

# MIS-USE CASES TO EXPLORE EMOTIONAL REQUIREMENTS

**Jennifer Rode**

University of Cambridge

Computer Lab, 15 JJ Thompson Ave.

Cambridge CB3 0FD

jennifer.rode@cl.cam.ac.uk

**Shailey Minocha**

The Open University

Faculty of Maths & Computing

Walton Hall, MK7 6AA

s.minocha@open.ac.uk

---

## ABSTRACT

The role of emotion in UIs has been discussed in the affective computing literature, but we address how to detect, identify, reduce and prevent *obstacles* from occurring. We have developed mis-use cases as a means of understanding how user's emotions alter the user-system interaction. They are generated from what-can-go-wrong scenarios and can be used to explore how to mitigate or prevent *obstacles* caused by users' emotional states and individual character traits when they are interacting with technology.

## Keywords

Emotions, Use Cases, Usability Requirements.

## 1. INTRODUCTION

We see little mention of emotion in discussions of how to create user interfaces, but ironically, when considering the defining characteristics of human and computer in science fiction the predominate difference is emotion. The android can take on the speech, appearance, and mannerism of a human, but emotion is outside of their reach. Only recently has emotion taken centre stage in UI design [5]. When looking carefully at the literature we can see a number of communities that talk about the role of emotion and personal traits in creating an interface. Emoticons (e.g. :) for smile) were developed to convey emotion in text-based communication. Usability goals have been extended to include emotional comments and list concepts like "emotionally fulfilling" and fun, but there is no systematic guidance on how to achieve these goals. Much work has been done in affective computing, where the computer takes on human traits to help the user complete tasks [4], [8]. Pickard and Klein's paper [6] reviewed work on systems that detect and label human emotion. They claim the goal

This rectangle must be left blank for the  
copyright notice

of this type of affective computing is to recognize and respond to users emotions in such a way to meet the users' needs. However, no concrete guidelines of how to explore and meet the emotional usability requirements have been established. When usability requirements are developed, a designer makes assumptions about the user's computing experience and physical abilities, as well as specific domain, and demographics. What we do not see is a discussion of how the emotions a user brings with him in the interaction should be identified during requirements elicitation phase or addressed when in design.

In this paper, we propose the application of concrete mis-use cases, a variant of the what-can-go-wrong-scenario to systematically evaluate the emotional needs and character traits that users bring with them to the system. Consideration of mis-use cases will help generate usability requirements to cater for these what-can-go-wrong situations. Before introducing our adaptation of mis-use cases we briefly summarize the mis-use cases origins: the scenario and the use case.

## 2. RELATED WORKS

A *scenario* is narrative description of a task, describing the user's interaction with the system. It describes, step by step, the procedures that a user has to follow in order to get the task done, as well as features and behaviour of the system with which the user interacted while performing the task.

While a *success scenario* describes a goal being achieved, a *what-can-go-wrong scenario* describes a situation where the user faces problems in his interaction and such scenarios might end with the user abandoning his goal. The following is an example of a scenario "A bank customer discovers that they need money, and go to the nearest Bank. They establish their identity, and specify they desire to withdraw funds. The system checks to see if the customer's balance is sufficient, dispenses cash and updates the balance. The user collects the cash, and signals the end of the transaction." Relevant what-can-go-wrong-scenarios include, the system being out of cash, the system being out of order, or the user forgetting means of identifying themselves.

A *use case* is a collection of related successes and what-can-go-wrong scenarios, that describe actors (users) using the system to support a goal [2]. A

scenario is an instance of a use case, one path through the use case [2].

Use cases are created during the requirement elicitation phase of the software development life cycle to describe functions the system should be able to perform. An *essential use case* [3], sometimes called a business use case, is a simplified, abstract, generalised use case that captures the intentions of a user in a technology- and implementation-independent manner. It focuses more on what a user would like to do (his intentions or purpose of using a computer system), and what the system's responsibilities need to be, than on how this will be achieved. A *concrete use case* is not as personalised as a scenario, and is less generic than an essential use case and is, in fact, derived from it. It describes the user's actions and the system's response in a two-column format. It has references to the user interface design features of the system with which the user interacts while performing his actions. For our ATM example, a concrete use case is:

<u>User Action</u>	<u>System Response</u>
Insert Card	Read card, request pin
Enter PIN	Verify PIN, Display option menu
Select Option	Display account menu
Select Account	Prompt for amount
Enter Amount	Display Amount
Confirm Amount	Return Card
Take Card	Dispense Cash if available
Take Cash	

We use the notion of concrete use case as a cornerstone for developing a means of systematically evaluating how users' emotional needs will affect the design process, which we will call *concrete mis-use cases*. We have chosen to work with concrete as opposed to essential use cases because dealing with the abstractions of technology and emotions simultaneously could be too intellectually taxing for the UI designer. Working within a defined set of technology will make the problem of understanding the affect of user emotions less taxing. Finally, we expect that analysing the interactions with user traits would occur later in the design process, once some technological constraints have been set. While we are limiting our application of concrete mis-use cases to dealing with user's emotions and traits, concrete mis-use cases can be applied to other aspects of usability requirements as well.

Mis-use cases have been proposed for exploring what-can-go-wrong scenarios. [1], [7]. Sindre and Opdahl define mis-use case as "the inverse of a use case, a function that the system should not allow" [7]. They replace the actor in the traditional use case with

a mis-actor who intentionally or un-intentionally (that is, by accident) is the key figure in these mis-use cases. A typical example of a mis-actor in a mis-use case would be the thief attempting to steal credit card information on an e-commerce site.

Mis-use cases can provide designers with a way to explore potential *obstacles*. We define *obstacles* as situations in user-system interaction when a user may not be able to fulfil his goals, or a user may not be able to fulfil his goals by expending reasonable amount of resources - cognitive resources that include cognitive effort, and other resources such as time. We call these *obstacles*, rather than simply usability problems to emphasize that these extend beyond interface issues into the social, cultural, environmental and organizational domains. Examples of *obstacles* include absence of crucial information or data, insufficient training to the users, or inappropriate skills of the users. Obstacles can often cause breakdowns.

A *breakdown* is a situation where the user is unable to achieve his goal because of obstacles(s). For example, a situation where a customer abandons shopping on an e-commerce site and moves to a competitor site represents a breakdown. In our ATM use case, failure to take the cash is a breakdown, because acquiring cash is the user's goal.

Mis-use cases have been applied so far to elaborate safety [7] and security requirements [1]. In this paper, we propose the application of mis-use cases to elaborate and validate usability requirements. Usability requirements typically neglect the human element: the user's emotional state and character traits, and instead focus on the skills and experience of the user. We propose the application of concrete mis-use cases as a means to evaluate how human's emotions can impact on user-system interaction.

### 3. MIS-USE CASES FOR USABILITY

We have chosen to personify a number of emotions and character traits as the Seven Dwarfs from Snow White. We feel being *happy*, *sleepy*, *sneezy* (e.g. ill), *dopey* (clumsy), *grumpy*, *bashful* (fearful of technology) are representative of the sorts of emotional states that users bring with them. *Doc* represents the literally-minded user. We do not suggest the dwarves represent an all-inclusive list of user emotions and traits; they merely serve as a mnemonic for addressing some of the basic emotions. The purpose for introducing the dwarves is as mis-actors in our mis-use cases.

This notion of using unconventional mis-actors was introduced by Alexander, when he suggested 'weather', 'bad-luck' and the 'devil' [1]. He explains: "The use of metaphor and anthropomorphism may appear colourful and even frivolous. However, human reasoning has evolved in a social context to permit sophisticated judgments about possible hostile

intentions of other intelligent agents. Use/Mis-use Case analysis deliberately exploits this faculty in the service of systems engineering.” While Alexander suggests that mis-use cases can be extended to other areas like usability, and gives an excellent example “Simple Simon presses the wrong button” no work has been done to fully extend mis-use cases for elicitation & elaboration of usability requirements [1].

Relevant emotions and traits which are likely to be produced by the interface can be gathered in the requirements elicitation phase through observation of existing systems, or by interviewing prospective users about their interaction with the current systems. While we see potential for technology-independent essential mis-use cases, we have chosen to focus on concrete ones. Human emotion often requires a trigger, which in our case is the technology, therefore we see concrete mis-use cases as coming into play after the initial requirements generation phases which involve essential use cases.

To illustrate how mis-use cases can be applied to usability requirements, let us look at some examples. While prior work with mis-use cases has been focused on pictorial mis-use case diagrams, we will be using text-only concrete mis-use cases in the two-column format. We feel the later will allow for a richer discussion of the mis-use case. They will allow us to expand on the different usability requirements generated in the requirements elicitation phase.

Let us assume a couple of usability requirements: (1) Training: This should be a walk-up and use interface, with no training required (2) Task time: Assuming the user has sufficient money to withdrawal, and knows their PIN, 75% of users with should be able to withdraw funds in under two minutes, 25% under five minutes. Successful task completion is defined as the user leaving with the receipt, amount of money desired, and their card.

Hypothetically, let us assume the Seven Dwarves are standing in line at the ATM waiting for their Disney Dollars first thing in the morning.

### 3.1 The Happy User

The concrete use for ATM described earlier is the ideal path— a success case. Unfortunately, the next six guys in line are not always so lucky.

### 3.2 The Sleepy User

The sleepy slightly dazed, before the first cup of coffee in the morning user, might walk off without their cash, as shown below:

<u>User Action</u>	<u>System Response</u>
Insert Card	Read card, request pin
Enter PIN	Verify PIN, Display option menu
Select Option	Display account menu

Select Account	Prompt for amount
Enter Amount	Display Amount
Confirm Amount	Return Card
Take Card	Dispense Cash if available
Walk off without cash	Alert user to their error
Come back and retrieve cash	

This mis-use case alerts us to the breakdown of the user walking off without their cash. This might be mitigated if an alarm goes off when if the user forgets their cash.

### 3.3 The Sneezey User

Sneezey has perpetual allergies; he is next in line holding Sleepy’s much needed coffee.

<u>User Action</u>	<u>System Response</u>
Insert Card	Read card, request pin
Enter PIN	Verify PIN, Display options
User sneezes, spills coffee	

This mis-use case might suggest adding a sealed keypad on the interface. Doing so would prevent clumsy users spilling drinks and shorting delicate circuitry. Additionally, this mis-actor reminds to meet the needs of ill users needs to accommodate slow reaction times, and poor vision caused by watery eyes, by anticipating needs for longer timeouts and larger fonts respectively.

### 3.4 The Dopey User

Many users are poor typists, and while well-intentioned could enter information incorrectly.

<u>User Action</u>	<u>System Response</u>
Insert Card	Read card, request pin*
Enter PIN incorrectly*	Reject Pin, eat the card

\*Sequence repeats some number of times

This user’s circumstances have prevented them from getting cash entirely—a complete breakdown. The security needs here might very well necessitate eating the card, however there are customer service options. A call button could allow the user seek assistance, alternatively on-screen instructions could tell a user how to react once the situation occurred. These instructions could tell the user to go into the branch, or circumstances permitting a friendly teller could come out and offer assistance.

### 3.5 The Grumpy User

Dopey then turns around and asks the guy behind him in line to borrow some cash. This puts the grumpy user in a fowler mood then usual:

<u>User Action</u>	<u>System Response</u>
Insert Card	Read card, request pin
Enter PIN violently	Keys lock up

This mis-use case might alert the designers to an additional requirement: the system must not only withstand the button pressure of the normal user, but the frustrated one as well. Especially, as it is common for frustrated users to press keys harder with the hope that the computer will understand.

### 3.6 The Bashful User

The bashful user might just be afraid of technology and approach tasks with excessive caution. They might have privacy concerns, shade the screen, and thus operate the whole UI one-handed. Caution might make them check their balance before proceeding, and bad-luck might come into play.

<u>User Action</u>	<u>System Response</u>
Insert Card	Read card, request pin
Enter PIN	Verify PIN, Display options
Select Option	Display account menu
Select Balance Inquiry	Dispense receipt w/balance Wind blows receipt away

So while Bashful darts off after his receipt, Grumpy and Dopey solve their financial problems, and Bashful returns to a much-diminished bank account. This use case alerts us to the possibility that paper receipts might present privacy problems. Or suggest the need for security cameras to check the person who takes the cash entered the pin.

### 3.7 And Doc

Doc is a particularly literally minded user thus would expect a key labelled "any key" instead of "enter."

<u>User Action</u>	<u>System Response</u>
Insert Card	Read card, request pin*
Enter PIN	
Looks for the "Any Key"	

## 4. CONCLUSIONS

The application of mis-use cases to usability requirements illustrate the need for additional requirements related to human emotional states and character traits. The ATM must be usable by individuals who are frustrated, ill, tired, intimidated by the interface, or users that are just inherently clumsy, or literal minded. This use of mis-use case is similar to "what can go wrong" scenarios. Mis-use cases provide a rich vehicle for evaluating the social, environmental, and technological obstacles in user's interaction when a user comes to a system with attitudes that differ from how the system is designed.

Concrete mis-use cases provide a means of systematically evaluating how users' emotions and

traits affect their interactions with the system and the obstacles that can arise. These emotions and traits serve as the mis-actors. By evaluating these what-can-go-wrong scenarios we have a means of exploring obstacles and potential breakdowns and can develop additional requirements to prevent or mitigate these threats and improve the user interfaces which are satisfying to the user on both cognitive and emotional levels.

## 5. FUTURE WORK

Based on Sindre and Opdahl's work [7], we have developed a template to guide the elicitation and documentation of obstacles, mis-use cases and usability requirements related to users' emotions and character traits. We will be discussing the template and its usage in a full-paper in the near future. In the meantime, those interested can contact the authors for more details of the template.

## 6. ACKNOWLEDGMENTS

This research was supported by the EPSRC Grant GR/R60867/01 (2002-2004), and Dr. Marian Petre of the Open University to whom we are most grateful.

## 7. REFERENCES

- [1] Alexander, I. (2003). "Misuse cases: Use cases with hostile intent". *IEEE Software*, 20, 1, 58-66.
- [2] Cockburn, A. (2001). *Writing Effective Use Cases*, Addison Wesley.
- [3] Constantine, L. L. and Lockwood, L.A.D. (1999). *Software for Use*, ACM Press.
- [4] Heckman, C. E. and Wobbrock, J. O. (2000). *Put Your Best Face Forward: Anthropomorphic Agents, E-Commerce Consumers, and the Law*. Agents, Barcelona Spain, ACM Press.
- [5] Norman, D. (2003). *Emotional Design: Why We Love (Or Hate) Everyday Things*. Basic Books.
- [6] Picard, R. W. and Klein, J. (2002). "Computers that recognise and respond to user emotion: theoretical and practical implications." *Interacting with Computers*, 14, 2, 141-169.
- [7] Sindre, G. and Opdahl, A.L. (2000). "Eliciting Security Requirements by Mis-use Cases." *Proc. TOOLS Pacific 2000*, 120-131.
- [8] Tsukahara, W. (2001). *Responding to Subtle, Fleeting Changes in the User's Internal State*. Proceedings of the SIGCHI conference on Human factors in Computing Systems, Seattle, WA USA, ACM Press, 77-84.