is preferred that it be that way” [Audiotape]. Manager Timo, after that, used political rhetoric to argue for its closure: “In practice, payment plans rarely arrive” [Audiotape]. Thus, he also seemed to downplay the problem. Soon, he, however, proposed that the issue should be thought over once more. But did he really mean that? All in all, during this error registration stage, the issue was left unsettled.

Supervisor Erkki refused to concede his suggestion although even Manager Timo downplayed it. He did not give in, but instead tried to strengthen his argument. To succeed, he employed the test log form or more precisely, put something in black and white in its error description field. He wrote: “On the ‘Payment Plan – Payment by RAY’ screen, you should be able to select the documents of the already chosen due item so that you don’t have to first go to the ‘Choose Payment Plan’ screen, select the documents of this due item, and then, after that, select the due item on the other screen”. He seems to advocate using fewer screens and even showing more information on a screen. Nevertheless, Erkki was quite aware of what was seen as acceptable and what was not. This is indicated by his remark in the error description field: “not an error but a suggestion”.

The test log form did not differentiate between errors in the program code and usability problems. The designers clearly assumed that the test log form was to be used for registering errors in the program code only, as was made clear, for example, in Jouni’s notes in which it was said that “errors will be separated from pure ideas for future development either by labelling them ‘future development’ or by putting them in a separate list”. However, a separate list was never explicitly handed over to the testers. In this light, the label ‘future development’ appears as a strategic expression, which separates acceptable errors from unacceptable suggestions. It is worth emphasizing here that the designers talked about suggestions with the testers mainly in development’ appears as a strategic expression, which separates acceptable errors from unacceptable suggestions. It is worth emphasizing here that the designers talked about suggestions with the testers mainly in political rhetoric to argue for its closure: “In practice, payment plans rarely arrive” [Audiotape]. Thus, he also seemed to downplay the problem. Soon, he, however, proposed that the issue should be thought over once more. But did he really mean that? All in all, during this error registration stage, the issue was left unsettled.

Supervisor Erkki refused to concede his suggestion although even Manager Timo downplayed it. He did not give in, but instead tried to strengthen his argument. To succeed, he employed the test log form or more precisely, put something in black and white in its error description field. He wrote: “On the ‘Payment Plan – Payment by RAY’ screen, you should be able to select the documents of the already chosen due item so that you don’t have to first go to the ‘Choose Payment Plan’ screen, select the documents of this due item, and then, after that, select the due item on the other screen”. He seems to advocate using fewer screens and even showing more information on a screen. Nevertheless, Erkki was quite aware of what was seen as acceptable and what was not. This is indicated by his remark in the error description field: “not an error but a suggestion”.

The test log form did not differentiate between errors in the program code and usability problems. The designers clearly assumed that the test log form was to be used for registering errors in the program code only, as was made clear, for example, in Jouni’s notes in which it was said that “errors will be separated from pure ideas for future development either by labelling them ‘future development’ or by putting them in a separate list”. However, a separate list was never explicitly handed over to the testers. In this light, the label ‘future development’ appears as a strategic expression, which separates acceptable errors from unacceptable suggestions. It is worth emphasizing here that the designers talked about suggestions with the testers mainly in an evasive manner. Only seldom did they commit a responsive dialogue with the users in those cases where the user interface was challenged. In contrast, what they emphasized was the significance of clearly describing suggestions. It is worth emphasizing here that the designers talked about suggestions with the testers mainly in political rhetoric to argue for its closure: “In practice, payment plans rarely arrive” [Audiotape]. Thus, he also seemed to downplay the problem. Soon, he, however, proposed that the issue should be thought over once more. But did he really mean that? All in all, during this error registration stage, the issue was left unsettled.

The test log form did not differentiate between errors in the program code and usability problems. The designers clearly assumed that the test log form was to be used for registering errors in the program code only, as was made clear, for example, in Jouni’s notes in which it was said that “errors will be separated from pure ideas for future development either by labelling them ‘future development’ or by putting them in a separate list”. However, a separate list was never explicitly handed over to the testers. In this light, the label ‘future development’ appears as a strategic expression, which separates acceptable errors from unacceptable suggestions. It is worth emphasizing here that the designers talked about suggestions with the testers mainly in an evasive manner. Only seldom did they commit a responsive dialogue with the users in those cases where the user interface was challenged. In contrast, what they emphasized was the significance of clearly describing errors in the program code.

All in all, it was shown that the test log form and its description field can be used for registering suggestions, as well. Even though the form was originally meant primarily for registering errors in the program code, it did not, however, include overly specified classifications preventing users from also registering usability suggestions. In this particular case, the form was used by one user to strengthen his own position in negotiations in which his suggestion would be negotiated with the designers and also the other stakeholders of the project.

CLASSIFYING THE SUGGESTION

During this stage, the designers moved the suggestion registered in the test log form via the error report form (Figure 3) to the error management system database and classified it. The suggestion was labelled; it was given attributes. That is, it was re-interpreted and re-determined by the designers in terms of a set of classifications. From this point forward, Supervisor Erkki’s suggestion was also ‘owned’ by the designers since they had labelled it.

After being classified by the designers, there were two conflicting views of the same suggestion in the form, the original suggestion set by one user and how the designers classified it. During the previous stage, Manager Timo

Sarkkinen, (Paper #209)

14th Australasian Conference on Information Systems

26-28 November 2003, Perth, Western Australia
had diplomatically postponed dealing with the usability issue in question. We may speculate that it was for this reason that the designers now labelled it as having the “expect information” status. This classification seems to indicate either that the designers felt they could not commit an explicit choice yet or that they complied with Manager Timo’s suggestion to postpone the issue, or both. This view is also supported by the empty “status history” field. The designers did not do anything about the suggestion for the moment. Leaving the field empty means that the suggestion will be discussed in the final prioritization session. Even though Erkki’s suggestion was marked only as postponed, not ignored, the designers, however, appeared to disagree with it. Their view was expressed implicitly by the low priority that was given to it. The designers appeared to use this “priority” field to strengthen their view for the prioritization meeting. They set the priority attribute to “minor”, which may look like a side issue at first sight. Later, it turned out to be a politically charged classification. This indicates that they regard the suggestion as insignificant. In comparison with other possible labels, “major” and “fatal”, it becomes clear that the designers set low value on the suggestion. Even though “minor” is only a word, this label puts an impressive characterization not only on the suggestion, but also on Erkki’s opinion. The distinction between “minor”, “major” and “fatal” is a strategic one, which may be capable of including and excluding perspectives. The designers’ negative attitude towards modifying the software may explain why there is such a tool in the first instance. There is, however, a reason for being negative about new suggestions. Jouni once let me know that otherwise there would be the risk that “discussing wishes would perhaps go on until the end of time” [My notes]. Interestingly, Supervisor Erkki was not told about this pre-prioritization. This supports the view that the designers were against dealing with new ideas at this stage.

<table>
<thead>
<tr>
<th>Error Number:</th>
<th>812</th>
<th>Date:</th>
<th>7.6.2001</th>
</tr>
</thead>
<tbody>
<tr>
<td>Title:</td>
<td>V3_Payment_Plan</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Description:</td>
<td>1/SUPERVISOR ERKKI 070601</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>Not an error but a suggestion:</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>On the ‘Payment Plan – Payment by RAY’ screen, you should be able to select the documents of the already chosen due item so that you don’t have to first go to the ‘Choose Payment Plan’ screen, select the documents of this due item, and then, after that, select the due item on the other screen.</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Status:</td>
<td>Expect information</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Priority:</td>
<td>Minor</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Error Noticed:</td>
<td>System Test</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Correction date:</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Author:</td>
<td>RAY</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Status history:</td>
<td>JOUNI</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 3: Error #812 on the error report form

PRIORITIZING THE SUGGESTION

Supervisor Erkki’s suggestion was finally prioritized in the prioritization meeting. By then, his interpretation in the error report form had been already modified by the designers’ classifications. Not surprisingly this stage appeared to be quite stressful for the designers. Designer Petri even called it the “climax”. One reason for its stressfulness may be that there was no extra time for developing the user interface. Now, the designers spent their time with what they considered the more fatal errors. Moreover, the designers felt it was no longer their task to take responsibility for usability issues. However, customer relations were “vulnerable”, as Jouni told me, and for this reason, the designers found it difficult to say no to new ideas. The irony that I found in the designers’ talk is telling. They, for example, tried to make me believe that testing is “fun”. Laughing at what they said, however, implied the opposite. Jouni even added that they are like “masochists”.

The prioritization itself was a matter of interpreting and negotiating. Designer Jouni, who chaired the session, used the error report form to bring up the suggestion for discussion. Copies of this form were handed out also to the others, with the attributes given by the designers. Immediately after he mentioned the suggestion, Jouni remained silent for a moment. Then he said: “Well, will it [the “Choose Payment Plan” screen] be opened or not, how is it?” [Audiotape]. IS Project Manager Keijo then rejected the attribute as “minor”. He re-interpreted Erkki’s suggestion, or its essence, as follows: “Erkki primarily means that one could open those documents also on the “Maintain Payment Plans” screen [“Payment Plan – Payment by RAY” screen] via the “Windows” menu selection” [Audiotape]. He used at least one understatement. For example, by uttering “also”, he avoided being too critical of the current implementation. In addition, he did not mention the possibility that Erkki referred to the direct manipulation of documents, which was implied by his remark “[o]n the…screen”. It is, however, impossible to know exactly what Erkki meant with his suggestion since he was absent. After Keijo’s turn,
Designer Jouni hesitated for a while. He, however, finally accepted the suggestion: “Well, let us put it that way” [Audiotape]. Also Designer Petri and Manager Timo finally agreed with the re-interpretation. That this happened was not only thanks to Keijo’s rhetoric, but also to his informal status as a user advocate (Mambrey et al. 1998) in the sense that he emphasized the user perspective. For Mambrey et al., a “user advocate” needs to have experience with user practices in the past, and he or she needs to understand the technical issues, as well. For example, during the requirements determination stage, he had encouraged the users to express their opinions. He also had expert knowledge of usability issues.

DISCUSSION

This study has investigated classifications as embedded in the forms and used by people during the software testing negotiations. The study confirms the initial assumption that what matters most is for what purposes a form is used and by whom and how meanings are co-constructed around it. Classifications in one software testing form appeared sufficiently weak to be used by a user to include the user perspective. Classifications in another form, however, appeared first sufficiently strong to be used by the designers to exclude this perspective. However, also these classifications were finally challenged by one person who occupied an informal ‘user advocate’ position.

Classifications embedded in the forms are not prescriptive. They cannot order interaction. Nevertheless it is possible that they may influence negotiations in one way or another. We may speculate that the more strongly-structured a classification is, the more difficult it may be to challenge it. It is, however, not that simple in any situation. In the end, what matters most are people. In negotiations, classifications embedded in the artifacts may be used as mediators for inclusive-exclusive ends by people. Then, these classifications may be used to influence and even persuade other people. All in all, classifications impose meanings on those who are supposed to ‘read’ them. However, in negotiations, meanings are not simply imposed on the ‘readers’ but also, and more likely, co-constructed by the ‘writers’ and ‘readers’. Moreover, it is natural that some people are, for some reason, more influential than others. The following discussion concentrates on for what purposes and how the forms were used.

The designers introduced the test log form for registering errors in the program code. Knowing this, a user, however, decided to register his usability issue. The error description field he used includes only weakly-structured classifications, which classify the world only to the extent that something needs to be written in it. Despite its original purpose, the form allowed the user perspective. The designers, of course, attempted to lead the user gently in the direction they favoured. By implicitly playing down the suggestion during the registration stage, they tried to exclude the user perspective. The user, however, used the form to strengthen his perspective. His suggestion was not just said, but there was something in writing.

The test log form was used by the user to impose the importance of his suggestion on the designers. The designers, however, continued to play down its importance. The error report form offered the designers two classifications that they used for this purpose. To play down the suggestion, and, at the same time, to exclude the user perspective, the designers set the priority of the suggestion to “minor”. The user had already once succeeded in challenging the software engineering perspective. In this light, it is quite understandable that the designers preferred to use these two classifications rather than continuing to argue about it. The error report form was used as a mediator intended for reconciling the conflicting views. The status, however, was not set to “ignored” to play down the importance of the suggestion even further. This may be because this kind of directness would have risked the secondary purpose of testing, that is, taking care of customer relations.

During the prioritization meeting, the error report form imposed two views, both what the user and the designers prefer. It is worth remembering, however, that the user was not present in this meeting. Moreover the suggestion was not the original one; it was labelled as insignificant on the form. In this regard, the form emphasized the designers’ view and was used to exclude the user perspective. In the meeting, the designers did not mention the “minor” priority aloud. The error report form, the mediator, was harnessed to this end.

As we remember, meanings are not only imposed, but also constructed by people. The software engineering perspective was challenged by IS Project Manager Keijo, the user advocate. Even though this advocate succeeded in challenging the label “minor”, he needed to re-interpret the suggestion, or filter it, as Mambrey et al. (1998) would call it. For Keijo, the classifications embedded in the form were not capable of excluding the user perspective. However, without the original suggestion, the ‘advocate’ could have been incapable of counteracting the software engineering perspective. What was imposed by the error report form was finally reconstructed around it.

It was shown that classifications may be used to distinguish between legal/acceptable and illegal/unacceptable. These distinctions may influence action in general, but they can also include and exclude different perspectives. As was already mentioned, the aim of both black and white box testing is to find errors in the program code. For
this reason, it is understandable that the classifications embedded in the error report form were used by the designers to exclude the user perspective. Happily, one could say, the suggestion by the user was accepted as re-interpreted. However, there is still one concern. We may ask what would have happened if there had not been a user advocate at all? Very likely, it would have been more unlikely that the suggestion would have gone through.

What should be done about new usability suggestions during black and white box testing activities? If the user perspective is likely to be excluded during the software testing activities, software quality could (and should) be ensured from the user perspective at some earlier stage. Every suggestion is important to its owner. For this reason, it is dangerous if suggestions cannot be handled. Already at some earlier stage, we could enlist users in a systematic manner, for example with usability inspection (Nielsen and Mack 1994).

The concern with their approach is that the existing user interface may include an array of strongly-structured classifications, which may stem from misconceptions about user needs. If a ‘high-fidelity’ user interface is to be tested, it may delimit users’ possibilities to address the usability problems. It may be assumed that users do not dare to criticize strongly-structured user interfaces. In contrast, weakly-structured conscription devices could be more appropriate tools for enlisting users. Unlike strongly-structured conscription devices, they could encourage users’ own classifications. For example, a whiteboard is a weakly-structured artifact. Also with this artifact, it is interaction that matters. Designers should avoid their own classifications to allow the user perspective. Not only would technologies be changed in these interaction situations, but also the ways that technologies are used could be improved. This would be one part of the unfinished revolution (Dertouzos 2001).

**SUMMARY**

Based on the data analysis, the following findings can be reported:

(i) The test log form is composed of weakly-structured classifications, which do not include the software engineering perspective.

(ii) The test log form does not exclude the user perspective; for example, one of the users succeeded to register a usability suggestion.

(iii) The error report form contains, among others, the strongly-structured “priority” classification, which appeared to be a politically charged mediator.

(iv) The designers used the priority label “minor” to play down the registered usability suggestion, and to exclude the user perspective.

(v) The designers used the priority label “minor” for distinguishing between the more significant errors and the less significant usability suggestions.

(vi) The priority label “minor” on the error report form was not just an intermediary to one of the present members, the ‘user advocate’; he saw it as a politically charged mediator and challenged the given priority label, re-interpreted the original suggestion and succeeded in pushing it through.

No panacea for including the user perspective can be provided. Only some insights into how to augment it may be provided.

(vii) If the user perspective is bracketed outside the white and black box testing activities, as it normally is, usability issues could (and should) be handled earlier.

(viii) Users could be enlisted earlier by using weakly-structured conscription devices, which do not tend to configure users.

(ix) These conscription devices that act as catalysts would contain weakly-structured classifications, augmenting users’ own classifications and inspiring their thinking.

Especially the latter two of the insights (viii and ix) are worth contemplating further. More concrete suggestions of how to augment users’ own classifications need to be developed.

**REFERENCES**


ACKNOWLEDGEMENTS

I want to thank all of you who have contributed to this paper. Especially, I wish to express my thanks to Pekka Muukkonen with whom I had discussions just before I polished the theory section. This conversation in a local restaurant stimulated my thinking. However, Professor Eija Karsten has been the most significant person of all. She has given me her valuable time and even written many sections of this paper with me. Thanks for that.
COPYRIGHT

Jarmo Sarkkinen © 2003. The authors assign to ACIS03 and educational and non-profit institutions a non-exclusive licence to use this document for personal use and in courses of instruction provided that the article is used in full and this copyright statement is reproduced. The authors also grant a non-exclusive licence to ACIS03 to publish this document in full in the Conference Papers and Proceedings. Those documents may be published on the World Wide Web, CD-ROM, in printed form, and on mirror sites on the World Wide Web. Any other usage is prohibited without the express permission of the authors.