Electronic Communications of the EASST
Volume 45 (2011)

Proceedings of the
Fourth International Workshop on Formal Methods
for Interactive Systems
(FMIS 2011)

Capturing the distinction between task and device errors in a formal model of user behaviour

H. Huang, R. Rukšėnas, M.G.A. Ament,
P. Curzon, A.L. Cox, A. Blandford, D. Brumby

16 pages

Guest Editors: Judy Bowen, Steve Reeves
Managing Editors: Tiziana Margaria, Julia Padberg, Gabriele Taentzer
ECEASST Home Page: http://www.easst.org/eceasst/ ISSN 1863-2122

Capturing the distinction between task and device errors in a formal model of user behaviour

H. Huang[1] (huayih@eecs.qmul.ac.uk), R. Rukšėnas[1], M.G.A. Ament[2], P. Curzon[1], A.L. Cox[2], A. Blandford[2], D. Brumby[2]

[1]Queen Mary University of London
School of Electronic Engineering and Computer Science

[2]University College London, UCL Interaction Centre

Abstract: In any complex interactive human-computer system, people are likely to make errors during its operation. In this paper, we describe a validation study of an existing generic model of user behaviour. The study is based on the data and conclusions from an independent prior experiment. We show that the current model does successfully capture the key concepts investigated in the experiment, particularly relating to results to do with the distinction between task and device-specific errors. However, we also highlight some apparent weaknesses in the current model with respect to initialisation errors, based on comparison with previously unpublished (and more detailed) data from the experiment. The differences between data and observed model behaviour suggest the need for new empirical research to determine what additional factors are at work. We also discuss the potential use of formal models of user behaviour in both informing, and generating further hypotheses about the causes of human error.

Keywords: human error, formal models of human behaviour, cognition.

1 Introduction

In the complex daily working environment of medical professionals, errors are occasionally made. Often these have only minor consequences, but sometimes much more costly outcomes result. Recent research, particularly in the field of human-computer interaction, has promoted a shift away from the perspective where blame is attached entirely to the individual when mistakes are inevitably made. This change in perspective has been partly motivated by the realisation that people are not, and cannot reasonably be expected to be, completely infallible. This is especially true in high-stress environments such as the hospital workplace. It is also in these situations that the costs of errors, both financially and in human terms, are often the highest.

One way of systematically detecting the potential for these somewhat rare but costly errors is by using plausible models of human cognition. However, the frame problem [MH69] must be dealt with when modelling people. In terms of logical models, this relates to the need to include additional 'common-sense' axioms in order to generate plausibly realistic patterns of inference. In the context of modelling cognition in general, it relates to the precise scope, form and amount of common-sense knowledge necessary for a cognitively plausible model. Also, when constructing a model of reality hard decisions have to be taken about which aspects of reality are abstracted over and which are represented with relatively high fidelity.

To enable the construction of a useful model of human cognition, we need to be clear from the outset about the problem we are trying to solve with the model. Here we try to detect *systematic* medical device errors. In particular the class of errors that may be rare, but are also not completely random. The errors of interest are ones which at least in principle could be *predicted*, *a priori*, to plausibly occur or reoccur in the future due to systematic cognitive causes. We are concerned most with those errors that have high-cost consequences associated with them.

Our ultimate aim is to have a system that can highlight design flaws that may lead to such preventable errors. For the purposes of this paper, we define an error simply as an action that deviates from some prescribed sequence for achieving the intended outcome. However, this is only one of several plausible definitions. Hollnagel [Hol05] gives a more detailed discussion of some of these alternative definitions of error.

1.1 Related work

In the past, a variety of experimental work has identified some of the causes of human error, such as Byrne and Bovair's work demonstrating the effect of working memory on post completion errors (Byrne & Bovair, [BB97]). Here we focus on task and device errors based on the experiments of Ament *et al*. [ACBB10]. This particular distinction dates back to various earlier works such as that of Cox & Young [CY00], Kirschenbaum *et al*. [KGEM96] and Hiltz *et al*. [HBB10].

The work presented in this paper is not specifically concerned with generating new experimental results but in supporting such work, as well as creating predictive formal tools that allow deeper explanations for observed human behaviour. In terms of using formal modelling to support empirical work we build on Ruksenas *et al*. [RBCB09] where model checking was used to help understand experimental results about human error and whether all relevant concepts were included in the explanations given. Su *et al*. [SBB07, SBBW09] adopt a similar idea using formal models of low level cognitive concepts related to the attentional blink phenomena. Their work differs to ours in being based on lower level aspects of human cognition and on simulation-based model exploration.

A simple approach to the modelling of plausible user behaviours involves writing both a formal specification of the device and task models for that device, to support reasoning about the behaviour of the interactive system [MD95, Fie01]. Task models, however, describe only a small fraction of real user behaviour - in particular they do not really deal with human fallibility. An alternative approach is to specify users as they are (Butterworth *et al*. [BBD00]). This is the idea underlying alternative approaches to formal user modelling [DD99, DBMD95]. By modelling the user 'as is', we can gain many insights into how and why specific user behaviour are generated/observed. To reason about the behaviour of an interactive system, a formal user model is combined with a formal specification of the device. Both models are then considered as central components of the whole integrated system in an approach known as *syndetic modelling* [DBDM98].

Ruksenas *et al*. [RBCB09] present one example of this approach. They model human cognition as a set of production rules expressed in higher-order logic. The generation of the next plausible user-behaviours then involves computing an 'overall salience value' for each possible user-action expressed in the model. A key difference between their model and most simulation-based models of human cognition is the underlying assumption of *non-deterministic* user behaviour throughout. Their model operates based upon *sets* of cognitively plausible user-actions at each point in the interaction. This allows for simultaneous exploration of the consequences of *all* of the plausible user-actions that could be taken at each point. This contrasts with most simulation-based approaches where only a single path of potential user-actions is explored at once.

Ruksenas *et al*. [RBCB09] also describe a case study based on a fire engine dispatch scenario using their generic user model (**GUM**). We present here a follow-on case study to further validate the model, investigating the behaviour and performance of the model using a different scenario and therefore under a different instantiation. As part of the process, we also gained valuable practical experience and insight into the utility of the iterative model-refinement method suggested by Ruksenas *et al*. [RBCB09].

1.2 Choice of scenario

We based this study on the experiment of Ament *et al*. [ACBB10]. It presents experimental data about the effect of memory load on device and task based errors. It is based on an experimental micro-world concerned with a doughnut-making task. This context provides a good proving ground for validation and further refinement of the GUM as it involves both an independent setting, and the investigation of a different aspect of human error. Furthermore the kind of errors considered (device and task based slip errors) fall within the intended scope of the generic model, but were not explicitly designed into it. As noted earlier, the class of errors that we are interested in are systematic deviations from a prescribed action-sequence as a result of slips, rather than say lack of skill or knowledge. In the case of Ament *et al*. [ACBB10], the intended outcome was to fulfil orders for doughnuts through following a single prescribed sequence of actions. The errors investigated were deviations from that sequence despite the participants being trained and having demonstrated their knowledge of, as well as ability to do the task.

1.3 Research questions and contribution

The first question that we address in this paper is whether the GUM is currently expressive enough to encapsulate *all* the concepts relating to human cognition as presented in Ament *et al*. [ACBB10]. We investigate the completeness and appropriateness of the current concept-space as defined in the GUM with respect to the concepts addressed in the experiment.

Our second question concerns the model-checking approach adopted for the GUM implementation and whether it agrees with the results of the experiment, assuming the particular conceptual mappings modelled. Our initial hypothesis was that the model would be able to replicate the results of the experiment as published concerning the link between device/task-errors and load. The initial aim was thus to provide evidence to further validate the model against these new results, and if it could not, to investigate how the model needed to be improved to capture the underlying behaviour.

Finally we consider the potential of this model-checking approach to generate both suggestions for further refinement of the model, and to motivate further empirical investigations.

The contributions of this paper are essentially two-fold. We explore the results of validating the generic user model on an independent set of experiments not directly designed for this validation. We also demonstrate how the use of formal methods, and model checking in particular, can generate ideas for model refinement and further experimental investigation.

1.4 Overview of the paper

Here we give a brief overview of the rest of the paper. We start by describing the experiment used for validation (Section 2) as well as the key concepts of the generic user model (Section 3). We then, in Section 4, specify the way in which we interpreted the experimental conditions outlined in Ament *et al*. [ACBB10]. In particular, we note the assumptions made as part of this interpretation process. In Section 5, we compare the behaviour of the model with the findings in Ament *et al*. [ACBB10], and show that the model demonstrates good agreement with the experiment in terms of distinguishing between task and device errors, as well as distinguishing between initialisation and post completion errors (a sub-categorisation of device errors). We then show, based on previously unpublished results from the experiment, that the model does not however predict initialisation errors correctly at the level of the individual steps. In Section 6, model checking is then used to further explore the discrepancies to understand the reasons for the observed model behaviour, leading to suggestions for further experiments. Finally, we draw conclusions in Section 7.

2 The Doughnut Machine Experiment

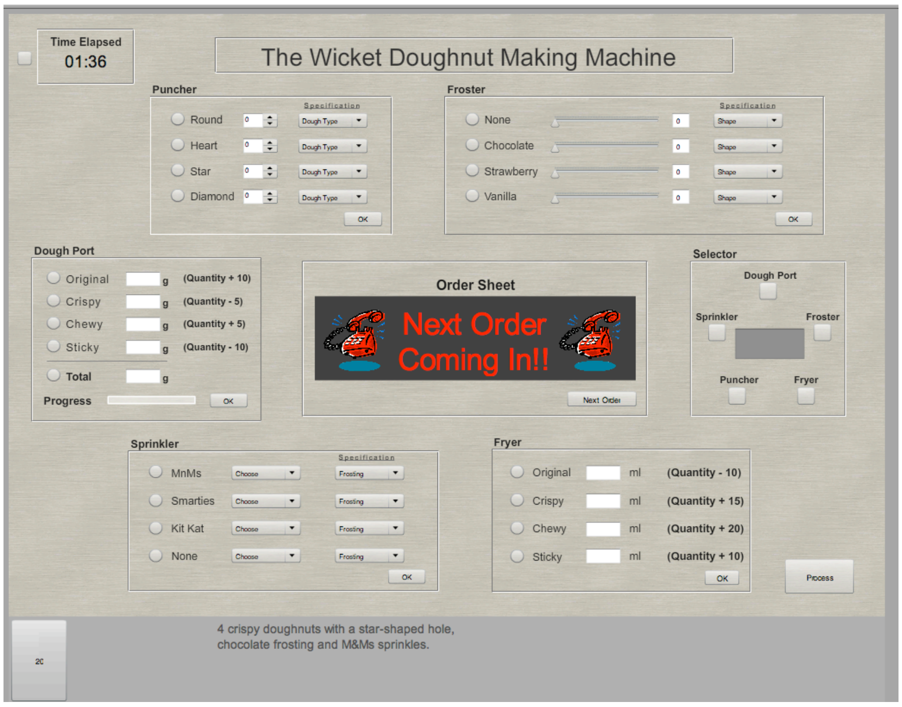
We first present a brief overview of the experiment used in this validation. Due to space constraints we only include the minimum details for understanding the rest of this paper. Interested readers should refer to Ament *et al*. [ACBB10] for more details.

They investigated how working memory load effects task and device-specific actions by using a doughnut-making task. In their paper, a *task-specific* action is defined as one that is central to the task. In particular it is an action that is perceived to move the participant closer to the completion of the main task. In contrast, a *device-specific* action is defined as specific to a particular device, and is not typically common to different devices used for carrying out the same task; Device actions are *not* perceived to move participants closer to main-task completion.

There were two independent variables in the experiment, *memory load* which can take values of high or low, and *type of action* which can be either device or task specific. The four combinations of these two independent variables affected the dependent variable – which was the rate of error for each of the different steps in the action sequence. Memory load was varied between participants, and the step-type, either device or task, was varied within participants. Each step in the sequence was pre-defined as either a task or device action. On the next page, a screenshot of the main interface is given in Figure 2.1, with the corresponding hierarchical task decomposition of the correct sequence of actions for the doughnut task given in Figure 2.2.

Figure 2.2 also indicates the distinction between task and device specific actions used in the experiment. With the coloured rectangles and arrows (6 in all) denoting the steps of the task which were designated *device* actions, with the rest designated as *task* actions. The device actions consist of the five initialisation steps (i.e. steps 2.2.1.1.1, 2.2.2.1.1, 2.2.3.1.1, 2.2.4.1.1, 2.2.5.1.1) and a final post-completion step (2.4.1). In this figure, rectangles represent actual actions taken on the device. To successfully complete the task of making the doughnut, the user had to work through the five data entry areas in the prescribed order, shown by subtasks 2.2.1 to 2.2.5 - with similar sub-steps for each (corresponding to the five outer areas in Figure 2.1, with the user having to select the appropriate radio button from the 'selector' area on the right hand side of Figure 2.1 before being able to enter the relevant data for each of these subtasks each time). Apart from step 2.4.1 (click 'clean'), mistakes at any of the other steps were immediately pointed out and corrected by the experimenter.

The results from the experiments suggest that error rates for device actions are significantly higher than for task actions under both load conditions. Additionally, while error rates on the task-steps *remained low* under a high memory load, the error-rates for device-steps increased significantly under a higher memory load - as shown in Figure 2.3.



4 crispy doughnuts with a star-shaped hole, chocolate frosting and M&Ms sprinkles.

Figure 2.1: The main interface of the doughnut machine.

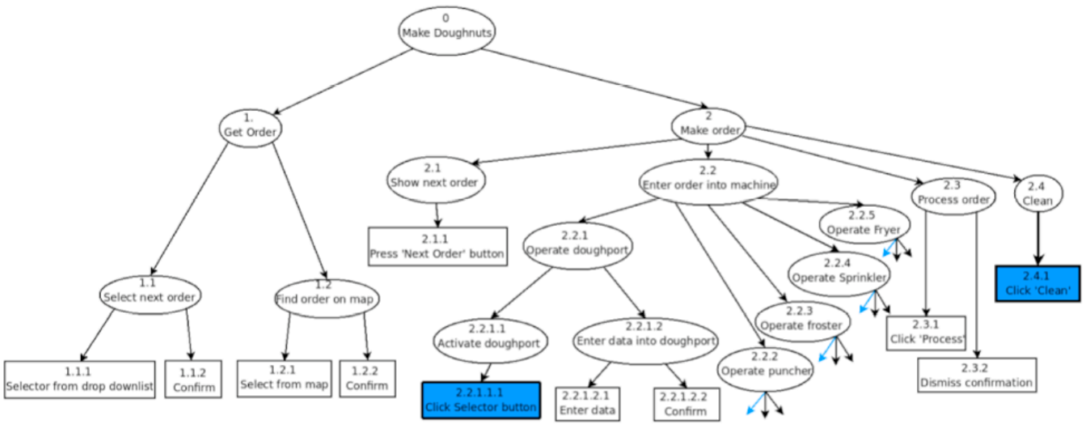


Figure 2.2: A hierarchical task decomposition of the actions involved in fulfilling an order (taken from Ament *et al.* [ACBB10]).

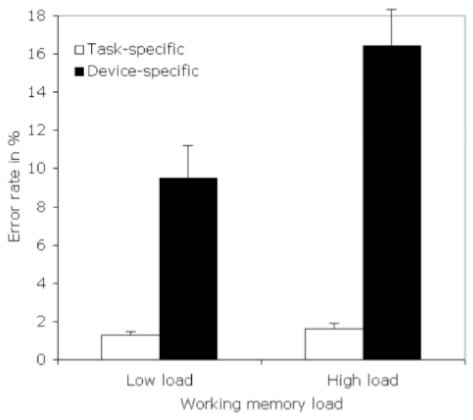


Figure 2.3: Error rates across different working memory load and type of step conditions. Error bars represent the standard error of the mean (taken from Ament *et al*. [ACBB10]).

3 The Generic User Model – Key Concepts

In this section we present a short overview of the key concepts in the generic user model of Ruksenas *et al*. [RBCB09]. The idea behind this approach is to try to encapsulate cognitive principles that are common to all device interactions within a generic parameterisable framework, which can be instantiated for particular scenarios. The ultimate aim is to create a predictive model that enables us to reason about plausible user behaviour with any arbitrary device.

The generic user model consists of five principal parameters, as described below.

Procedural cueing: This parameter primarily deals with the idea of habitual cueing, i.e. that of a particular action following from the previous one. It also includes other kinds of unconscious or 'learnt' sequences of behaviour due to past experience. In general, this kind of cue could be either due to prior instruction from a third party, and/or due to past usage or experimentation by the user; it deals primarily with the (unconscious) sequential ordering of task-steps.

Cognitive cueing: This parameter deals with actions that spring to mind due to the importance of the action for successful task completion. For example, an action central to the successful completion of the main task would typically have a relatively high cognitive cue.

Sensory cueing: This parameter represents cueing derived from sensory sources. It currently represents the combined influence of all of the five senses. Examples of relevant sensory cues include visual interface aspects such as using colour to highlight particular actions (like a big red button for aborting the task for example), or the use of sound to draw attention to an action.

Intrinsic load: This parameter is related to the inherent 'difficulty' of a task or action. For example, doing difficult mathematical calculations as a task-step would have a much higher intrinsic load than multiple-choice check boxes.

Extraneous load: This parameter covers 'external' influences of the context in which the task is taking place. This includes concepts relating to memory load, and other interfering aspects of the external environment, such as from a visually or acoustically complex environment.

Salience

To determine the next set of actions taken by the model at some specific point in an unfolding scenario, the five 'raw' parameters outlined previously are 'summed' via a production-rule system to derive an associated salience value for each of the potential next user-actions. For details of the actual rules see Curzon *et al.* [CRB10]. Figure 3.1 shows the functional relationship between these cues and loads, and the overall salience. For example, in the current model, procedural salience is affected only by procedural cueing and the intrinsic load of a task or action. Intermediate 'grouping' concepts (*procedural*, *cognitive* and *sensory salience*) give a way for the model to be at a relatively high level of abstraction. It also improves the ease with which the model may be understood conceptually. In addition, these three intermediate concepts are used as a means for the model to be context-sensitive.

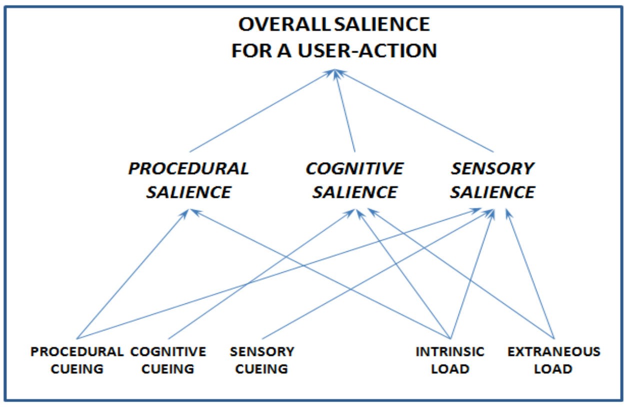


Figure 3.1: How the 'raw' parameters influence the overall salience of a user-action, via the three intermediate concepts. The arrows show the direction of influence.

4 Interpreting the Experimental Conditions in terms of the GUM

In this section, the way in which we interpreted the experimental conditions is described. We discuss the rationale behind our particular interpretation, as well as stating the assumptions made.

4.1 Concept mapping of the five principal parameters

In order to carry out a detailed comparison between the experimental data and the GUM instantiation, it was necessary to decide precisely how to map concepts from the experiment to our instantiation. We now present in detail our interpretation of the concepts and experimental context discussed in Ament *et al.* [ACBB10]. Note that these five principal parameters in the GUM are currently assigned values from a *binary* range.

Procedural cueing was assumed to be initially present (i.e. set with a 'high' value) between all pairs of actions in the prescribed ('correct') action sequence only. This reflected the fact that each of the users had prior instruction, as well as a period of guided exploration before the actual experiment. Data from a small number of users was also excluded from the analysis presented in Ament *et al.* because they had not learnt the task well enough, reinforcing the fact that those whose data was used *were* indeed trained to follow the correct sequence. In both the paper and the modelling presented here, we are interested only in slip errors with an identifiable underlying cognitive cause, not ones occurring as a result of lack of knowledge or skill. All other pairs of actions were assigned 'low' procedural cueing values.

The task/device-action distinction of Ament *et al.* was represented by the *cognitive cueing* parameter of the generic user model. A weak value was given to the device-specific actions whilst a strong one was given to task-specific actions; actions relating specifically to the task at hand were deemed to have a higher cognitive salience than ones that are only incidental actions required by the specific device. We mapped this concept according to their task/device classification - observed in the hierarchical task decomposition presented in Figure 2.2.

Due to the relatively bland nature of the doughnut machine interface, the values set for *sensory cueing* were based essentially on only positioning and size. Only two kinds of actions were defined with strong sensory cues. The steps that involved filling in data (Steps 2.2.1.2.1 in Figure 2.2, for example) were given strong sensory cueing due to their relatively large size. We, and Ament *et al*. both group the individual interface elements where the data was actually entered holistically into one atomic action. Secondly, the final confirmation step (after the doughnuts were made) was given high sensory cueing, as this was a typical popup modal dialog box, which allowed the user to proceed with other actions *only* after its dismissal. All other actions were defined with a weak sensory cue.

For *intrinsic load*, only the subtasks involving some mental arithmetic were assigned a high load value. This was due to the relatively more complex data entry steps within the dough-port and fryer-port subtasks (see Figure 2.1). All other subtasks were defined with a low intrinsic load.

The variation in memory load was reflected in the model instantiation by varying the values for *extraneous load*. A high value (rather than a low one) corresponded to the high memory-load situation, where the user was required to actively monitor additional secondary information fading in and out on the horizontal panel near the bottom of the screen whilst carrying out the main task (see bottom of Figure 2.1).

4.2 Concept mapping – task grouping

The GUM provides a facility for grouping individual user-goals into larger collections of mini-tasks. For our instantiation, the three steps taken within each port (i.e. activate / fill / confirm) were grouped into mini-tasks, one for each of the five ports. The remaining user actions were grouped into a sixth collection. These groupings have some influence on the selection process for the next set of actions at each step – mainly by affecting cognitive cue levels. However this is not a major aspect of the model. Although we had decided on the groupings reasonably independently, the configuration eventually chosen do roughly match the hierarchical task-decomposition presented in Figure 2.2.

4.3 Other assumptions

Several other assumptions were made. We modelled the system as though there was no potential for errors or mismatches relating to the user's direct perception of the *effects of visible device actions*. We also assumed that there were no misinterpretations in their perception of the *effects of their actions on the device*. This does not rule out *all* misunderstandings of a user, it simply means that we model that when they press a button, that button is always actually pressed according to expectations, and perceived as such. However, the user model can still potentially end up with 'misconceptions' relating to the *internal state* of the device. The assumption about the user's perception was deemed appropriate because the doughnut machine is quite a simple device and the participants *were* trained in its use, so there is unlikely to be misunderstandings about the role of each interface element.

A further assumption was that once defined for the instantiation, the particular pattern of parameter assignments would not be subject to arbitrary dynamic alterations. This reflects the fact that the experiment was carried out under strictly controlled circumstances – so there were no unexpected mid-experiment perturbations to the environment, caused by events such as disruptions etc.

We also assumed that in general, users would try to make full use of what few interface cues were available. We therefore took a strictly minimalist position with respect to the necessity of adding extra 'memory variables' for the user model, assuming no additional complexity regarding user-memory unless absolutely necessary.

5 Results

In this section, an analysis of the behaviour obtained with this instantiation of the GUM is presented. We start in Section 5.1 by comparing with the experimental data in terms of task and device-specific actions only, and determine whether the results of the model match the conclusions of Ament *et al*. [ACBB10]. In Section 5.2 we look in more detail at the two specific kinds of device errors observed, notably initialisation and post-completion errors. This is followed by a more detailed step-by-step analysis in Section 5.3 comparing the results from the model against more detailed, previously unpublished data from the experiment. This allowed a more fine-grained analysis of the model's performance. All results presented here were obtained by following the interpretation of experimental concepts and conditions outlined in Section 4.

5.1 Task versus device errors

The original research question we set out to answer was whether our model replicates the behaviour as exemplified in the conclusions of Ament *et al*. [ACBB10]. Their primary conclusion was that error rates were significantly higher for device actions than task actions. We address this first.

The GUM is intended to predict only errors that are *systematic*, i.e. errors that have some underlying cognitive cause, and are not random, one-off errors. Furthermore the model is an abstraction of the actual underlying causes. It therefore works at a certain level of detail. The idea is for the model to highlight plausible situations where design error is likely to be the cause. It is essentially a binary threshold model however, so does not rank erroneous actions in terms of probabilistic estimates of likelihood. Rather, either the model *can* make an error in a given situation or it *cannot*. Ament *et al*. [ACBB10] found that under both memory load conditions, *device errors* were significantly higher than task errors. Therefore the model ought to (in the first instance) be able to correctly predict that these errors are possible in systems such as the doughnut machine.

The results of the model checking are summarised in Table 5.1. We can see that the model makes device errors under both load conditions and makes task errors in neither. Thus it predicts that device errors will be made, and predicts that a significant number of task step errors will *not* be made. This matches the main conclusion of Ament *et al*. [ACBB10]. The threshold nature of the model means that it does not make the further distinction as drawn by Ament *et al*. that a higher memory load leads to *more* errors. Whatever the load, devices errors are made, so a predictive tool needs to highlight this. For a real system (rather than an experimental one as in this case), the system design ought to then be fixed accordingly.

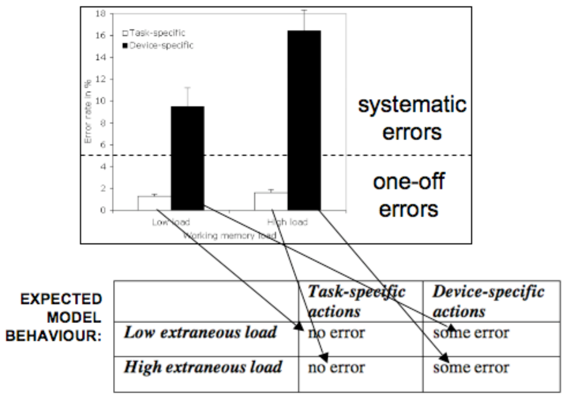


Figure 5.1: Our interpretation of the behaviour to expect from the model.

	Task-specific actions	Device-specific actions
Low load	**No error made by model** √	**Some errors made by model** √
High load	**No error made by model** √	**Some errors made by model** √

Table 5.1: Comparison of the model's behaviour with the experimental results
(√ indicates model behaviour matches experimental result)

5.2 Initialisation errors and post completion errors

The previous section shows that our model can indeed capture the device and task distinction, and make the appropriate predictions at this level of granularity. However, this categorisation potentially groups together a number of different kinds of errors. Those errors are actually made due to *different* rules of the model activating. In particular, both initialisation errors and post completion errors fall under the grouping of 'device-specific' errors. Ideally, as well as predicting the potential for error, the model should suggest how the system design might be fixed. It is therefore important not just that the model can match the results in general, but that it is able to also accurately predict the particular kind of device errors made.

We therefore now focus on a more fine-grained analysis with respect to the specific kinds of device errors made, to gain a more detailed understanding of the validity of the behaviour of our model. The device errors identified by Ament *et al.* [ACBB10] compose of post completion, as well as initialisation steps. They put the overall error rate for initialisation steps at just over 27%; the post completion error rate was also found to be high at over 21%. Ideally the model should therefore predict both post completion errors and initialisation errors, not just one or the other, allowing them both to be predicted and fixed.

	Post-completion step	Initialisation steps
All load conditions	**Error made by model** √	**Some errors made by model** √

Table 5.2: The model's behaviour for post completion errors and initialisation errors
(√ indicates model behaviour matches experimental result)

As indicated in Table 5.2 the model does predict both that post completion error and initialisation errors will be made. Thus at the level of the kind of device error made, the model's behaviour does match the behaviour as seen in the experimental data. However, as will be discussed in the next section, the model does not predict initialisation errors in detail as might be expected. Further work is needed to investigate the underlying causes of those errors and how they may be modelled. It is also important to note here that what is meant by an 'initialisation' step in Table 5.2 is any one of the five selection steps before filling in the data for each of the five areas shown in Figure 2.1. This is the same concept as the 'selector errors' described in Hiltz *et al*. [HBB10].

5.3 A step-by-step analysis

Whilst the previous results show the model does match the experiment at the level of device-error types, there are actually several points in the experiment for initialisation errors to be made. In the paper a distinction is made for the error rate for the first initialisation step (with an error rate of about 27% and the later ones with a combined error rate of about 7%). This does not correspond to the pattern predicted by the model checker as we now discuss. The analysis in this section is based on previously unpublished data from the experiment that gives error rates for each individual step in the Doughnut scenario.

Whilst conducting this more detailed analysis, it was observed that the model demonstrated the same pattern of behaviour under both low and high extraneous load, given the particular way in which we interpreted the experimental conditions for our model. We therefore do not further consider the effect of load here. More detailed investigation of this is left for further work.

Figure 5.2 gives the step-by-step error rates, showing the correspondence between the steps and the specific actions available on the device as given in the hierarchical task-decomposition of Figure 2.2. The distinction between device and task errors here is that adopted by Ament *et al*. [ACBB10]. The data available groups some steps. For example the four individual actions in the task hierarchy relating to obtaining a new order are grouped together (i.e. subtask 1 in Figure 5.2).

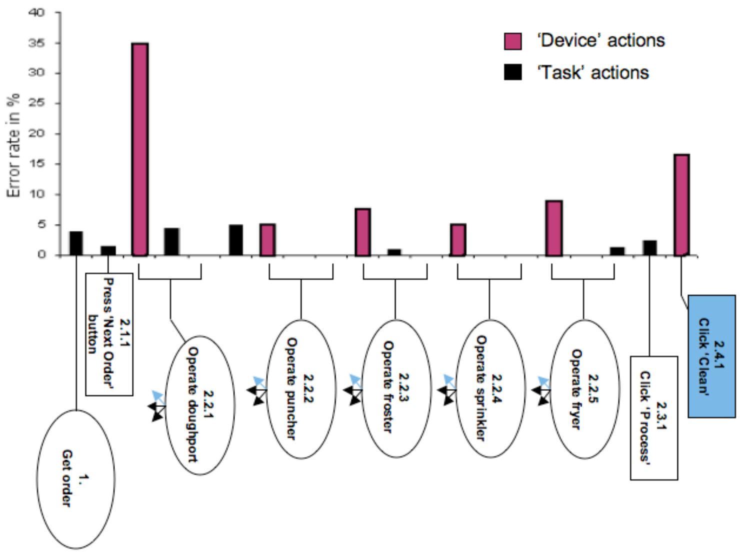


Figure 5.2: Step-by-step error rates for the task.

This shows a much more nuanced picture of the position and magnitude of initialisation errors. All task errors have very low error rates largely justifying the results of the previous sections. Three goal steps are, while still low, noticeably higher. The first step to get the order for example is just below 5%. However, this may be explained by the fact that it combines four steps. The step to enter the data when doing the ‘Operate doughport’ part of the task also has an error rate of around 5% as does the step between subtasks 2.2.1 and 2.2.2 (this step is simply being patient enough to wait for a progress bar to fill).

Looking at the initialisation errors a more intriguing pattern emerges. There are 5 opportunities for initialisation errors. However the errors are not spread evenly between them. The initialisation step for the first ‘Operate doughport’ activity is much higher than the others with a 35% error rate. The other initialisation steps all do have errors but these are at much lower levels of between 5% and 10%. Thus in some cases the error rate is barely above the level that might be expected if the errors were stochastic. Clearly, from error rates alone, there is little to distinguish between the later initialisation steps (which the model predicts to be error prone) and the other task based steps (which the model does not predict to be error prone). Furthermore the model, with the settings we gave, only predicts initialisation errors on the first step of subtask 2.2.2. It does not predict errors on the first or later initialisation steps despite the first initialisation step having an extremely large error rate in the experiment.

This suggests either problems in the model or our understanding of the experimental conditions and how they should be mapped to our model. The model ought to at least predict that the first initialisation step is error-prone, and possibly the later ones too. Given the discrepancy with the data in Figure 5.2, perhaps the model needs a more sophisticated mechanism to determine *when* initialisation errors occur. Further investigation is needed to better understand the underlying causes of these error steps.

We use the formal user model to explore these issues in more detail in the next section.

6 Further Exploration of the Results

In this section, we further explore the formal model and its relation to the experimental results. The particular examples investigated were motivated by the mismatch in behaviour between model and experimental data, as described at the end of the previous section. We deal first with the mismatch at the second of the five initialisation steps (Section 6.1), and then investigate the mismatch at the first of these steps (Section 6.2). Finally we discuss some issues raised by this extended investigation (Section 6.3).

6.1 Presence of errors at the activate-puncher step but not at the other initialisation steps

First we investigate the reasons behind the model making an omission error at the activate-puncher step (the ‘Operate puncher’ activity’s initialisation step) but not at the other initialisation steps. The activate-puncher step corresponds to the first step of Subtask 2.2.2 in Figure 2.2. Some errors were made at all initialisation steps but mostly at a low level, suggesting that the underlying causes for an error here were weak compared to the first initialisation step, at least within the conditions of the experiment as modelled.

We suspected that the reason that the model demonstrated this behaviour was due to the high level of intrinsic load at the previous step prior to the activate puncher step – i.e. when the confirm-doughport step was executed by the user. The previous subtask was set to a high intrinsic load as the user had to do a series of simple, but different arithmetical calculations to determine the values to actually enter. It was not entirely clear at the outset whether this was complex enough to be considered as a high load or not, though in the original instantiation of the scenario we decided to set these kinds of steps to high. Intrinsic load for the other subtasks were low.

Giving the previous subtask a high intrinsic load caused a reduction in the procedural cueing for the ‘activate-puncher’ step, and therefore reduced the overall salience of this initialisation step. This meant that both the ‘activate’ and ‘fill’ steps of the puncher port ended up with the same level of overall salience in the model. As such they were both plausible actions for the model to choose next at this point of the doughnut task. This understanding was confirmed formally by using Linear Temporal Logic properties to model-check a slightly adjusted instantiation. The only change from the initial interpretation of settings was the assignment of a low rather than a high intrinsic load value to the subtask prior to activating the puncher-port. As expected, there were now no errors for this step of the model under the new setting. This same model behaviour was observed under both high and low extraneous loads.

6.2 No error at the activate-doughport step

The second, and more major issue given that it had by far the highest error rate, is why the model did not predict errors at the activate-doughport step. This was the very first sub-activity initialisation step (the first step of Subtask 2.2.1). This seemed likely to be due to the strong influence of procedural cueing upon whether an action is considered plausible or not when, as in this scenario, the sensory cueing for most cues is set to neutral values. In this case sensory cueing effectively did not contribute towards determining model behaviour.

When we verified this formally by removing the procedural cueing between the initial button-press to get the next order, and the activate-doughport step, we found that the model demonstrated the same kind of behaviour as initially seen for the activate-puncher step (again under both high and low extraneous load, refer to Section 6.1 for details). Without procedural cueing the model predicts the potential for an omission error at this step. This would bring the model-behaviour into closer correspondence with the experimental results.

6.3 Discussion

From the more detailed results described in Sections 6.1 and 6.2, we see that the model does not exactly match the experimental data with respect to initialisation errors with the initial settings we chose, though the model could have done so with different settings.

In the experiment initialisation errors were made at every initialisation step, though the error rates differed. Except at the first, rates were relatively low. The original experiment was not explicitly set up to explore the causes behind differing error rates at the initialisation steps of the activities, so this was an unexpected result. Given the low error rates, further experimentation explicitly designed to determine the causes of such errors is needed. The model checking work however suggests interesting areas for further work.

The most obvious failure of the model checking was that it did not predict an error on the first initialisation step when in fact the error rate at this point was extremely high. Further exploration and verification of the model shows that by removing the procedural cueing between each of the major activities (as indicated in the task hierarchy) the model would predict initialisation errors at each step. If the task hierarchy is an arbitrary structure imposed as a way to describe the task, there doesn’t seem to be any strong motivation to do this instead of the original interpretation of a single procedural chain for the prescribed sequence. However, the fact that errors are made at each step, even if at a low level for most, suggests that the way a person mentally breaks a task into activities as guided by training and/or the interface may play a role. Including procedural cueing only within major activities according to likely conceptual models of users would be one solution.

However, there is clearly something distinctly different about the first activation step compared with the other four activation steps. There is clearly *some* kind of discrepancy here. Amongst other possibilities, it could mean there is some sort of concept representing *inter-activity cueing* that is not expressed in either the current model or the experimental concepts presented in Ament *et al.* [ACBB10]. In fact, in parallel to the work described here, we have been developing a more nuanced version of the GUM based around activities and resources, where a new kind of cueing between activities, weaker than the procedural cueing within activities, is explicitly modelled. This will allow us to explore the potential for such a mechanism to model these results. Further experimental evidence to give more detailed insight into the precise scope and nature of the phenomena observed is needed too.

As the model checking shows, the model predicts initialisation errors when the previous activity was of high intrinsic load, but not, all else being equal, when the previous load was low. An issue was how complex the activity needed to be for the load to trigger initialisation errors. For this kind of modelling to be useful in a purely predictive sense, guidance on how to make such decisions would be needed. This suggests an alternative way to use the model, however. Where one could take a scenario-based approach – modelling various load levels and exploring, for a given design, the effects of those different load levels on the errors that occur in the model. This could suggest crunch points where designers should aim to keep intrinsic load levels low, or the design changed to avoid offering device-specific actions at those points.

The results of this study suggest that the abstraction needed for modelling of intrinsic loads of various actions is perhaps not quite as straightforward as thought from previous empirical data. Experiments to investigate this kind of load in more detail and its interaction with procedural cueing are needed to clarify its effect in a wider variety of situations. In addition, for the model having a high (instead of low) intrinsic load at a prior step *with* procedural cueing from it has the same effect as having a low intrinsic load *without* procedural cueing. It would be interesting to find out whether this correlation does indeed hold in reality. Further experimentation is also needed to explore the causes behind the relatively small number of task step errors. If these also had a common cognitive cause that could be determined, then the model could be expanded accordingly to also take those causes into account.

7 Conclusions

We have presented a study that aimed to further validate an existing generic user model, based on new experimental data that was not related to the original development of the model. We also aimed to investigate how model checking can help explore issues that arise, helping to suggest areas for further empirical data collection.

The work in this paper suggests that the generic user model is conceptually complete with respect to the experiment as presented in Ament *et al.* [ACBB10]. We were able to map naturally all of the concepts presented in the empirical investigation into concepts in the GUM. There were no ‘leftover concepts’ from the empirical experiment that needed to be somehow ‘fitted’ to a concept in the model in an awkward or superficial way.

Secondly, taking the initial interpretation of the experimental settings, we arrive at a situation where the model also demonstrates a definite distinction between task and action steps. In terms of the GUM, we obtain errors on device-specific actions and no errors for task-specific actions. In terms of the empirical experiments, we see a result that demonstrates both the predominance of device errors over task-specific ones, as well as a greatly increased sensitivity to memory load for device-specific error rates. The presence of a clear difference between task and device-actions in both cases provides some positive evidence for the general approach assumed for the GUM, and also gives some additional justification for the utility of classifying user-actions according to *task* and *device-specific* actions.

More detailed analysis however suggests that some concepts are perhaps missing from the description in Ament *et al.* [ACBB10] and the generic user model. In particular, the specific pattern of errors observed on a step-by-step basis, especially with respect to initialisation errors, is apparently subtler than currently expressible using only concepts from the experiment.

Further Work

The results from our study indicate that it would be useful to further investigate the precise relationship between intrinsic load and procedural cueing. In particular, in the generic user model they have opposite effects. The same local space of plausible actions could be achieved either by inhibiting the effect of procedural cueing from the previous step (with a high intrinsic load from the previous step), or by simply having no procedural cueing at all at that point in the action sequence. Further collaboration between experimentalists and modellers to investigate this issue is needed. For example, interesting plausible parameter sets could be determined initially by experimenting with the generic user model, and then used for further investigation via empirical experimentation. Alternatively, if the experimenters were unclear about whether to investigate some particular value of a parameter, the GUM could be used to offer some rational suggestions about whether those values are likely to have a significant effect on the overall user-behaviour.

While the classification of actions into task and device-specific types is relatively uncontested in this paper, the particular choice of assignments to be used still depends very much on context. For example, Hiltz *et al.* [HBB10] explores how giving them an alternative initial brief can change people's perceptions about which actions are task-orientated and which are more device-orientated. Instead of talking about making doughnuts as their main task, another group of subjects were told that their main job was to instead *test* the virtual doughnut machine. As a result, these participants demonstrated significantly less errors on the initialisation steps compared with the group operating under the original 'making a doughnut' brief, most probably due to their reconceptualising of previous 'device-specific' steps now as task-specific. A potentially very useful follow-up to the study presented in our paper would be to investigate the degree of agreement between model and data under these two alternative perceptual schemas based on concepts and data from Hiltz *et al.* [HBB10].

In summary, this paper has investigated the validity of an existing generic formal model of cognitively plausible behaviour against data from an independent experiment that was not designed with our model in mind. We have shown that the model does sufficiently capture the concepts described in Ament *et al.* [ACBB10], and agrees with its conclusions with respect to device and task errors in general. At a more detailed level however, some behaviours demonstrated by the model do not match the experimental results regarding initialisation errors and the effect of procedural cueing and load. The resulting analysis suggests that additional concepts are needed in the model, as well as further experimental work to determine more precisely the fundamental principles behind the behaviours observed. Our work also suggests that model checking based on a formal model of cognitively plausible behaviour may help both to explore the results from empirical investigations, as well as generate further research questions of interest.

Acknowledgements

This work is supported by the EPSRC grants on ‘Extreme Reasoning’ (EP/F02309X/1) and ‘CHI+MED: Safer Medical Devices’ (EP/G059063/1).

References

[ACBB10] M. Ament, A. Cox, A. Blandford, D. Brumby. *Working Memory Load Affects Device-Specific but not Task-Specific Error Rates*. In Ohlsson and (eds.) Catrambone (eds.), CogSci10, 32nd Annual Conference of the Cognitive Science Society. pp 91–96, 2010.

[BB97] M. D. Byrne, S. Bovair. *A Working Memory Model of a Common Procedural Error*. Cognitive Science 21(1):31–61, 1997.

[BBD00] R. Butterworth, A. Blandford, D. Duke. *Demonstrating the cognitive plausibility of interactive system specifications*. Formal Aspects of Computing 12(4):237–259, 2000. http://eprints.ucl.ac.uk/5127/(accessed 22-9-2011)

[CRB10] P. Curzon, R. Rukšėnas, J. Back. *The HUM generic user model: An informal overview of the main features*. CHI+MED Working Paper no. 9, 2010. http://dms.chi-med.ac.uk/knowledgetree/browse.php?fFolderId=30(accessed 22-9-2011)

[CY00] A. L. Cox, R. M. Young. *Device-Oriented and Task-Oriented Exploratory Learning of Interactive Devices*. In Proceedings of ICCM 2000: Third International Conference on Cognitive Modeling. pp 70–77, Universal Press, 2000.

[DBDM98] D. Duke, P. Barnard, D. Duce, J. May. *Syndetic modelling*. Human-Computer Interaction 13(4):337–393, 1998. DOI: 10.1207/s15327051hci1304_1

[DBMD95] D. Duke, P. Barnard, J. May, D. Duce. *Systematic Development of the Human Interface*. Second Asia-Pacific Software Engineering Conference, pp 313-321, Computer Society Press, 1995. DOI: 10.1109/APSEC.1995.496980

[DD99] D. Duke, D. Duce. *The Formalization of a Cognitive Architecture and its Application to Reasoning About Human Computer Interaction*. Formal Aspects of Computing 11(6):665-689, 1999. http://dblp.uni-trier.de/db/journals/fac/fac11.html#DukeD99(accessed 22-9-2011)

[Fie01] R. E. Fields. *Analysis of Erroneous Actions in the Design of Critical Systems*. PhD thesis, University of York, York, 2001.

[HBB10] K. Hiltz, J. Back, A. Blandford. *The roles of conceptual device models and user goals in avoiding device initialization errors*. Interacting with Computers 22(5):363-374, 2010. DOI: 10.1016/j.intcom.2010.01.001

[Hol05] E. Hollnagel. *The Elusiveness of ‘Human Error’*. Technical report, based on Hollnagel, E. & Amalberti, R. The Emperors New Clothes, or whatever happened to “human error”? Invited keynote at the 4th International Workshop on Human Error, Safety and System Development, 2005. http://www.ida.liu.se/~eriho/HumanError_M.htm(accessed 22-9-2011)

[KGEM96] S. S. Kirschenbaum, W. D. Gray, B. D. Ehret, S. L. Miller. *When using the tool interferes with doing the task*. In Conference Companion on Human Factors in Computing Systems: Common Ground. CHI ’96, pp. 203–204. ACM, New York, USA, 1996. DOI: 10.1145/257089.257281

[MD95] T. Moher, V. Dirda. *Revising mental models to accommodate expectation failures in human-computer dialogues*. In Design, specification and verification of interactive systems (DSV-IS’95). pp 76-92, 1995.

[MH69] J. McCarthy, P. J. Hayes. *Some Philosophical Problems from the Standpoint of Artificial Intelligence*. In Machine Intelligence. Volume 4, pp 463–502, Edinburgh University Press, 1969. http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.85.5082(accessed 22-9-2011)

[RBCB09] R. Rukšėnas, J. Back, P. Curzon, A. Blandford. *Verification-guided modelling of salience and cognitive load*. Formal Aspects of Computing 21(6):541-569, 2009. DOI: 10.1007/s00165-008-0102-7

[SBB07] L. Su, H. Bowman, P. Barnard. *Attentional capture by meaning: A multi-level modelling study*. In Proceedings of the 29th annual meeting of the cognitive science society (CogSci 2007), Lawrence Erlbaum Associates, NJ, pp 1521-1526, 2007. http://www.cs.kent.ac.uk/pubs/2007/2594(accessed 22-9-2011)

[SBBW09] L. Su, H. Bowman, P. Barnard, B. Wyble. *Process algebraic modelling of attentional capture and human electrophysiology in interactive systems*. Formal Aspects of Computing 21(6):513-539, 2009. DOI: 10.1007/s00165-008-0094-3.