

# Case study: Are CUP attributes useful to developers?

**Trausti Þorgeirsson**  
University of Iceland  
Hjardarhaga 2-6, 107 Reykjavik  
traustt@hi.is

**Marta Kristín Lárusdóttir**  
Reykjavik University  
Ofanleiti 2, 103 Reykjavik  
marta@ru.is

## ABSTRACT

In this paper a case study of a classification scheme for usability problems called CUP (Classification of Usability Problems) [1,3] is described. The individual attributes are analyzed according to how helpful they are for developers to understand, prioritize and fix a defect. Additionally factors are analysed that determine whether developers decide to fix a defect or not.

The results show that developers found information on in what task the problem was found, the context in the UI (User Interface) where the problem was found and the description of the problem helpful in understanding, prioritizing and finding a solution to the problems. Other attributes like the severity rate was not rated helpful at all by one of the developers and only helpful for prioritizing the problem by the other developer.

The developers had fixed 7 out of 53 problems two months after the usability tests and had decided not to fix 33 out of the 53 problems at that time. The 13 remaining problems were on schedule still 11 months after the usability tests. Further research is needed to understand what can be done to get higher success rate for the usability problems being solved by the developers.

## Author Keywords

Case study, CUP, usability problems, classification attributes, downstream utility

## ACM Classification Keywords

## INTRODUCTION

When conducting think-aloud sessions usability evaluators not often analyse the usability problems found in a structured way [2]. This could be because a systematic approach and a classification scheme is needed. This study was conducted to measure to what extent developers find useful the results of usability tests which have been

analysed by a classification scheme called CUP (classification for Usability problems).

The CUP scheme was developed by Hvannberg and Law [1,3]. The goal of CUP is to: *“Classify Usability Problems (UPs) further in order to give the developers better feedback to improve their understanding of the UP, help them manage the maintenance of usability, enable them to find effective fixes for UPs as well as preventing such problems from reoccurring in the future”*.

Three studies have been conducted using the CUP scheme. The first was in 2005 on The Owl (The University of Iceland’s teaching web), the second one was carried out in spring 2006 on Galvos at the Institute of Energy [3] and the current study. The first two studies were conducted to check the validity and reliability of CUP. In the current study the goal is to study what factors of the CUP scheme are important to developers when they decide which defects to remove and how to prioritize them.

## MATERIALS AND METHOD

Ten usability tests were made, the results were analysed and described according to CUP and delivered to the developers. Two months later a meeting was conducted by the usability specialists to gather information on how useful the classification was to developers.

## The usability test

Usability tests were conducted on a new version of software called Workhour. An old version had been in use for several years, but in the new version the user interface was changed extensively. There are four main user groups of Workhour; ordinary users that work on shifts and those that work regular hours. The other two main user groups are managers that work on shifts and those that don’t.

The main tasks for ordinary users working on shifts is to check their monthly plan for shifts, ask for a day off and check if they have fulfilled all their work obligations for that month. The main tasks for regular users are asking for holidays and check if they have been too many hours off work. The Workhour system is very useful to managers, because they can do much of their organizing work in Workhour like check if all timestamps for their employees are correct, insert information about an employee that is sick and get an overview of how many have been sick over a particular period to name a few.

Ten usability tests were conducted by two usability specialists on the new version of Workhour running on a test database two months before it was installed. Five regular users took part in the test, four working on shifts in hospitals and one working on regular hours in a state institution. There were also 5 managers that took part, two working on shifts at hospitals and three working regular hours, two at a state institution and one at a software company.

Each user solved six or seven tasks in think aloud tests which were adjusted to their ordinary tasks. The total number of tasks in the study was 17. The tasks were made by one of the developers of the user interface that has good connections to the users. The user tests were conducted at their ordinary working place, so a lot of contextual information was also gained. Two usability specialists conducted the tests; one was the organiser and one the data recorder. Additionally everything that was said was recorded on tape.

### Analysing usability problems according to CUP

From the verbal protocol and observation 53 specific usability problems were analysed according to CUP classification scheme in table 1. One usability specialist described the problems for the attributes: Defect ID, trigger, frequency, description, context and defect removal activity. Two usability specialists judged the impact factors individually; severity, failure qualifier and expected phase. On a meeting the mismatching rating was discussed until agreement.

Two versions were made of the usability problem lists, one long list with all the usability problems and another one where the problems had been split up according to the tasks it was found in. Additional information on the task was also added in that format. The main outcomes of the user tests were presented to the stakeholders at the software company; the project manager, developers and their manager. At that meeting the two developers involved in the study got the two lists each of the usability problems.

### Analysing what CUP attributes were useful

Two months after the usability tests, the developers gave status on how many of the problems they had fixed, how many they planned to fix and how many they wanted to solve another way. For fourteen of the 53 problems developers filled out a form with 10 questions. On a five point scale, developers were asked to rate three things. First, how confident they were that the described problem was really a usability problem. Second, they were asked how well they understood the problem. Third, they were asked to prioritize fixing the problem. Finally, they were asked what CUP-variables were useful for understanding the problem prioritizing it and thinking up a solution for it. These results were gathered on a meeting with the developers.

Attribute		Values	
Defect ID		Identification number	
Frequency		Number of users/experts that experience/predict a UP	
Trigger		Describes what an user is doing when s/he discovers the UP i.e. task scenario, heuristic, reflective question	
Context		Describes in what part of the user interface the user/expert was when the UP occurred	
Description		Concise description of the UP	
Defect Removal Activity		Usability Evaluation Method e.g. user test and heuristic evaluation	
Impact	Severity	Indicates what effects the UP had on the user/expert: Severe, Moderate, and Minor.	
	Task Efficiency (%/min)	Only for UPs found using user tests (UT), calculations based on the tasks scenario that the user was performing.	
	Task Effectiveness		Completion rate
			Mean time on task (min)
	Instances of Frustration		
Instances of Help Sought			
Failure Qualifier		Describes how the user/expert experienced a UP: Missing, Incongruent Mental Model, Irrelevant, Wrong, Better Way and Overlooked.	
Expected Phase		Indicates in which phase of a software development lifecycle the developer is expected to be able to fix the UP: Task analysis and context of use (TAN), Functional requirements (FUR), Quality attribute analysis (QAN), Conceptual modelling (COM), Dialogue design (DIA), Navigational design (NAV), Presentation design (PRE) and Implementation (IMP)	
Actual Phase		The phase where the UP was fixed: Same values as Expected Phase.	
Type of Fault Removed		Describes what in the user interface was changed to fix the UP.	
Cause		Aimed to understand why the developers committed the error: Personal, Communication, Technical, Methodological, Managerial and Review.	
Error Prevention Technique		Ideas on what can be done in the future to prevent the fault	

Table 1 List of CUP attributes and their values. The Post-CUP attributes are highlighted in grey.

## RESULTS

In this section statistics are given on how the 53 usability problems gathered in this study were categorized. The CUP scheme has mainly three sections, descriptive attributes including description of the problem, how many users experienced the problem and in what context in the user interface the problem occurred. Secondly there are three subjective attributes that the evaluators judge, the severity rating, the failure qualifier and the expected phase. Third there are the Post-CUP attributes which are filled in by the developers when the problem has been fixed.

### The Descriptive Attributes

In table 2 the frequency of users experiencing problems is shown. Most of the problems are only experienced by one user. While describing the usability problems the level of detail was kept very high, so problems were not joint if there was some slight difference in the way the users experienced the problems.

**Table 2: The frequency of users experiencing a problem**

Problem experienced by	Number of problems
7 users	1 problem
4 users	1 problem
3 users	1 problem
2 users	5 problems
1 user	45 problems
<b>Total</b>	<b>53 problems</b>

It has been suggested in [5] to split the usability problems up in two categories: low-frequency and high-frequency. High-frequency defects are those which four or more users experience and low-frequency those below that. When using this definition we have 50 low-frequency and 2 high-frequency defects. Another idea is to classify high-frequency defects as those which more than one user experiences. That categorization gives 8 high-frequency defects in this study.

The trigger describes in what task or tasks the problem was found. From that information the frequency of problems found in each tasks can be calculated and is listed in table 3.

**Table 3: List of number of problems found in each tasks**

Total number of problems per task	Frequency of tasks
9 problems	1 task
8 problems	1 task
6 problems	2 tasks
5 problems	1 task
4 problems	5 tasks
3 problems	1 task
2 problems	2 tasks
1 problem	3 tasks
No problem	1 task
<b>Total</b>	<b>17 tasks</b>

As can be seen in table 3 it was most common to find 5 problems in a task, but the variation of problems found in each task is quite high.

The data on the context describes where in the user interface the problem was found. In table 4 the frequency of problems found in each context is listed.

**Table 4: Frequency of problems found in each context**

Context of the User Interface	Frequency of problems
Menu	12 problems
Shift schedule	6 problems
Calendar	4 problems
Drop down lists	4 problems
Time records	4 problems
Links on the home page	3 problems
Contexts with 1 or 2 problems	20 problems
<b>Total</b>	<b>53 problems</b>

Most of the problems were in menu context, which are the links on the left hand side of the screen used for navigating in the system. These were confusing to the users, they mixed up words that are used for different things so they chose first a wrong link often because the word used was not familiar to them.

### The Subjective Attributes

In table 5 the results are shown for how the usability specialists rated the severity of the problems.

**Table 5: The severity rating of the usability problems**

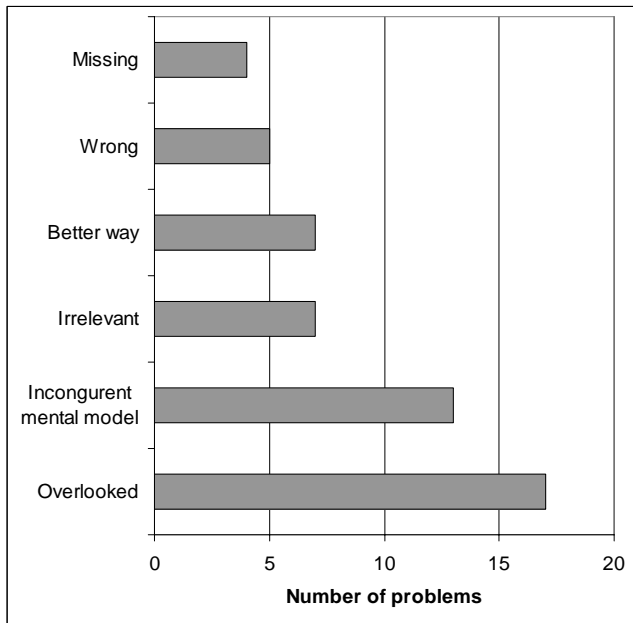
Severity rating	Number of problems
Severe	22 (42%)
Moderate	20 (38%)
Minor	6 (11%)
Irrelevant	5 (9%)
<b>Total</b>	<b>53 (100%)</b>

These results show that the usability specialists rated more than 40% of the problems being severe and almost 40% of them as being moderate. Both of the usability specialists attended all the usability tests, so they had observed all the problems.

In figure 1 the frequency of each category of the *Failure Qualifier* is given. An interesting result here is that incongruent mental model and overlooked make up for half the cases. Other categories are used less. This means that in most cases, the user doesn't notice a part of the interface, or has the wrong idea about how it works. In this category there is a significant overlap, in particular between missing, incongruent mental model and overlooked. The suggested reason here for this overlapping is that the user may expect some entity to be present in the UI or misses some entity

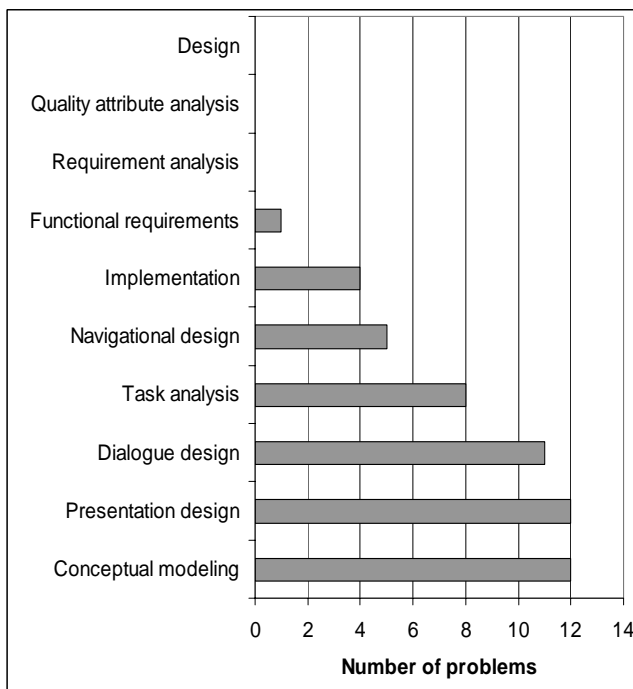
that is present because his mental model of the UI does not match with it.

**Figure 1: Frequency of problems in each category of Failure qualifier for Worktime**



The frequency of usability problems in each category of the *Expected* phase is shown in figure 2. Three categories are never used; requirement analysis, quality attribute analysis, and design. The functional requirements category is only used once. This implies that this attribute can be simplified.

**Figure 2: Frequency of problem in each category of the Expected phase rating for Worktime**



## DECISION ON SOLVING THE PROBLEMS

A new version of the software tested was done two months after the usability tests and installed at the hospitals and state institutions. At that point information on how much of the usability problems were fixed was gathered. In table 5, information on how the developers decided to react to the problems is listed.

**Table 5: Reactions to the problems**

Reaction	Frequency of problems
Fix right away	7 problems
On schedule for fixing	12 problems
Will not be fixed	28 problems
Add to the help of the system	5 problems
Checking it out	1 problem
<b>Total</b>	<b>53 problems</b>

These results were rather disappointing, especially that the developers decided only to fix 7 problems and that they had decided not to do anything with more than half of the problems. The severity rating of the problems fixed were: 3 severe problems, 3 moderate and one minor, so the fixing did not seem to be in correlation to the severity rating of the usability specialists.

In private conversations with one of the developers and one of their managers, 11 months after the usability tests were conducted no additional problems had been fixed. The reason the manager gave was that there was very much shortage of people in the group.

## ATTRIBUTES INFLUENCING ACTIVITIES

To do an indepth analysis of what CUP attributes of the developers considered helpful when deciding on solving the problems, the developers were asked to fill out a questionnaire for 14 problems out of the 53 problems found.

In table 6 and 7 the rating of how helpful each attribute in the CUP sceme is according to the two developers are shown. The percentages show how large a proportion of the 14 defects that the two developers considered helpful. In most cases the two developers agree. As expected, the attribute *frequency* does not help with understanding a problem or thinking up a solution but it does help with prioritizing it. Both developers believe that *trigger*, *context* and *description* are helpful with all three categories. One of the two developers does not think *severity* helps with prioritizing a problem which is somewhat surprising. Neither developer considers *expected phase* or *failure qualifier* helpful with understanding a problem. –That is probably normal since they both wrote that they found the text description of problems sufficient. Neither developer finds *failure qualifier* helpful in finding a solution for a problem. This is disappointing since *failure qualifier* should be easy to understand and relate to for a developer,

especially in this context. Perhaps, the developers do not find it helpful to know the exact nature of a user interface defect but only what caused it so they can directly pursue that in the code. That way, there is perhaps too much emphasis on the symptoms, at the cost of the cause in the developers' minds.

According to this data, trigger, context and description are mainly what the developers use for understanding, prioritizing and finding a solution to a problem.

**Developers' rating of problem descriptions regarding confidence, prioritization and understandability**

**Table 6: The ratings of developer 1.**

CUP attribute	Understanding defect	Prioritizing	Solution
Frequency	0%	71%	0%
Trigger	100%	93%	71%
Context	100%	93%	71%
Description	100%	93%	71%
Severity	0%	0%	0%
Failure q.	0%	0%	0%
Exp. Phase	0%	0%	0%

**Table 7: The ratings of developer 2.**

CUP attribute	Understanding defect	Prioritizing	Solution
Frequency	0%	93%	0%
Trigger	100%	100%	100%
Context	100%	100%	100%
Description	100%	100%	100%
Severity	0%	93%	0%
Failure q.	0%	93%	0%
Exp. Phase	0%	0%	86%

As previously stated, the developers rated the defects on a 5 point scale regarding their confidence that a defect was indeed a defect, its prioritization and its understandability. Both developers agreed that they understood the defects well. The prioritization was also similar between them. Both developers put most problems in categories 1 and 2 (where 5 is most significant and 1 least significant). This implies that they don't find the defects very important to fix. The priorities were low in spite of the fact that about half the defects were classified as severe. Even so, one developer thinks *severity* is an important classifier in prioritizing defects.

**CONCLUSION**

We have described results from a study using the CUP scheme. The developers rated what CUP-attributes helped them with understanding defect, prioritizing it and finding a solution for it. We found that both raters believe trigger, context and description are helpful with all three categories. Neither rater thinks *failure qualifier* is helpful in finding a solution for a problem. This is disappointing since *failure qualifier* should be easy to understand and relate to for a developer.

The developers also rated the problems descriptions regarding confidence, prioritization and understandability. We found that developers did not put very high priorities on the problems, which is a bit surprising given how many severe defects there were. They generally said that they understood the problems well but did not express a high confidence in the problems indeed being problems. The developers had fixed 7 out of 53 problems two months after the usability tests and had decided not to fix 33 out of the 53 problems at that time. The 13 remaining problems were on schedule still 11 months after the usability tests. Further research is needed to understand what can be done to get higher success rate for the usability problems being solved by the developers.

**REFERENCES**

1. E.T. Hvannberg and E. L. Law, 2003, *Classification of Usability Problems(CUP) Scheme*: Interact, p. 655-662
2. Norgaard, M, Hornbæk, K., 2006, *What Do usability Evaluators Do in Practice? An Explorative Study of Think-Aloud Testing*, DIS 2006, p. 209 – 217.
3. S.G. Vilbergsdóttir, E.T. Hvannberg, and E.L Law 2006. *Classification of usability problems (CUP) scheme: augmentation and exploitation*. In Proceedings of the 4th Nordic Conference on Human-Computer interaction: Changing Roles (Oslo, Norway, October 14 - 18)
4. S. G. Vilbergsdóttir (2006): *Classification of Usability Problems: Engaging Software Development Teams in Analyzing Usability Issues*. Master thesis.
5. S.G. Vilbergsdóttir, E.T. Hvannberg and E.L. Law 2007, *Impacts of Classification of Usability problems (CUP) on system Redesign*. Chi 2007 Workshop (San José April 29<sup>th</sup> 2007)