The Role of Goal Relevance in the Occurrence of Systematic Slip Errors in Routine Procedural Tasks

Maartje Gertruda Anna Ament

A dissertation submitted in partial fulfilment
of the requirements for the degree of
Doctor of Philosophy
of
University College London.

Department of Computer Science
University College London

2011

I, Maartje Gertruda Anna Ament, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, I confirm that this has been indicated in the thesis.

Abstract

Slip errors can have severe consequences but are notoriously difficult to reduce. Training, visual cues and increasing motivation are generally not effective in eliminating these slips. Instead, the approach this work takes is to identify which steps in a routine task are most error prone, so that these can be designed out of device interactions. In particular, device- and task-oriented steps are investigated. Device-oriented steps are "extra" steps imposed by the device that do not directly contribute towards the task goal. Conversely, task-oriented steps directly bring the user closer to their goal. The main hypothesis addressed in this work is that device-oriented steps are more problematic than task-oriented ones.

The concepts of device- and task-oriented steps are investigated more closely, by analysing the literature on routine action and mental representations of different steps. The core difference between the steps is found to be how relevant a step is to the goal. This is further supported by two qualitative studies.

A series of experimental studies investigates the cognitive mechanisms underlying device- and task-oriented steps. This is addressed through six experiments that address error rates, step times, proportion of omissions and sensitivity to working memory load. Participants learned one of three routine tasks, with several carefully controlled device- and task-oriented steps. The results show that on device-oriented steps, error rates are higher, step times are longer, the proportion of omissions is greater, and working memory load has an increased effect. These findings support the hypothesis that activation levels are lower on device-oriented steps.

The thesis concludes that a step's relevance to the task goal plays an important role in the occurrence of errors. This work has implications for both our understanding of routine procedural action as well as the design of devices.

Acknowledgements

I would like to thank the following people:

Anna Cox, my primary supervisor - for providing guidance and advice, for making me do seminars and lab groups, and for teaching me to have confidence in myself and my work

Ann Blandford, my secondary supervisor - for pushing me to do better and reach my full potential, and for always finding the time to help me no matter how busy she is

Duncan Brumby - for taking the time to read my work, for providing a fresh perspective, and for providing inspiration for the Frankenstein task (no not for the monsters :P)

Richard Young - for taking the time to read an earlier draft of this thesis, and providing helpful comments and advice

Rowan - for always being there for me, for lifting me up from the bad and celebrating with me the good, and for being my favourite man in the whole wide world

Papa en Mama - for their continued and loving support, for the unconditional confidence in me and my work, and for teaching me everything that a university education cannot provide

Everyone at UCLIC - for making my PhD time a fun and stimulating experience, and for asking evil questions at the seminars so that my work may improve

Costa coffee @ Waterstone's and @ 55 Baker Street - for making the best lattes and letting me write this thesis in their shop

Paul Cairns - for providing valuable advice on statistical analysis

All participants - for their time and effort

The EPSRC - for providing the funds that enabled me to do this PhD

Contents

1 Introduction **1**

1.1 Motivation 1

1.2 Research Problem 2

1.3 Research Approach and Hypothesis 3

1.4 Overview of the Remaining Chapters 4

2 Correct and Erroneous Execution of Routine Procedural Tasks **5**

2.1 Overview 5

2.2 Defining Human Error 6

2.2.1 What is an Error? 6

2.2.2 Taxonomies of Error 8

2.2.3 Scope 12

2.3 Goals, Tasks and Skilled Procedures 12

2.3.1 Level of Analysis 13

2.3.2 The Building Blocks of Tasks 13

2.3.3 Skilled Tasks 15

2.4 Cognitive Models of Routine Procedural Action 16

2.4.1 The Memory-for-Goals Model 17

2.4.2 Contention Scheduling and Schema Network Models 19

2.4.3 Simple Recurrent Network Model 25

2.5 Experimental Studies of Slip Errors 28

2.5.1 Post-Completion Error 28

2.5.2 Interruptions 30

2.5.3 Working Memory Load 32

2.5.4 Task Structure 33

2.5.5 Slip Errors and Step Times 35

2.6 Error Mechanisms 36

2.7 Conclusion 37

3 Device- and Task-Oriented Actions 39

3.1 Overview . . . 39

3.2 Previous Research . . . 40

3.2.1 Mental Models . . . 40

3.2.2 Contribution to the Task Goal . . . 44

3.2.3 Behavioural Effects . . . 48

3.2.4 Cognitive Modelling . . . 50

3.2.5 Summary and Working Definitions . . . 52

3.3 Qualitative Study . . . 53

3.3.1 Approach . . . 53

3.3.2 Analysis and Results . . . 54

3.3.3 Discussion and Revised Definition . . . 61

3.4 Hypothesis and Experimental Predictions . . . 65

3.4.1 Low Activation Levels . . . 66

3.4.2 High Activation Levels on Incorrect Steps . . . 67

3.4.3 Degraded Context Representations . . . 68

3.5 Conclusion . . . 69

4 Study on the Classification of Device- and Task-Oriented Steps 70

4.1 Overview . . . 70

4.2 The Experimental Tasks . . . 71

4.2.1 The Doughnut Task . . . 71

4.2.2 The Spy Task . . . 75

4.2.3 The Frankenstein Task . . . 78

4.3 Classification Study . . . 78

4.3.1 Introduction . . . 78

4.3.2 Method . . . 80

4.3.3 Results . . . 81

4.3.4 Discussion . . . 82

4.4 Conclusion . . . 85

5 Exploration of the Error Pattern on the Doughnut Task: Experiments 1 and 2 87

5.1 Overview . . . 87

5.2 Experiment 1 . . . 89

5.2.1 Introduction . . . 89

5.2.2 Method . . . 89

5.2.3 Results . . . 91

5.2.4 Discussion . . . 92

5.3 Experiment 2 . . . 95
5.3.1 Introduction . . . 95
5.3.2 Method . . . 96
5.3.3 Results . . . 98
5.3.4 Discussion . . . 100
5.4 General Discussion . . . 101
5.4.1 Evaluation of the Doughnut Task . . . 102
5.5 Conclusion . . . 103

6 Further Investigation of the Cognitive Mechanisms Underlying Device- and Task-Oriented Steps using the Spy Task: Experiments 3 and 4 105
6.1 Overview . . . 105
6.2 Experiment 3 . . . 107
6.2.1 Introduction . . . 107
6.2.2 Method . . . 107
6.2.3 Results . . . 108
6.2.4 Discussion . . . 111
6.3 Experiment 4 . . . 114
6.3.1 Introduction . . . 114
6.3.2 Method . . . 115
6.3.3 Results . . . 117
6.3.4 Discussion . . . 119
6.4 Overall Discussion . . . 121
6.4.1 The Spy task . . . 122
6.5 Conclusion . . . 122

7 Direct Manipulation of Goal Relevance Using the Frankenstein Task: Experiments 5 and 6 124
7.1 Overview . . . 124
7.2 Experiment 5 . . . 125
7.2.1 Method . . . 125
7.2.2 Results . . . 128
7.2.3 Discussion . . . 133
7.3 Experiment 6 . . . 135
7.3.1 Method . . . 136
7.3.2 Results . . . 136
7.3.3 Discussion . . . 139
7.4 Overall Discussion . . . 140
7.4.1 Evaluation of the Frankenstein Task . . . 141

7.5 Conclusion . . . 143

8 General Discussion and Conclusion 144

8.1 Overview . . . 144
8.2 Experimental Findings . . . 145
8.2.1 Error Rates . . . 145
8.2.2 Step Times . . . 146
8.2.3 Type of Error . . . 146
8.2.4 Working Memory Load . . . 147
8.2.5 Correlation Between Device- and Task-Oriented Steps . . . 148
8.2.6 Eye Movements . . . 148
8.3 Activation Levels . . . 149
8.3.1 Error Mechanisms . . . 150
8.3.2 Implications for Cognitive Models . . . 153
8.4 Limitations . . . 157
8.4.1 Limited Duration and Routine Procedural Performance . . . 157
8.4.2 Generalisation to Real-World Tasks . . . 158
8.4.3 Role of Training . . . 158
8.4.4 Task Structure . . . 159
8.4.5 Definition of Omission Errors . . . 160
8.5 Contribution . . . 161
8.5.1 Theoretical Contribution . . . 161
8.5.2 Practical Contribution . . . 162
8.5.3 Methodological Contribution . . . 162
8.6 Directions for Future Research . . . 162
8.6.1 Cognitive Modelling . . . 162
8.6.2 Identification of Device- and Task-Oriented Steps . . . 163
8.6.3 Error Mitigation . . . 165
8.7 Conclusion . . . 165

Bibliography 167

A Instructions given to participants 174

A.1 Classification Study . . . 174
A.2 The Doughnut Task . . . 176
A.3 The Spy Task . . . 179
A.4 The Frankenstein Task . . . 181

B Classifications 190

C Examples of Errors Occurring in Real Life 194

List of Figures

2.1 The components of the IAN model 23

3.1 Enabling States analysis 47

4.1 The Wicket Doughnut Making Machine 72
4.2 The call centre 74
4.3 HTA diagram of the Doughnut task 74
4.4 The Spy task interface 75
4.5 HTA diagram of the Spy task 77
4.6 The Frankenstein task interface 78

5.1 Error patterns on previous studies using the Doughnut task 88
5.2 Error pattern found on the Doughnut task in experiment 1 92
5.3 Number of errors per trial on the Doughnut task 93
5.4 The Doughnut Live Feed 97
5.5 Error rates per trial on the Doughnut task 99
5.6 Error rates across working memory load and type of step conditions 100

6.1 Error rates per trial on the Spy task 109
6.2 Error pattern on Spy task 111
6.3 Example of a hidden step on the Spy task 115

7.1 Error rates per trial on the Frankenstein task 128
7.2 Error rates per step on the Frankenstein task 130
7.3 Step times per step on the Frankenstein task 131
7.4 Proportion of omission errors per step on the Frankenstein task 132
7.5 Error rates on device- and task-oriented steps for high and low load 137
7.6 Step times on device- and task-oriented steps for high and low load 138
7.7 Proportion of omissions on device- and task-oriented steps for high and low load 139

List of Tables

3.1 Task-action mapping for an example calculator 42

4.1 All steps required to complete one trial on the doughnut machine 73
4.2 All steps required to complete the Spy task 76
4.3 All steps required to complete the Frankenstein task 79
4.4 Kappa (agreement) values for all tasks 82

5.1 Error rates on the Doughnut task 99

6.1 Summary of results on the Spy task, experiment 3 109
6.2 Summary of results on the Spy task, experiment 4 118

7.1 Overview of target steps on the Frankenstein task 126
7.2 Error rates on device- and task-oriented steps on the Frankenstein task 129
7.3 Step times on device- and task-oriented steps on the Frankenstein task 130
7.4 Proportion of omission errors on device- and task-oriented steps 131
7.5 Eye movements on device- and task-oriented steps 133

8.1 Overview of experimental predictions and error mechanisms 150

Publications

- Ament, M. G. A., Blandford, A. and Cox, A. L. (2009). Different Cognitive Mechanisms Account for Different Types of Procedural Steps. Presented as an oral presentation at CogSci 09, Amsterdam.
- Janssen, C. P., Young, R. M., Ament, M. G. A., Back, J., Brumby, D. P., Cox, A. L. and Grace, J. (2010). Cognitive Modelling at the UCL Interaction Centre. In Proceedings of the European ACT-R Workshop 2010.
- Ament, M. G. A., Cox, A. L., Blandford, A. and Brumby, D. P. (2010) Working Memory Load Affects Device-Specific but Not Task-Specific Error Rates. Presented as an oral presentation at CogSci '10, Portland, Oregon.
- Ament, M. G. A. (2011) Frankenstein and Human Error: Device-Oriented Steps are More Problematic than Task-Oriented Ones. Presented at CHI '11, Vancouver, BC.

Chapter 1

Introduction

In this chapter:

- The research problem is introduced, and a motivation for the current work is presented.
- The work focusses on identifying which steps in routine procedures are more prone to errors than others, so that these steps can be designed out of interactions.
- The main hypotheses addressed in this thesis are then set out.
- An overview of the remaining chapters is given, in which the hypotheses are tested through a series of experimental studies.

1.1 Motivation

"To Err is Human" is a phrase often used to highlight that human error is persistent and ubiquitous. From forgetting your keys in the morning, to leaving the original on a copy machine after retrieving the copies, their occurrence is disruptive, or, at best, annoying. However, in safety critical situations, such as medicine or nuclear power plants, errors can have far more severe consequences. Casey (1998, 2006) presented a number of case studies of human error which have led to disaster. These anecdotes highlight how often simple and trivial failings can have disastrous consequences, including the loss of life.

The study of human error is important for two reasons. First, from a Human-Computer Interaction (HCI) or Human Factors (HF) perspective, understanding why errors occur allows the development of strategies to reduce their occurrence or their impact. An example of how error research has helped mitigate the occurrence of a particular error comes from the use of ATM machines. An error frequently observed in their use is leaving the bank card behind after retrieving the money. Recent research on this type of error has led to a detailed understanding of its occurrence. As a result, ATMs have been redesigned, with a subsequent reduction in the occurrence of this error. While the complete elimination of human error is unlikely, nor a sensible goal to pursue (Reason, 2002), the rigorous study of errors can identify where they are likely to occur, and what the most effective way to mitigate them is.

Second, understanding why errors occur is also useful from a cognitive perspective. Not only will research on human error help us understand why the cognitive system occasionally breaks down, but also how it operates under normal circumstances. For instance, the occasional but persistent nature of certain errors has provided support for the idea that memory representations are subject to noise (e.g. Altmann and Trafton, 2002). This noise occasionally leads to the selection of an incorrect memory representation, resulting in an error. Conversely, the development of cognitive models of routine action has allowed researchers to predict when errors are likely to occur (e.g. Cooper and Shallice, 2000). As such, research on human error informs the study of correct task performance and vice versa, making the two closely connected.

1.2 Research Problem

A type of error that has been subject to recent investigation is the slip error. This particularly persistent error is not the result of a lack of knowledge, but instead results from a temporary failure of working memory (Byrne and Bovair, 1997). As such, slip errors occur only occasionally, but they are persistent, even in highly practised routine procedures executed by expert users. Moreover, they are notoriously difficult to reduce. Several attempts to mitigate slip errors have been shown to be ineffective, such as retraining (Byrne and Davis, 2006) or increasing motivation (Back, Cheng, Dann, Curzon and Blandford, 2006), while others, such as visual cues, must be highly aggressive to have an effect (Chung and Byrne, 2008).

Slip errors do not occur at random. Various studies have shown that some steps in a task are more prone to errors than others. An example is the post-completion error, which occurs when the last step in a task is omitted *after* achieving the main goal. This error occurs more often than would be expected by chance alone (Byrne and Bovair, 1997). Indeed, tasks often have a systematic error pattern that reflects at which points in the task errors are most likely to occur (Tsai and Byrne, 2007). An alternative approach to providing visual cues or otherwise trying to combat frequent errors is to design out these error prone steps from the interaction in the first place. Such an approach has been shown to be highly effective for the post-completion error (Byrne and Davis, 2006). Of course, a post-completion task structure is not the only factor that affects error rates. The general aim of the current thesis is to identify further aspects of the task structure that can influence error rates, so that these can be designed out of devices altogether.

More specifically, the current work focusses on one such factor that could explain the error pattern observed on the Doughnut task, a well-known experimental task originally developed by Li (2006). Li et al. (2008) suggested that many of the steps with a particularly high error rate are "extra steps imposed by the device", or in other words they are device-oriented. This means that these error-prone steps are more concerned with the operation of the device, rather than with the achievement of the task goal. Task-oriented steps, on the other hand, are crucial for the achievement of the main task goal. An example of a device-oriented step is selecting a text box before entering text into it: this step helps to operate the device, but does not directly bring the

user closer to their main goal. The action of entering text, on the other hand, is task-oriented, as it makes a direct contribution to the main goal. Taking the error pattern on the Doughnut task as a starting point, the current thesis investigates the idea that device-oriented steps are more problematic than their task-oriented counterparts.

1.3 Research Approach and Hypothesis

While there has been some previous work on concepts similar to device- and task-oriented steps, the definitions and terminology used are inconsistent. As such, a first aim of the current work is to develop a well-grounded definition on which the remainder of the thesis can build. This is achieved through an extensive literature review and a qualitative study with a small number of experts in the field of HCI. While a limited number of studies have suggested that device-oriented steps may be more prone to errors (e.g. Gray, 2000; Hiltz et al., 2010), it is not clear *why* this may be the case. Therefore, a second aim of the current work is to gain an understanding of why device-oriented steps may be more problematic than task-oriented steps. This is achieved through an empirical investigation of the cognitive processes underlying these steps.

The main hypothesis put forward in the current work is that device-oriented steps are more problematic because they have lower activation levels than their task-oriented counterparts. 'Activation' is a theoretical concept often used in models of human memory and routine action, and represents how well a goal, schema or other item in memory is able to compete with others. The more active a memory representation is, the better the chance that it is able to direct behaviour. Conversely, a lower activation level makes this *less* likely, thereby increasing the chance that an error will occur. It is difficult to directly study the activation levels on different task steps, without developing a cognitive model. Nevertheless, the hypothesis of this thesis that activation levels are lower on device-oriented steps leads to a number of testable predictions:

- Error rates on device-oriented steps are expected to be higher than those on task-oriented steps.

- It is expected that it takes longer to execute a device-oriented step than a task-oriented step.

- There should be a higher proportion of omission errors (forgetting a single step) on device-oriented steps.

- A high working memory load is expected to have a greater effect on device-oriented than on task-oriented ones.

These predictions are tested through a series of empirical studies. The findings from these experiments provide support for the main hypothesis that device-oriented steps are more problematic because they have lower activation levels.

1.4 Overview of the Remaining Chapters

The next chapter (chapter 2) presents a general overview of previous research on human error and routine procedural action. It defines the scope of the current thesis, and sets out the activation level hypothesis in more detail. Chapter 3 discusses in more detail the distinction between device- and task-oriented steps. First, previous research on similar concepts is discussed, which leads to the development of a working definition. This is then further developed through a small qualitative study using structured interviews with experts in the field of HCI. The definition is validated in chapter 4, which presents a study that classifies the steps in the three experimental tasks used in this thesis. This study serves two functions: to validate the core concepts and definitions, and to validate the experimental tasks.

The thesis then presents the experimental work done. Chapter 5 uses the Doughnut task as a starting point, and presents a study that tests whether the error patterns found previously are robust, and whether a distinction between device- and task-oriented steps can help explain it. It shows that the approach is highly successful on this particular task, but the findings cannot be generalised to others. Chapter 6 addresses the shortcomings of the Doughnut task and directly compares device- and task-oriented steps. It provides some preliminary support for the activation level hypothesis. Chapter 7 presents a novel task that directly manipulates goal relevance by providing two 'cover stories' for a single task interface, allowing interface and step difficulty issues to be accounted for. Two experiments on this task are presented, which provide further support for the main hypothesis.

The final chapter (chapter 8) discusses the results and their implications. It carefully reviews the findings, and interprets their significance for the theoretical models of routine action and error. It also discusses the limitations of the current work. Finally, suggestions for future research and a conclusion are presented.

Take-home message:

Slip errors occur despite the presence of correct knowledge, and are very difficult to mitigate. This thesis aims to identify when these slips are likely to occur, so that error prone situations can be designed out of interactions. Specifically, it tests the hypothesis that device-oriented steps are more problematic than task-oriented steps because they have lower activation levels.

Chapter 2

Correct and Erroneous Execution of Routine Procedural Tasks

In this chapter:

- Errors are defined and several taxonomies are used to scope the thesis.
- The type of task studied is introduced, and several cognitive models of these tasks are discussed.
- Empirical studies on slip errors are examined in light of the cognitive models.
- The main hypothesis of this thesis, and associated predictions, are set out in detail.

2.1 Overview

Despite having the correct knowledge, people still make errors while executing routine procedures. Often, these errors are inconsequential, but in safety-critical domains such as aviation, medicine and nuclear power plant operations they can have severe consequences. The role of human error in accidents should not be underestimated. For example, approximately half of all accidents and incidents in commercial aviation are caused by pilot error (Baker et al., 2008). Casey (1998, 2006) also presented a number of case studies that highlight how relatively minor errors can sometimes combine to produce disastrous consequences.

As a result, researchers have become interested in studying why errors occur, and how they can be avoided. According to Reason (1990), the holy grail of human error research is to be able to predict when an error is going to occur, *before* it actually occurs. Indeed, research on human error has steadily increased over the past decades. Numerous studies have sought to classify (e.g. Norman, 1981), understand (e.g. Gray, 2000), mitigate (e.g. Chung and Byrne, 2008), or predict (e.g. Ratwani et al., 2008) human error, both in the laboratory (e.g. Byrne and Bovair, 1997) and in naturalistic settings (e.g. Rasmussen, 1982).

However, the investigation of human error does not stop at erroneous actions. To fully understand why errors occur, it is important to also understand how tasks are carried out

correctly. As Byrne and Bovair (1997) summarise:

> "In order to understand how errors occur, it is necessary to consider the cognitive mechanisms that govern correct human behavior."

The benefit is mutual: just as understanding how tasks are executed correctly will help us comprehend why they break down, the properties of these errors can inform us about the structure and functioning of the correctly operating system. Take for instance the study of post-completion errors (PCEs). These errors occur when the last step in a task has been forgotten, after the main goal has already been completed. The various studies on these slips have not only taught us that interruptions can increase PCEs, but also that associative links between steps can aid routine performance (Altmann and Trafton, 2002; Li et al., 2008, see section 2.5.2 for a discussion).

In short, to study errors is to study correct behaviour, and vice versa. As such, this thesis begins with a thorough review of the current literature on errors and correct task performance. The objectives of this chapter are to set the context for the current work and to motivate its scope. The chapter commences with a brief overview of early work on human error. It looks at how errors are defined, and discusses a number of taxonomies that seek to classify them. The current work focusses on slip errors in routine procedures, and a motivation for this is provided. The chapter then turns to routine procedural performance. It discusses how tasks are normally executed without errors, and how slip errors can occur. A number of cognitive models are introduced that describe the cognitive processes underlying correct and erroneous performance. For each of these models, the chapter discusses how it represents concepts such as goals, decay and errors. Recent experimental work on slip errors is then discussed, as well as how their findings fit into the various cognitive models. The chapter concludes with a discussion of the hypotheses addressed in the experimental work in this thesis.

2.2 Defining Human Error

The discipline of human error is a relatively young one. While a number of studies in the early 20^{th} century argued for the study of error as a phenomenon by itself (e.g. Spearman, 1928), it was not until the late 1970s that human error started to get the attention it deserves. Much of this early work focussed on defining and classifying errors (e.g. Reason, 1979; Norman, 1981; Rasmussen, 1982). This section examines the various definitions of human error, and then discusses several attempts to develop taxonomies. Finally, these are used to motivate the scope of the current thesis.

2.2.1 What is an Error?

As Senders and Moray (1991) and Rasmussen (1982) argue, there is no single, agreed definition of human error. This may be related to the different disciplines that are involved in studying the topic, such as psychology, human factors and cognitive science, each with a slightly different

viewpoint. In this section, a number of these definitions are reviewed, and a working definition for the current thesis is proposed.

Rasmussen (1982) proposed a general definition of human error:

> "[Errors can be identified when] a system performs less satisfactorily than it normally does - due to a human act or to a disturbance which could have been counteracted by a reasonable human act".

More simply put, he argues that human errors can be seen as "instances of man-machine or man-task misfits", where these misfits are occasional and caused by variability on the part of the human. Rasmussen (1982) used this definition in his classification of performance levels, which can be used to categorise different types of errors (this will be discussed in more detail in section 2.2.2.2).

The definition as proposed by Rasmussen (1982) is relatively general, as 'less satisfactory' performance could range from the lateral deviation of a car towards the lane boundary, to the meltdown of a nuclear power plant. The former may be rather inconsequential, while the latter may result from a succession of errors. As such, this definition may not be helpful in setting the scope of the current work. A more specific definition comes from Reason (1990):

> "Errors [are] all those occasions in which a planned sequence of mental or physical activities fails to achieve its intended outcome, and when these failures cannot be attributed to the intervention of some chance agency."

Thus, Reason (1990) specifies that the error must have some unintended consequence in the real world.

Senders and Moray (1991) reported on the outcomes of a NATO conference that sought to define human error and establish the major themes in the field. They identified a number of attributes common to the various definitions of human error:

1. **Intention of the actor:** the erroneous action is not intended by the actor
2. **Desirability of the outcome:** the performance outcome is not desired by a set of rules or an external observer
3. **Assessment against a certain criterion:** the error leads the task or system outside its acceptable limits

Taking these commonalities into account, Senders and Moray (1991) proposed the following definition:

> "An error occurs when a planned series of actions fails to achieve its desired outcome, and when this failure cannot be attributed to the intervention of some chance occurrence."

The definitions discussed above are closely aligned with the outcome of a task: an error is acknowledged only if the outcome of the task is affected. However, as Rasmussen (1982) argues, this may not always be an appropriate approach. He notes that the outcome of a task is not always immediately observable, such as when the effects of errors are delayed in time, or depend on further steps in a sequence. Moreover, this approach does not take into account those actions which are erroneous, but subsequently corrected. While in this case the overall task outcome may not be affected, the reason for this is that the error was detected. However, the act of correcting an error arguably does not change the fact that the action went wrong in the first place.

This argument is closely related to the approach taken by the various experimental studies of human error. For experimental purposes, errors are often defined as "any incorrect action deviating from the correct sequence" (e.g. Li, 2006). This approach looks at errors at the level of individual actions, and is less concerned with the overall task outcome. While task outcome may be relevant in applied domains such as when assessing human error in aviation, it is less suitable in experimental work (such as presented in this thesis). First, the experimental tasks used often have zero tolerance for incorrect action sequences. This means that tasks generally cannot progress unless an erroneous action has been corrected (e.g. Li et al., 2008; Hiltz et al., 2010). Second, for the purpose of studying the underlying causes of errors, it may make more sense to look at individual actions rather than the overall task outcome. Whether or not the task outcome is successful is not thought to affect the underlying cognitive causes of each particular erroneous step. Therefore, the current thesis adopts Li's (2006) working definition of an error as *any incorrect action that deviates from the correct task sequence.*

2.2.2 Taxonomies of Error

Having established a working definition of human error, the chapter now discusses the different types of errors that can occur. Early work on human error mainly sought to classify errors, and thereby understand what causes them (e.g. Norman, 1981; Rasmussen, 1982; Reason, 1990). However, as Senders and Moray (1991) argue, just as there is no single agreed definition of error, there is also no consensus on taxonomies of errors. Arguably, different taxonomies fulfil different roles, so this is not surprising, nor necessarily problematic. Reason (1990) identified three different types of taxonomies:

- **Behavioural taxonomies:** these classify errors according to their superficial behavioural characteristics.
- **Contextual taxonomies:** these consider the local factors involved in error production.
- **Conceptual taxonomies:** these look at the cognitive mechanisms that underlie errors.

A number of taxonomies will now be discussed in more detail.

2.2.2.1 Behavioural Classifications

One of the earliest taxonomies comes from Kollarits (1937), who classified errors according to their superficial characteristics. He identified four types: substitution, insertion, omission, and repetition errors. These, and similar ones, have also been described in many other studies (e.g. Senders and Moray, 1991; Cooper and Shallice, 2000).

Substitution errors can be thought of as the replacement of one action by another (or one object by another). A classic example is putting shaving cream on a toothbrush, instead of toothpaste. Insertion errors happen when an inappropriate action is erroneously inserted into an action sequence, for instance habitually adding a spoonful of sugar to a cup of tea, while having decided to cut down on sugar. Omission errors are characterised by the omission of a single action, and instead continuing the task with the *next* action in the sequence. An example of this is forgetting to take the lens cap off a camera before taking photos. Repetition errors occur when an action is erroneously repeated. An example of this type of error is boiling water in a kettle twice when making tea. In addition to these, other types of errors have been identified, such as anticipation errors, which are similar to omission errors but are easily resolved (Trafton et al., 2011). An example is trying to pour from a bottle without first removing its cap. Another type is a capture error, in which a lapse occurs from a less frequent task into a similar, more frequent one (Cooper and Shallice, 2000). An example is typing in the password for one's work computer into one's home computer.

It has often been argued that behavioural classifications are problematic and that their use is limited (Kollarits, 1937; Reason, 1990). First, an obvious problem with these classifications is that errors do not always neatly fit into one category, and can be difficult to classify. For instance, a substitution error could be seen as a combination of an omission and an insertion error. Similarly, substitution and capture errors are very similar. Second, behavioural taxonomies generally do not make predictions about the underlying cognitive mechanisms of the error[1], nor do they look at the different contexts in which errors occur. As such, they may group together errors that have very different underlying causes.

In short, behavioural classifications do not greatly contribute to our understanding of human error. As such, the chapter turns to a later taxonomy developed by Rasmussen (1982), who classified errors according to the level of processing in which they occur.

2.2.2.2 Skill-, Rule- and Knowledge-Based Performance

Rasmussen (1982) noted that most attempts to classify errors were based on the task in which the error occurred, rather than on human behaviour. In response, he developed a taxonomy of different levels of human behaviour, and described the errors that can occur on each of these levels:

[1] An exception is Cooper and Shallice's (2000) IAN model (further discussed in section 2.4.2.3). While the purpose of their work is not to provide a taxonomy of errors, it does make predictions about the underlying mechanisms of a number of behavioural error types.

- **Skill-based domain:** this comprises automated, skilled, routine performance. Behaviour on this level generally does not require conscious control, but is governed by stored patterns of implicit, procedural knowledge. Errors in this domain are related to the inappropriate control of actions.

- **Rule-based domain:** this includes performance that is controlled by stored rules, which govern the coordination of (skill-based) subroutines. This domain is concerned with familiar situations, and often requires conscious mental effort in recalling and applying the correct rules. Errors are made when a situation is recognised wrongly, or when the wrong rules are recalled or incorrectly applied.

- **Knowledge-based domain:** this is used when novel situations are encountered for which no rules are present. A combination of problem solving processes and stored knowledge are used to plan the actions. Errors in this domain may occur as a result of a lack of correct knowledge or the presence of incorrect knowledge.

The skill-, rule- and knowledge-based framework is further elaborated by Reason (1990), who used the three domains to explain a variety of errors.

2.2.2.3 Slips and Mistakes

Reason (1990) further builds on Rasmussen's (1982) performance domains in classifying errors, relating it to the widely used distinction between slips and mistakes. Slips are defined as

> "an error in carrying out the intention" (Norman, 1981),

whereas mistakes are defined as

> "an error in the intention itself" (Norman, 1981).

Or, in Reason's (1990) words, slips and lapses can be seen as execution failures, and mistakes can be seen as planning failures. Reason (1990) further distinguishes between skill-based slips and lapses, rule-based mistakes, and knowledge-based mistakes.

However, while the distinction between slips and mistakes is a simple and efficient way to classify errors, in reality it is not always as clear-cut. Gray (2004) argues that

> "the knowledge-based, rule-based, and slip-based approach to errors is neither as neat and clean nor as theory-based as it may first appear. Whether an error is classified as skill-based, rule-based, or knowledge-based may depend more on the level of analysis than on its ontogeny" (p. 3).

Moreover, while this distinction identifies the general cognitive processes that underlie different errors, it makes only limited assumptions about causality. For a more conceptual taxonomy, the next section will discuss Norman's (1981) taxonomy of action slips, which classifies them according to their underlying causes.

2.2.2.4 Categorisation of Action Slips

Norman (1981) is also a proponent of the distinction between slips and mistakes. He developed a taxonomy of different types of slips, based on his Activation-Trigger-Schema (ATS) theory (Norman, 1981). Norman's taxonomy can be argued to be a hybrid approach, as he makes an attempt to classify slips according to their underlying causes, but also looks at their superficial characteristics. The ATS model is briefly outlined here for the purpose of categorising action slips. The model, and its successors, will be discussed in greater detail in section 2.4.2.1, on the execution of routine procedural action.

The basic building block in the ATS model is the schema, which is a sensori-motor knowledge structure. Schemas have an activation value and triggering conditions. Actions are governed by the activation and selection of schemas. However, activation itself is not sufficient to ensure the correct order of actions, and as such the triggering conditions also need to be satisfied for the operation of a schema. The triggering conditions can be met when they are matched by the current situation, but the match does not have to be precise. In fact, Norman (1981) argues that this 'fuzzy' matching makes it possible to account for action slips. The triggering of a schema is a trade-off between the activation value and how well the triggering conditions are met.

Norman (1981) argues that slips can occur in the formation of intention, the activation of schemas, and their triggering.

Slips in the formation of intention occur when situations are incorrectly classified, or when the intention formed is incomplete. An example of the former is a mode error, which occurs when a person makes incorrect assumptions about the state of a system, and responds with an action that is appropriate only for a different mode. An example is starting to type some text while the caps lock key is on. Other errors in the formation of intention are description errors, which occur when the intention formed is incomplete or too vague, leading to incorrect but closely related actions being executed.

Errors in the activation of schemas can occur in two ways. First, an incorrect schema may gain too much activation. A typical example of this is a capture error, which occurs when an infrequent action is substituted by a more familiar action. Reason (1979) provides a description:

> "Like the Siren's call, some motor programs possess the power to lure us into unwitting action, particularly when the central processor is occupied with some parallel mental activity. [...] The more frequently (and recently) a particular sequence of movements is set in train and achieves its desired outcome, the more likely it is to occur uninvited as a 'slip of action'." (Reason, 1979)

Second, a correct schema may lose its activation before it can direct behaviour. Activation can be lost through decay or interference. This type of slip occurs when initiating an action sequence, but along the way forgetting what the goal was. A typical example is walking into the kitchen, but upon opening the fridge door discovering that you no longer remember what

your goal was. Omission errors, in which a single step is forgotten, also occur as a result of a loss of activation.

Errors in the triggering of schemas can also manifest in two ways. First, a schema can be triggered at the wrong time, leading to reversals of actions, or attempting to execute two actions at the same time. A famous example is the 'spoonerism', in which the syllables of two words are incorrectly combined, such as 'wave the sails' instead of 'save the whales'. Second, the correct schema can fail to be triggered. This happens when the triggering condition is poorly specified, or when the activation of the schema is low.

As this section discussed, Norman's (1981) categorisation of slips explores the causal mechanisms underlying errors. As such, this thesis has now considered the three different types of taxonomies: behavioural, contextual and conceptual (Reason, 1990). The next section summarises this in order to define the scope of the current thesis.

2.2.3 Scope

The initial development of definitions and taxonomies has been a useful starting point for research on human error. While most taxonomies have their shortcomings, they are invaluable tools for scoping human error research, and indeed the current work.

The focus of this thesis is on slip errors, or failures in the execution of skilled procedures. The errors being studied are not the result of a lack of, or incorrect, knowledge. As such, they are infrequent, but nevertheless persistent. This also makes these slips difficult to reduce by training or 'trying harder', as will further be discussed in section 2.5 below. This makes them particularly interesting to the study of cognitive science, because they can be seen as failures in the cognitive processes that underlie the execution of skilled procedures.

Having defined the type of error being studied in the current work, it is also important to further define the type of task under investigation. The next section describes the notion of a task, and examines the type of task relevant for the current work. Moreover, it will analyse the basic 'building blocks' of the task, and discuss the level of analysis on which the current thesis operates.

2.3 Goals, Tasks and Skilled Procedures

The current thesis focusses on slip errors, which occur when people have the correct knowledge to do a task. Therefore, the tasks being studied in the current thesis are well-learned, and largely sequential. While skilled tasks can occur in any domain, the current thesis is most concerned with those that use a device of some sort, such as a computer interface, an airplane cockpit or an infusion pump. The experiments presented in this thesis primarily use computer-based tasks that, to an extent, resemble physical device interfaces. This section looks more closely at what these tasks involve, and how they can be defined and represented.

2.3.1 Level of Analysis

Many researchers have argued that sequential tasks can often be represented hierarchically (e.g. Byrne and Bovair, 1997; Cooper and Shallice, 2000; Altmann and Trafton, 2002). This means that they can be analysed at different levels of the hierarchy.

Cooper and Shallice (2000) describe three levels of action. The lowest level is specified by motor response schemas, and is concerned with fine-grained biomechanical control and the physical properties of the target. At the highest level the overall goal is specified, such as 'going to the doctor' or 'having breakfast'. The details of the physical environment, such as the route to the doctor's office and the type of breakfast being consumed, are not relevant at this level. Between these levels is an intermediate level, in which the actions are cognitively represented as discrete units. The environment is important at this level, while the lowest level actions are abstracted over. The intermediate level forms the basis of Cooper and Shallice's (2000) IAN model, which will be discussed in more detail in section 2.4.2.3. Other levels of analysis have been proposed as well, such as Newell's (1990) four 'bands' of abstraction: a biological band, a cognitive band, a rational band and a social band.

Indeed, the current thesis is concerned with Cooper and Shallice's (2000) intermediate level, and Newell's (1990) cognitive band. For the purpose of the experimental work presented, the relevant level of analysis is a discrete action or step at the task interface that effects a change. This change can be in the state of the interface or the state of the task, and may or may not be visible. For instance, a click on a button or entering a number into a text box can be seen as an individual action. Finer-grained acts, such as moving the mouse, or pressing down the mouse button are not considered to be separate actions.

2.3.2 The Building Blocks of Tasks

The idea that tasks can be divided into smaller building blocks is relatively uncontroversial. However, there is less agreement on what the appropriate units are, and how they are organised. Schemas (e.g. Norman, 1981; Cooper and Shallice, 2000), operators (e.g. Payne et al., 1990), goals (e.g. Altmann and Trafton, 2002), steps (e.g. Li et al., 2008), and actions (e.g. Byrne and Bovair, 1997) are examples of the building blocks used. These are closely related concepts and there may be considerable overlap between them. Because the basic task unit is a core concept in the current work, it is important to further discuss it. This section examines the different approaches taken by different lines of work, aiming to identify the most appropriate approach for the current work. It does not, however, look at how tasks are governed; this is covered in the next section on cognitive models of routine procedural action.

2.3.2.1 Goals

Altmann and Trafton (2002) approach tasks in terms of the overall goal and its constituent subgoals. They define a goal as:

> "A mental representation of an intention to accomplish a task, achieve some specific state of the world, or take some mental or physical action."

Cooper and Shallice (2000) also use goals in their computational model, and similarly define them: "a goal is a condition that may or may not be satisfied by the world." In both these definitions, goals have no physical effect in the real world. Rather, they provide structure and governance to physical actions through their hierarchical representation. Altmann and Trafton (2002) further argue that the lowest-level subgoals direct actions.

2.3.2.2 Schemas

As discussed in section 2.2.2.4, Norman (1981) uses schemas as the basic building blocks of tasks. Schemas have been widely used in many areas of psychology, and as such there are a variety of different definitions, according to different needs. For the purpose of his ATS model, Norman (1981) defines them as

> "organised memory units",
>
> "mental structure that represents some aspect of the world",
>
> "sensori-motor knowledge structures", or
>
> "organized body of knowledge, including procedural knowledge that can direct the flow of control of motor activity".

Cooper and Shallice (2000) also use schemas as a fundamental unit of organised behaviour. They argue that schemas are goal-directed, in the sense that each schema achieves a goal. Thus, schemas are the 'methods' by which goals are achieved. For instance, if a goal is that a cup of coffee is sweet, then the associated schema may be to add a spoonful of sugar to the coffee. Both Norman (1981) and Cooper and Shallice (2000) argue that, like goals, schemas are organised hierarchically.

2.3.2.3 Actions and Steps

Other related concepts that have been used to describe routine procedural performance are 'actions' and 'steps'. They are not very well defined, but their meaning and usage are intuitive. While goals and schemas can be seen as internal representations, actions and steps are more physical, and can have an effect in the real world. Li (2006) uses the term 'step' to describe a single operation on a task interface, such as clicking a button. Similarly, Cooper and Shallice (2000) use the term 'action' to describe a closely related concept. Indeed, Altmann and Trafton (2002) use both terms interchangeably. As such, actions and steps are taken to mean single, discrete operations in a task.

2.3.2.4 Operators

A notion closely related to steps and actions is the 'operator', as used by Payne et al. (1990) in their Yoked State Space hypothesis (see section 3.2.1.2 for a discussion). They argue that

"device operators allow the user to transform states in the device space". They use the terms 'step' and 'operator' more or less interchangeably, although they argue that operators consist of one or more basic interactions with the device. For instance, clicking a button with a mouse can be seen as an operator, while this can further be decomposed into the basic interactions of moving the mouse, pressing down the button and releasing it.

As this discussion demonstrates, goals, schemas, actions, steps and operators are all closely related entities. It can be argued that goals and schemas are the internal representations of actions, steps and operators. In the experiments described in this thesis, the main focus is on steps, actions and operators. After all, unlike goals or schemas, these physical operations can be observed and measured, and are thus the most suitable unit in experimental studies. Nevertheless, it is arguably the mental representations of these building blocks that are theoretically the most interesting. As such, all are considered at different points in this thesis.

2.3.3 Skilled Tasks

As section 2.2.3 highlighted, the current work focusses on slip errors, which occur in tasks in which the user is skilled and has the correct knowledge. As such, the tasks that are relevant in this thesis are those that are well-learned, and that are approximately sequential in nature. Tasks that are less relevant to the current work are those that primarily require problem solving, such as the Tower of London task (e.g. Hodgetts and Jones, 2006, also referred to as the Tower of Hanoi task).

Users of these tasks are skilled and can be thought of as approximating expert performance. Fitts and Posner (1967) proposed a three-stage model of the stages people go through when they acquire skills. The first stage is the cognitive stage, which is concerned with developing an understanding of the task, and constructing a method to achieve the goal. In the second, associative stage, the method is practised and performance becomes consistent. In the last stage, the autonomous stage, the skill becomes automatic and performance no longer requires conscious thought or attention.

Based on this work, Anderson (1995) also distinguished between three stages of skill acquisition. In the first stage, people acquire the declarative knowledge needed to execute the task. In the second stage, rules are formed from this declarative knowledge. In the third stage, these rules can be applied more automatically, and fewer cognitive resources needed.

Gray (2000) argued that many instances of routine procedural skill in daily life rely on skills that are "somewhere around the beginning of stage three". Indeed, this is arguably the most relevant stage to the current work, and closest in skill level to expert users of complex systems. It can be argued that performance on highly complex tasks such as surgery or flying a plane may never become fully automatic. For instance, even a skilled surgeon may still need a degree of conscious cognitive control while operating, and as such their performance may not be fully automatic. Therefore, like Gray (2000), the current work is concerned with performance at the beginning of the third stage.

This section considered the properties and organisation of the tasks that are studied in the current thesis. However, a question that remains unanswered is how these tasks are governed. How is each action executed in the right order, and at the right time? To answer these questions, the chapter now turns to the correct execution of skilled tasks. The next section describes how the basic units of tasks are governed to produce skilled procedural behaviour.

2.4 Cognitive Models of Routine Procedural Action

As argued in the beginning of this chapter, in order to study errors, it is also important to understand how tasks are executed correctly. The previous section discussed how tasks are structured using goals, schemas and actions. The current section will explore how this structure is used to produce the execution of skilled, routine procedures. The primary focus of this section is on a number of different cognitive models, which explain how sequential procedures are controlled. The section starts with a brief discussion of the concept of activation, which is commonly used in cognitive models, but inherently ill-defined. It will then continue with an in-depth discussion of a number of different cognitive models, and how they occasionally make errors.

2.4.0.1 Activation

Many cognitive models rely on the 'activation' of goals (or schemas) as the main mechanism that drives sequential behaviour (e.g. Norman, 1981; Norman and Shallice, 1986; Byrne and Bovair, 1997; Cooper and Shallice, 2000; Altmann and Trafton, 2002). In these models, activation is seen as a theoretical construct that indicates how competitive a goal or schema is in relation to other goals. The most active and therefore most competitive goal will direct behaviour. In these models, errors generally occur when the activation of the relevant goal is lower than that of an incorrect, competing goal.

However, it can be argued that activation is more than a theoretical construct. Jonides et al. (2008) argued that activation can be seen in terms of short-term memory. In their review, they argued that short-term memory is the neuronal activation of long-term memory representations. Thus, for an item (or goal) to be 'active' means that it is currently represented in short-term (or working) memory. Moreover, the activation of memory elements is the principal selection mechanism. Thus, it can be argued that activation is not merely a theoretical construct, but the concepts used in memory research and cognitive models are closely related.

Three models that have been used to explain errors and routine procedural performance are the Memory-for-Goals model (Altmann and Trafton, 2002, 2007; Trafton et al., 2011), the contention scheduling models (Norman and Shallice, 1986; Cooper and Shallice, 2000, 2006), and the Simple Recurrent Network model (Botvinick and Plaut, 2002, 2004). These will now be discussed in detail.

2.4.1 The Memory-for-Goals Model

Altmann and Trafton (2002) developed the Memory-for-Goals (M-f-G) model[2], which aims to explain goal-directed performance using ordinary memory structures. It was developed in response to the many theories of cognition and cognitive control that use an explicit goal stack to keep track of the current goal (e.g. Anderson and Lebiere, 1998; Newell, 1990). Altmann and Trafton (2002) argued that such a stack is not necessary to account for goal-directed behaviour; it is merely an artefact of the use of the 'stack' metaphor during years of research in the area. The M-f-G model was one of the first that explicitly did not use a goal stack. Instead, it makes use of existing memory constructs to implement memory for goals. As such, goals are represented in the same manner as other types of memory elements. They are, therefore, subject to the same constraints as ordinary memory elements, such as noisy retrieval and decay. Altmann and Trafton (2002) implemented their model in ACT-R (Anderson and Lebiere, 1998) and demonstrated that it accurately predicts performance on a well-known problem solving task. Their simulation provided support for their claim that a separate goal stack is not necessary to achieve sequential behaviour.

2.4.1.1 Core Assumptions

A major assumption of the M-f-G model is that goals, like any other memory element, have an activation level; the most active goal will direct behaviour. The model is largely based on the ACT-R cognitive theory (Anderson et al., 2004), and takes many of its features from this architecture. In addition, the M-f-G model specifies three core concepts:

- The interference level represents the effect of other (distractor) items in working memory. A goal's activation must be above the interference level if it is to be sampled over other goals. The interference level is equal to the activation level of the most active distractor.

- The strengthening constraint requires that the activation of goals is increased to overcome the interference level. This is achieved through repeated sampling of the goal by the system. Strengthening is important in the planning and encoding of new goals: a new episodic trace is created for each new (sub)goal. A (sub)goal that is retrieved more often or more recently will have a higher activation value than others with less history. In other words, the higher the frequency of sampling, the higher the activation of the goal will be.

- The priming constraint states that suspended goals must be primed before they can be resumed. For instance, after an interruption, the original goal will have decayed to below the interference level. To overcome this retroactive interference and reinstate the goal, it must be primed from contextual (such as visual cues in the interface) or mental cues to which the goal is associatively linked. This provides a boost in the activation of the goal, so that it once again may direct behaviour.

[2]Also termed the AGM model (Li, 2006) or the MAGS model (Tsai and Byrne, 2007).

2.4.1.2 Execution of Routine Procedures

While Altmann and Trafton's (2002) M-f-G model is primarily aimed at explaining goal-directed action such as problem solving, it has also been used to describe routine procedural performance (e.g. Li et al., 2008). The basic mechanism through which this occurs is based on associative links. These are part of the priming constraint, and provide an activation boost for the relevant goal at the right time. For routine procedural tasks, the activation boost comes from the previous step in a procedure. Altmann and Trafton (2002) argue that routine procedural action can be seen as a chain of associative links, with each step providing associative activation for the next goal.

Trafton et al. (2011) further developed the M-f-G model as an account of routine procedural action. They argue that for each step in a task sequence, an episodic control code is created that serves as a place-keeper. Unlike the original M-f-G model (Altmann and Trafton, 2002) in which codes were associated with (sub)goals, in Trafton et al.'s (2011) implementation the codes are linked to individual steps in the task procedure. For as long as they are active, these guide the behaviour of the system.

Trafton et al.'s (2011) M-f-G model operates as follows. Upon completion of a step, the episodic code for the next step is generated. In order to do this, the system must consult its (declarative) knowledge to determine what the next step is. Each of the chunks in declarative memory that represent a step also include an associative link to the next step. Thus, using the control code of the previous step, the chunk for the next step is retrieved. An episodic control code is then generated, which guides the behaviour on that next step. Once formed, the activation on this code starts to decay, like any other memory element. However, the mental context can provide associative activation for the current control code, representing the priming constraint (Trafton et al., 2011).

2.4.1.3 How the M-f-G Model Makes Errors

Besides explaining correct performance, models must also be able to explain how performance breaks down and leads to errors. The M-f-G model provides a simple mechanism for errors: when a goal's activation is not high enough to overcome the interference level, it is unable to direct behaviour and its associated action will not be executed. However, the real question is why activation levels are sometimes not high enough. The occasional but persistent nature of slip errors can be explained by Altmann and Trafton's (2002) argument that the activation values associated with goals are noisy. This means that a goal can have the highest activation level on average, but occasionally its activation may randomly fall below the interference level. If this occurs at a time that the system is sampling for the next goal, an incorrect distractor goal may erroneously be selected. This provides a mechanism for the occasional but persistent occurrence of slip errors.

Trafton et al. (2011) also make more specific predictions about how errors occur in routine procedures. In their version of the M-f-G model, the mechanism for errors relies on the oc-

currence of an interruption in the task sequence. It assumes that interruptions generally occur neatly in between steps. In this situation, the previous step has been executed, while the next control code has not yet been generated. Thus, upon task resumption, the most active control code will be the one of the step that was last executed, which will then guide the generation of a new control code. This can explain how tasks are normally resumed correctly. However, occasionally random noise on the activation levels for the control codes can lead to an earlier step being the most active instead. If this step is then resumed, a perseveration error occurs. Alternatively, it can happen that the control code for the next step is created *before* the interruption occurs. Upon resumption, the system retrieves this step, but crucially assumes that it has already been executed. Indeed, the control code does not represent whether or not the step has been completed.

Trafton et al.'s (2011) explanation of sequence errors relies upon the disruptive effect of interruptions. However, sequence errors have also been observed in tasks without interruptions or on experimental trials in which no interruptions were present (e.g. Li et al., 2008). It is not clear what errors their updated model makes when no interruptions are present at all.

In short, the M-f-G model provides a plausible account of goal-directed behaviour. In addition, it is also a useful model of routine procedural action, as demonstrated by Trafton et al.'s (2011) extension of the model. The current work also considers other accounts of routine sequential behaviour. The next section discusses contention scheduling and interactive activation networks, and examines whether they provide a suitable alternative for the limitations of the M-f-G model.

2.4.2 Contention Scheduling and Schema Network Models

2.4.2.1 The Activation-Trigger-Schema Model

As discussed in section 2.2.2.4, in his attempt to classify slip errors, Norman (1981) developed a model of routine procedural action called the Activation-Trigger-Schema (ATS) model. Recall that the basic unit in the ATS model is the schema, which has an activation value that governs its selection state. Triggering conditions allow actions to be executed at the correct time.

Norman (1981) argues that schemas are organised in a hierarchy, with the highest-level schema as the parent schema. Its subschemas are called child schemas, which can in turn be parent schemas to further child schemas. Norman (1981) argues that intention can be seen as the activation of the top-level schema, which in turn will activate the child schemas in order to execute the associated action sequence. Thus, for a skilled routine procedure to be executed, only the highest level schema needs to specified. All component child schemas will automatically be activated, allowing for the associated actions to be executed without much conscious control.

Multiple schemas can be active at the same time, to allow complex procedures to be executed, which may consist of a large number of component schemas. Moreover, multiple intentions and associated schemas can be active at the same time, such as when doing multiple

tasks simultaneously. An example would be walking to work, but on the way stopping to buy a sandwich.

Activation can come from multiple sources, such as the external world, internal processing (e.g. other associated memory structures) or familiar habits. Norman (1981) argued that slips can occur in the formation of intention, the activation of schemas, and their triggering. However, while the ATS model is capable of explaining various action slips, it does not provide a full account of normal routine procedural performance. Later studies have built on the ATS model, and developed a more complete account. These will be discussed in the next sections.

2.4.2.2 Contention Scheduling

Norman and Shallice (1986) proposed a model of action control, which can account for both deliberate, conscious action such as problem-solving, as well as unconscious, routine procedural action. The model proposes that there are two processes through which action sequences can be controlled. The first, called contention scheduling, allows the execution of simple or well-learned actions. The second, called the Supervisory Attentional System (or SAS), is responsible for the conscious control of actions, such as in problem solving or other more complex tasks.

At the heart of Norman and Shallice's (1986) model are schemas, which are organised hierarchically for a particular task. As in the ATS model, each schema has an activation value, which determines whether or not it will be selected. Once the schema's activation level is higher than a threshold, it will become selected, and, if it is a low-level schema, its associated action will be executed.

A schema's activation level is influenced by a number of factors: the satisfaction of triggering conditions, the selection of other schemas, "vertical thread" influences, and contention scheduling. Triggering conditions specify the environmental conditions under which the schema can be executed. They control the precise timing of schema selection. The amount of activation received through this mechanism is determined by how well the conditions are met. The selection of a schema influences the activation of other schemas as well. First, source schemas provide activation to their component schemas. Second, competing schemas inhibit each other, while complementary schemas excite each other. Contention scheduling works through the activation and inhibition of supporting and conflicting schemas, and regulates the use of resources.

Norman and Shallice (1986) argue that their model is capable of making various slip errors. A capture error occurs when an incorrect schema is selected instead of the correct schema. Norman and Shallice (1986) argue that this occurs when the supervisory system is otherwise engaged. If an incorrect schema then gains too much activation through the contention scheduling system, the supervisory system may fail to notice it, resulting in a capture error. Thus, in essence, these errors result as a failure of the supervisory system to monitor the activation of schemas effectively. However, as Cooper and Shallice (2000) point out, this is 'unprincipled', because action lapses occur in routine behaviour, and as such would not be under control of the supervisory system. They argue that "action lapses could be viewed instead as the loss of

top-down volitional control within the routine action control system itself".

Norman and Shallice's (1986) contention scheduling system has been influential in the study of routine procedural action. It has featured in many textbooks on cognitive science (e.g. Gazzaniga, 2010), and sparked the development of further refinements and computational implementations. Once such implementation, the IAN model, will be discussed in the next section.

2.4.2.3 Interactive Activation Network Model

Norman and Shallice's (1986) model has been influential in the study of routine procedural action. However, since it originally existed only as a verbal account, it was not possible to accurately test its predictions. To remedy this, Cooper and Shallice (2000) developed a computational model of the control of action, based on Norman and Shallice's (1986) theory. This model is called the IAN (Interactive Activation Network) model.

Core Assumptions

An important addition in Cooper and Shallice's (2000) model is the use of goals. Norman and Shallice's (1986) original model did not make use of goals, and represented action sequences only in terms of schema networks. However, Cooper and Shallice (2000) argue that goals play an important role in the selection of schemas. As such, the IAN model describes the relations between both goals and schemas. Goals are defined as "a state of affairs that an agent aims to achieve" (Cooper and Shallice, 2006), and schemas can be seen as 'methods' that achieve these goals. Schemas and goals are organised in a hierarchical network, with alternating layers of goals and schemas. A (sub)goal may have several schemas that achieve it, but only one needs to be executed to achieve it. In contrast, a (sub)schema may have several subgoals, but each must be achieved to carry out the schema.

In the model, schemas are represented as a node in the schema network. Each node has an activation value that is regulated by a number of different processes. Unlike in the M-f-G model (Altmann and Trafton, 2002), goals do not have activation levels. Schemas become selected when their activation value exceeds a threshold. Once a schema is selected, it provides activation for its component schemas (in case of a higher level schema), or its associated action can be executed (in case of a low level schema). Multiple schemas can be selected at the same time. For instance, a source schema will remain selected for the duration of the action sequence, and one of its component schemas may be selected as well.

In the IAN model, there are a number of ways in which a schema's activation can be influenced:

- Environmental triggering situations: the presence of a particular object in the environment may provide activation for schemas that utilise this object.

- Top-down influence from higher-level schemas: selected parent schemas can provide activation for their component schemas. The amount of activation that is received from a

parent schema is inversely proportional to the number of 'siblings' a schema has. Another way in which this top-down influence works is through the Supervisory System. Cooper and Shallice (2000) argue that "the direct excitation or inhibition of schema activation values is the only way in which the Supervisory System may affect behaviour". Thus, it cannot directly change the selection state of schemas.

- Lateral influence: schemas that are in competition for resources have an inhibitory influence on each other, and conversely complementary schemas excite each other.

- Self-influence: this works to excite schemas, and can counter the lateral inhibitory influence.

In addition to these sources of activation, there are a number of further factors that determine a schema's activation. First, activation is persistent in the absence of any influence. Second, schemas have a 'rest activation' to which activation will eventually return if there is no net influence. In addition, normally distributed pseudo-random noise will modulate activation levels.

In the IAN model, the activation sources are further governed by a number of parameters. The 'Self:Lateral' ratio controls the relative influence of self-activation versus lateral inhibition. These are the competitive factors. The 'Internal:External' ratio governs the balance between external activation sources and top-down influence. These are the non-competitive factors. The third parameter concerns the balance between competitive and non-competitive processes ('Competitive:Non-competitive').

Besides the schema network, the IAN model also uses two other networks: the object network and the resource network. Figure 2.1 shows how these components are organised. Nodes in both these networks also have activation values, and therefore they are also subject to interactive activation. The object network represents objects that are available for use in the action sequence, such as a spoon or a pair of scissors, and the resource network represents the resources available to the system, such as the left and right hands. Object nodes and resource nodes interact with each other and with schema nodes, through processes such as lateral inhibition or excitation. A difference with schemas, however, is that objects can have different functions, such as 'implement', 'source' or 'target', which are represented by separate activation values. For instance, when adding sugar to coffee, the sugar pot has a high 'source' activation value, but low 'target' and 'implement' activation values.

When a basic-level schema becomes selected, its argument roles will become filled with the most active object representation for the relevant function. This means that each action type only needs to be represented by one schema. For instance, a schema for the action 'stir' takes object arguments for its implement and its target. This allows for flexibility in action sequences. If the goal is to stir a cup of coffee, the 'stir' schema may use a spoon as its implement, or, if this is not available, it may use a disposable stirrer instead. It is not necessary to have separate schemas for each action/object combination.

Figure 2.1: *Diagram showing all the components of the IAN model, and the relations between them. From Cooper and Shallice (2000).*

Execution of Routine Procedures

An important issue in any model of routine action is how schemas are executed in the correct sequential order. Cooper and Shallice (2000) identify three constraints on the ordering of schemas. First, actions that are not physically possible will not receive environmental influence. In other words, their triggering conditions, which serve as gates for environmental influence, are not satisfied. Second, some actions may be physically possible, but if executed at the wrong time would lead to the non-achievement of the goal. These are mediated by preconditions, which act as gates for top-down influence. Third, there can be variation between people if the order of actions does not matter.

Once selected, schemas can become unselected in two ways. First, their activation can fall below the activation level of a competitor or below the threshold. Second, once a (sub)goal is achieved, its associated schema becomes unselected. This is mediated by postconditions, which are important in monitoring the achievement of goals. They specify the conditions that must be true after the schema is executed. When a component schema is deselected (i.e. its action has been executed), it is 'ticked off' and its goal is marked as achieved. If all subgoals of a source schema are marked as achieved, the schema is inhibited. Similarly, if the source schema becomes unselected, the component schemas will also be deselected.

How the IAN Model Makes Errors

An important feature of the IAN model is its capability of making a variety of errors. Three of these errors are highlighted here.

Capture errors occur when a well-practised action sequence seizes control over a similar, less well-practised sequence. In the IAN model, schemas are triggered when the state of the environment is compatible with their execution. Normally, this triggering is not in itself sufficient to select the associated schema. However, Cooper and Shallice (2000) argue that capture errors can occur when the Internal:External parameter is biased towards external sources of activation. If the environmental influence increases, the satisfaction of triggering conditions can be sufficient to select a low-level schema. If the schema in question is incorrect, a capture error occurs. Similarly, the presence of an object in the environment provides activation for the schemas that use it, and as such can lead to the selection of the incorrect schema. Alternatively, capture errors can occur at higher levels in the schema hierarchy when "competition is inappropriately resolved" (Cooper and Shallice, 2000). This can happen when the self-activation parameter is over-represented.

Omission and anticipation errors occur when an action or subgoal is forgotten. In the IAN model, this can occur in two ways. First, a schema's activation level may not be high enough to reach the threshold, and will thus not be selected. This can happen when there is too little self-activation, or when the competitive processes are over-represented. Second, anticipation errors can occur when the schema might be selected, but its action cannot be executed because the appropriate arguments or resources are not available. An example of this is trying to pour from a bottle before its cap has been removed.

Perseveration errors occur when a previously completed action is accidentally repeated. This occurs through the failure of competitive processes, specifically when self activation is too high or when lateral inhibition is too low. The result of this can be that a selected schema will not become unselected. Another way in which perseveration errors can occur is through the presence of a particular object in the environment. If the activation value of a recently utilised object remains high, then it will continue to provide activation for the corresponding schema.

In short, the IAN model provides a detailed and testable account of how routine actions are selected. Schemas are in direct competition with one another through lateral inhibition and activation. This provides an intrinsic mechanism for limiting the capacity of cognitive processing. The model is capable of making a variety of errors, which occur at different levels. While no learning mechanism is specified, the model is flexible enough to adapt to small changes in the environment. As such, the IAN model is a highly plausible and useful account of routine procedural action.

Nevertheless, the model (and others like it) has been criticised by Botvinick and Plaut (2004, 2009) for its use of hierarchical representations of goals and schemas. Botvinick and Plaut (2004) argue that it is not likely that the brain's structure mirrors its environment. Moreover, they criticise the lack of a learning mechanism in the IAN model. The next section will discuss the simple recurrent network model developed by Botvinick and Plaut (2002, 2004), which aims to address the shortcomings in traditional, symbolic models.

2.4.3 Simple Recurrent Network Model

The majority of models of routine procedural action use explicit representations of goals as an important factor in generating behaviour. However, the various models do not all agree on what this representation should look like. For instance, Altmann and Trafton (2002) argue that goals have an activation level, and the most active goal directs behaviour. Cooper and Shallice (2000) argue that schemas are the 'carriers' of activation, while goals mediate between schemas but have no activation value. Botvinick and Plaut (2004) go a step further and argue that explicit goal and schema representations are not necessary to achieve routine sequential behaviour. Their Simple Recurrent Network (SRN) model Botvinick and Plaut (2002, 2004, 2009) uses distributed, graded patterns of active and inactive neuron-like units, instead of explicit representations. Despite the lack of explicit goal representations, the model is capable of producing routine action sequences. Moreover, Botvinick and Plaut (2004) argue that it is capable of showing goal-directed behaviour, such as repeatedly taking a sip of tea until the cup is empty, and it can learn that several different actions can accomplish the same goal.

2.4.3.1 Basic Operation of the Model

The SRN model consists of three layers of neuron-like units. The first is the input layer, which receives input from the environment, as well as instructions. The third layer contains the action units, and is used to make changes to the environment. The action units specify which action is to be executed. In between these layers is the hidden layer, which consists of context units (Botvinick and Plaut's tea and coffee making model used 50 of these hidden units). Each hidden unit is connected to every other hidden unit in this layer, as well as to all units in the input and output layers.

The model operates in cycles; on each cycle, activation spreads between units through weighted connections. This means that the hidden units receive activation from the input layer as well as recirculated activation from each other. An action is selected for each cycle. The patterns are determined by the different connection weights between units.

2.4.3.2 Benefits of the Model

Botvinick and Plaut (2002) argue that not only can their model account for sequential actions, it also has several advantages over more traditional accounts that use explicit representations of goals or schemas. First, it allows for 'underspecification', which means that task representations can be vague. It is this vagueness that allows the model to make errors like those observed in human behaviour, such as capture errors (the mechanism for these errors will be described in more detail in the next section). Botvinick and Plaut (2002) argued that symbolic models such as the IAN model cannot represent such vagueness, and as such have trouble explaining certain error patterns.

Second, the SRN model allows for information sharing, which is based on the idea that similar (sub)tasks rely on common task representations. For instance, when making tea or coffee, one of the subtasks may be to add sugar from a bowl. In the SRN model, these two

would largely share this representation. They would be distinguished only by the representation of their context. Botvinick and Plaut (2002) argue that symbolic models are unable to fully explain information sharing. They may share subgoals, but there is no flexibility to adapt these subgoals to the tasks needs. For instance, a person may require a larger scoop of sugar in their coffee than in their tea. They argue that in traditional models, these two similar subtasks would have to be represented separately to allow for the difference in scoop size. However, Cooper and Shallice (2006) argued that in their IAN model, this could easily be represented by 'manner features' that can accompany schemas.

2.4.3.3 How the SRN Model Makes Errors

The SRN model is capable of making errors that are also found in human behaviour, such as capture errors. Because similar tasks share task representations, occasionally the execution of one task will lapse into the other, causing a capture error. Botvinick and Plaut (2002) represent (sub)tasks as points in n-dimensional space, where each of the n nodes in the hidden layer of their model correspond to one dimension. Because task representations may overlap, tasks that are similar will be closer to one another in space then those that are different. When tasks are underspecified, or when random noise is applied to the system, their point in space may shift slightly. When two tasks are represented close to one another in space, such as tea and coffee making, it is possible that a lapse from one task into the other occurs when the active representation is degraded. In contrast, this is much less likely to happen between tasks that are further away from one another in representational space. This capture process is able to explain a variety of behavioural manifestations of error. For instance, if the context representation comes to resemble the context for the *next* step, an omission error may occur.

Botvinick and Plaut (2002) argue that more traditional computational models, like Cooper and Shallice's (2000), are not able to fully explain why these capture errors occur. Since the IAN model's task representations are discrete, a lapse from tea to coffee making should be no more common than a lapse from tea to sandwich making.

2.4.3.4 Shortcomings

The SRN model has a number of weaknesses, many of which are highlighted by Cooper and Shallice (2006). First, all errors that the model makes can be related back to the mechanism for capture errors. These errors occur when two task representations are similar, and a lapse from one into the other occurs. The model's mechanism for errors is greatly dependent on training or previous experience. As Cooper and Shallice (2006) showed in their reimplementation of the SRN model, it makes omission errors of the kind A → (B) → C (with (B) being the omitted step) only if the model was also trained on a sequence A → C. Similarly, it makes perseveration errors of the kind A → B → C → (B) (with (B) being the repeated step) only if the model was also trained on a sequence C → B. However, it seems clear that errors in routine procedural tasks do not only occur when the erroneous sequence has been previously learned as well. For instance, Trafton et al. (2011) used a version of the Doughnut

task (discussed in detail in section 4.2.1) to study perseveration and omission errors. The task consists of 5 subtasks which are very similar in structure to one another. Trafton et al. (2011) defined an omission error as skipping a subtask, whereas a perseveration error was defined as the repetition of an already completed subtask. Participants were trained only on the correct task sequence. However, as expected, they found that participants made a substantial number of omission and perseveration errors, even though the erroneous sequences used were not part of the training. Indeed, it seems common sense that not all errors made on routine procedural tasks were once part of a training sequence. As such, while the SRN model provides a plausible account for capture errors, which can even account for some omission and perseveration errors, it is problematic in explaining errors beyond this mechanism.

A second shortcoming is the lack of predictions about the timing of action sequences. In the SRN model, an action is selected on the majority of cycles. Since each cycle takes the same amount of time, this is leads to each step the SRN model executes taking the same amount of time. This is problematic for a variety of reasons. First, it means that the model currently does not make predictions on the time courses of action sequences. This has often been an invaluable tool in determining how well a model fits human data (e.g. Anderson and Lebiere, 1998; Altmann and Trafton, 2002, 2007; Hodgetts and Jones, 2006). Second, the lack of a representation of time in the SRN model means that time-based decay of activation patterns (as shown by Altmann and Trafton (2007), amongst others) cannot easily be represented or explained.

Third, the SRN model relies heavily on training. This can be seen as both a strength and a weakness of the model. Botvinick and Plaut (2004) argue that most models of routine sequential action do not explain how they are learned. Instead, the SRN model is highly dependent on the training it receives. In fact, Cooper and Shallice (2006) argue that this is detrimental to the model, as it cannot do anything that it has not learned before. The model does not generally execute erroneous sequences that it has not learned before. Moreover, it is not able to execute novel task sequences or display flexible behaviour, such as using a disposable stirrer in case a spoon is not available.

In short, the SRN model is capable of executing well-learned action sequences without the explicit representation of goals and schemas. Moreover, it is capable of making errors, although the mechanism for these action slips is limited to different versions of capture errors. As such, it provides a valuable and interesting contribution to current research on sequential action. However, the model has a considerable number of shortcomings that are not easily resolved within the architecture.

The current section discussed a number of different cognitive models of routine procedural action, and showed the models' strengths and weaknesses in explaining the correct and erroneous execution of action sequences. However, apart from this, the cognitive models must be

able to account for other experimental findings associated with routine procedural action, such as working memory load and interruptions. The following section discusses a number of experimental studies of human error, and examines how the models discussed above account for their results.

2.5 Experimental Studies of Slip Errors

While early research on human error has mainly focussed on naturalistic settings (e.g. Reason, 1979; Norman, 1981; Rasmussen, 1982), more recent research has taken an experimental approach. Certain types of routine procedural errors can be reliably provoked in laboratory settings and are relatively easy to study empirically (e.g. Byrne and Bovair, 1997; Li, 2006; Ratwani et al., 2008), thus allowing for the controlled study of these errors. The current section aims to gain a deeper understanding of the properties of slip errors. Moreover, the experimental findings will be assessed against the models discussed in the previous section. This will further test the validity of these models and their use in explaining action slips.

2.5.1 Post-Completion Error

Much of the experimental work on slip errors has focussed on the post-completion error (PCE). This error occurs when the last step in a procedure is forgotten *after* the main goal has been completed. Examples of the PCE include forgetting to retrieve the original after making copies on a copy machine, forgetting to replace the gas cap after getting petrol at a gas station, or forgetting a semi-colon after a line of code (see appendix C for additional examples). Byrne and Bovair (1997) have argued that the PCE occurs because after the main goal has been completed, it no longer provides activation for the PC subgoal. Thus, the PC subgoal has a relatively low activation level, and as a result it is sometimes forgotten. PCEs are particularly persistent errors (Back et al., 2006; Byrne and Davis, 2006; Chung and Byrne, 2008), and therefore cognitive models of routine actions should provide an explanation for their occurrence.

The M-f-G model is the only of the three models presented here that explicitly provides an explanation for the post-completion error. Altmann and Trafton (2002) argue that PCEs occur as a result of goal-forgetting. Because the PC step must be executed at the end of the task, its activation has decayed the most and therefore the error should occur by default. However, PCEs (and indeed forgetting the final step of any task, post-completion or not) are usually avoided because of an associative link between the pre-PC and the PC subgoal that provides the required activation. Li (2006) argues that this early M-f-G account of post-completion errors is problematic, because the argument is not specific to the PC subgoal. By the same reasoning, the error rate should steadily increase towards the end of any task, since the subgoals will have decayed substantially. Similarly, in a very long task, performance should be much worse than in shorter but similar tasks. Li's (2006) argument seems to be based on the idea that the PC subgoal has been initialised at the beginning of the task, and subsequently suspended. However, there is substantial evidence that such planning does not occur in routine procedural

tasks (e.g. Payne, 1994). Indeed, Trafton et al.'s (2011) implementation of the M-f-G model does not initialise subgoals at start-up.

Trafton et al. (2011) further argue that omission errors occur when an interruption occurs *after* the next control code has been generated but *before* its step can be executed. While this error mechanism appears plausible, it is not satisfactory in explaining PCEs. First, it predicts that, in the absence of interruptions, no PCEs are made. Second, it cannot explain why the PCEs occur so much more frequently than omission errors on other steps. In other words, this explanation is not specific to the PC step. Moreover, the proposed mechanism for errors predicts that the majority of errors after an interruption must be perseveration errors. However, this is not supported by experimental evidence (e.g. Li et al., 2008). Indeed, as Trafton et al. (2011) argue, a more specific account of the PCE in terms of the M-f-G model is needed.

The SRN model does not provide an explicit explanation for post-completion errors. Indeed, it is difficult to see how it would account for this specific error. First, it is unlikely that for each post-completion situation, there will also have been a training sequence *without* the post-completion step. Second, Byrne and Bovair (1997) have argued that goal forgetting plays an important part in the PCE. Because the SRN model does not use explicit and hierarchical goals, the explanation of the PCE as an error that occurs after the main goal has been completed is problematic. On the other hand, the false completion signal can indicate to the system that another context is now relevant, such as the next task. This could lead the system to execute actions relevant to this new context, effectively resulting in a post-completion-like error.

The IAN model may be better suited to account for the PCE. The PCE is an omission error, which can occur in the IAN model due to low activation levels. This can happen when lateral inhibition is too strong, or self-activation is too weak. While there is no obvious reason why self-activation would be lower specifically on the PC step, the lateral inhibition may be increased due to competing signals. Indeed, Li (2006) showed that a false completion and competing signal increased the frequency of the PCE. However, while this may contribute, it cannot fully explain the occurrence of the post-completion error.

Alternatively, as noted before, the notion of goal completion is important in explaining the PCE, and indeed the IAN model accounts for this (see section 2.4.2.3). Goal achievement is mediated by postconditions, which, when satisfied, mark a (sub)goal as completed. Once all subgoals of a source schema are satisfied, the source schema is inhibited. In the case of the PCE, the final subgoal of the top-level schema is omitted. A PCE could then be seen as the inhibition of the source schema before all of its subgoals are satisfied. The IAN model does not currently seem to provide a mechanism for this, as this would require the post-condition mechanism in the IAN model to be 'vague', for instance through noise or mechanisms similar to ACT-R's partial matching. Indeed, Tamborello (2009) built a model of Byrne and Bovair's (1997) Phaser task in an ACT-R implementation of the IAN model, which made PCEs on the basis of such similarity.

In short, the M-f-G and IAN models can partially explain the occurrence of the PCE, while the SRN model makes less plausible predictions. It can be argued that Byrne and Bovair's (1997) original account of the PCE is still the most satisfactory one. Nevertheless, the IAN model seems promising in explaining the occurrence of the post-completion error, because the structure of the model provides a locus for the error that seems highly plausible.

2.5.2 Interruptions

Real-world tasks are frequently interrupted, requiring people to recover and resume the original task. How does cognition support this task resumption? Not only is this knowledge relevant because errors often occur after interruptions, but also because the resumption process may reveal important things about cognition itself.

Hodgetts and Jones (2006) studied interruptions using the Tower of London problem solving task. While doing this task, participants were occasionally interrupted at an unexpected point to do a secondary task. It was found that longer interruptions led to an increase in the time it takes to resume the primary task. Also, more complex interruptions were associated with longer resumption times.

Li et al. (2006) conducted similar experiments using the Doughnut task (this task is discussed in detail in section 4.2.1). They interrupted participants just before some of the steps, and studied the error rates upon resumption. They found that interruptions just before the post-completion step led to higher error rates on this step, in contrast with interruptions that occurred earlier on. Contrary to Hodgetts and Jones (2006), however, they found no effect of interruption length, although the lengths used (15 and 45 seconds) were of a different order of magnitude than those of Hodgetts and Jones (2006) (6 and 18 seconds). Indeed, given the logarithmic nature of the decay curve (Altmann and Trafton, 2002), it is likely that the decrease in activation between 6 and 18 seconds is much larger than the decrease between 15 and 45 seconds.

These experimental studies provide support for activation-based accounts: after longer interruptions, previous goals will have decayed more and it will be more difficult to remember them. Also, more complex interruptions provide more interference, again making it more difficult for the old goal to regain sufficient activation. As such, both the M-f-G model and the IAN model are promising in providing a plausible account.

One of the main goals of the M-f-G model is to account for the effect of interruptions. When an interruption is encountered, the primary task is suspended and the associated (sub)goal's activation starts to decay. When the interruption ends, it is likely that the activation level has fallen below the interference level, and as such must be primed to become the most active again. This priming can be done through external (visual) cues in the task interface, such as flashing lights, or through internal reminders or associative links. However, interruptions are disruptive to these associative links, and therefore occasionally the suspended goal fails to be retrieved.

Trafton et al.'s (2011) extension of the M-f-G model provides an account of recovery from interruptions in routine procedural tasks. Upon resumption, the system will sample the most active control code, which typically represents the previous step in the primary task sequence. As such, the control code can be seen as a place keeper in the task sequence.

The IAN model does not make explicit predictions about interruptions. However, it can be argued that its mechanism for dealing with these is similar to that of the M-f-G model. From the onset of the interruption, the currently active schema will slowly lose its activation, and, given a long enough interruption, will become deselected once the activation falls below the threshold. Upon resumption, the previously suspended schema must regain its activation, though unlike Altmann and Trafton (2002), Cooper and Shallice (2000) do not specify a mechanism through which this must occur. It should be noted that interruptions were not evaluated in the computational model, and as such these assumptions remain untested.

In the SRN model (Botvinick and Plaut, 2004), interruptions are instantiated by applying noise to the system. This can be thought of as representing the change in the system's patterns to adapt to the interrupting task. The noise degrades the representations of the original task, which can lead to difficulties resuming it and increase the likelihood of an error occurring. While Botvinick and Plaut (2002, 2004) do not directly address the effect of interruptions on the SRN model, it can be inferred that task resumption after an interruption may be problematic. Recall that the SRN model typically only executes sequences that it has been taught (e.g. Cooper and Shallice, 2006). As such, it is expected to be able to deal with interruptions only if it has been trained on them, and if the interruption occurs at the same point in the action sequence.

A related study by Botvinick and Bylsma (2005) investigates the effect of interruptions on routine tasks and explains the findings using the SRN model. Previous work had found that action slips are more likely to occur at decision points in the task, rather than during subtask sequences (Norman, 1981; Reason, 1990, see also section 2.5.4). This was explained in terms of a failure in supervisory, attentional processes, such as failing to do an attentional check at the right time. However, Botvinick and Bylsma (2005) present a different hypothesis for why action slips might be more likely at subtask boundaries, based on the SRN model. Specifically, they test the slightly counterintuitive prediction by Botvinick and Plaut (2004) that interruptions or distractions that occur in the *middle* of a subtask are more likely to lead to an error at the next subtask boundary than interruptions that occur *at* this subtask boundary.

Recall that the SRN model represents the task context, such as whether it has already added sugar to a cup of coffee, using distributed, graded patterns. If this context representation becomes degraded, for instance through distractions or interruptions, the information will be lost: the system will 'forget' that it has already added sugar, and may accidentally add it twice.

In relation to task structure, Botvinick and Bylsma (2005) argue that the execution of subtasks is more automatic, and requires little conscious processing. At decision points, however, supervisory involvement is necessary to determine the next action sequence. Building on

Botvinick and Plaut's (2004) work, Botvinick and Bylsma (2005) argue that during the execution of subtasks, the representation of the context is less distinct, because it is not directly relevant to the individual steps in the subtask. At decision points, on the other hand, the context is more strongly represented, because it is directly relevant to determining the next actions. Therefore, the system is more likely to 'forget' the context during the subtask than at the decision point. As described above, interruptions and distractions are instantiated by applying noise to the system. The prediction is that if noise is applied during the subtask, when the representation is most degraded, it is more likely to 'forget' the context. On the other hand, if noise is applied during decision points, when the context representation is stronger, the chance that such a lapse occurs is smaller.

This prediction is indeed supported by experimental data. Botvinick and Bylsma (2005) asked participants to make 50 cups of coffee, and occasionally interrupted them during and after a subtask. They found that participants made significantly more errors at the subtask boundary when the interruption occurred *during* the subtask, rather than after it had finished. While the SRN model can account for this finding, it is not clear how the other two models of routine action would explain these results. Moreover, Botvinick and Bylsma's (2005) explanation is at odds with earlier assumptions that errors at decision points are due to failures in attentional checks.

In short, all three models can account for the disruptiveness of interruptions. In the IAN and M-f-G models, the primary goal or schema loses activation during the interrupting task, while in the SRN model the representation of the primary task context becomes degraded. Both of these lead to difficulties in resuming the task. However, only the M-f-G and IAN models seem able to account for how tasks are resumed. The goal or schema can be reactivated for instance through internal or external primes. As such, both of these models are thought to be successful in accounting for the effects of interruptions, and are in line with the experimental findings.

2.5.3 Working Memory Load

Byrne and Bovair (1997) argued that slip errors occur due to a failure in working memory. Indeed, they found that increasing working memory load increases the occurrence of the post-completion error. They argued that a higher load leads to increased competition between items in working memory. As a result, they lose their activation faster, and goals may fail to reach the required activation level to be executed. This section looks at how the three models analysed here account for working memory load and its effect on error rates.

The M-f-G model is based on the ACT-R cognitive architecture, and as such does not have an explicit working memory. However, it can be argued that the buffers represent working memory, which can each contain a single chunk. However, since the size of chunks is not limited by the architecture, it is not immediately clear how working memory load or capacity can be represented. Chung and Byrne (2008), for instance, represented a high load by having 'dummy chunks' that occupy slots of the chunk currently in the goal buffer. Since spreading activation

from the chunk in a buffer is spread over all its slots, these dummy chunks "stole" activation away from the 'real' chunks (Tamborello, 2009).

Altmann and Trafton (2002) argue that goal decay in the M-f-G model is indexed by time. This means that, under a high working memory load, there are more memory elements competing for the available retrieval cycles. This, in turn, will cause more interference. Interestingly, this can be seen as serial competition rather than parallel competition, though it can be argued that such an approach is slightly awkward from a theoretical point of view.

The IAN model has an intrinsic mechanism that deals with working memory load. In the model, both decay over time as well as interference are represented. Cooper and Shallice (2000) specify a decay parameter that defines how schemas lose their activation over time. Moreover, goals are in direct competition with each other, as instantiated by the lateral influence they have on each other. As such, if only two schemas compete for activation the effects of lateral inhibition are relatively small, and it is easier for a schema to gain activation. However, if there are twenty schemas competing for activation, the effects of lateral inhibition are much larger, making it more difficult for any schema to gain enough activation to be selected. This is consistent with Byrne and Bovair's (1997) model of post-completion errors, as well as with their experimental results.

It is not immediately clear how the SRN model accounts for working memory load. One possibility is that working memory load could be seen as the increase of noise in the system, resulting from competing representations. If multiple items are active at the same time, and thus share the representational space, then each item's representation will be degraded by the representation of the other items. As such, the larger the number of actively represented items, the more the patterns degrade and the greater the chance that an error is made. This seems to be partially consistent with Byrne and Bovair's Byrne and Bovair (1997) predictions.

In short, all models are capable of explaining the finding that working memory load increases slip errors. The way the IAN model deals with working memory load arises from an inherent property of the architecture, rather than an explicit mechanism. While the SRN model has no explicit mechanism for competition, it can be argued that the simultaneous representation of multiple items will lead to the degradation of all items. As such, the SRN model also has an intrinsic capability of explaining the effects of working memory load. The M-f-G model's account of working memory load relies on competition between memory elements, but in terms of a limited number of strengthening cycles rather than direct interference.

2.5.4 Task Structure

Many tasks in the real world are organised hierarchically, with the main task goal being decomposed into several subtasks, which can, in turn, be further subdivided into smaller subtasks. For instance, the widely used example of making coffee consists of subtasks such as adding coffee grounds, adding sugar and adding milk. Each of these consists of several individual actions, such as picking up a spoon, scooping the ingredient and stirring. Such hierarchical organisations of

task steps are highly useful, as they make larger tasks more manageable and organise individual steps into meaningful chunks.

A hierarchical nature of the task structure can also have an impact on the type of errors that occur on a task. For instance, this allows entire subtasks to be omitted, as opposed to individual actions. Another example is the substitution of a subtask with a different one, or otherwise misordering them. The different types of errors that can arise as a result of task structure or other factors are nicely illustrated by a study by Li et al. (2008). They studied routine errors using the Doughnut task, which has a highly regular subtask structure. Li et al. (2008) observed that, at the beginning of each subtask on the Doughnut task, two types of persistent errors frequently occurred. One involved the omission of a single action (activating the widget), while the other involved the selection of the wrong widget altogether. They argue that the former is more related to individual steps in the task sequence, while the latter is a result of a misordering at a higher level in the task hierarchy. Li et al. (2008) argued that, while occurring at the same point in the task, these two errors may have different underlying causes, although they do not discuss in detail what these may be.

A task's hierarchical structure can also have an impact on the error profile of the task. For instance, Tsai and Byrne (2007) compared performance on three routine tasks with identical subtask structure but different interfaces and 'cover stories'. They predicted that errors would be more frequent on the first step of each of the subtasks. Indeed, their data shows a clear trend towards higher error rates at sub-task boundaries, though no statistical comparisons to support this were presented.

Similarly, Ruh et al. (2010) found that in two hierarchically structured tasks, participants were slower to execute actions at subtask boundaries than actions in the middle of a subtask. Ruh et al. (2010) explain this finding by arguing that at subtask boundaries, more deliberate cognitive involvement is necessary to determine what the next subtask is. This supervisory involvement is slower than the more automatic processing that accounts for mid-subtask performance. Moreover, this can also impact the errors that occur at subtask boundaries: if these attentional checks are omitted or are otherwise faulty, the wrong subtask may be selected. Within a subtask, on the other hand, processing is thought to be more automatic, so any attentional failures may have a smaller impact.

While task structure is an interesting topic that is highly relevant to human error and routine procedural action, it will not be the focus of the experimental studies in this thesis, since it has been addressed extensively in earlier work. However, since the experimental tasks are organised hierarchically and consist of several subtasks, it is inevitable that the task structure will have an impact on the error patterns. Where necessary, this will be discussed in the experimental work.

2.5.5 Slip Errors and Step Times

While there have been a number of different studies that have investigated human error in the laboratory, this remains inherently difficult. One issue often encountered is that participants tend to make very few errors when performing a well-learned task, especially in a laboratory setting where they may be especially inclined to self-monitor. Different researchers have attempted to increase error rates using different techniques, such as introducing interruptions (e.g. Li et al., 2008), adding working memory load (e.g. Byrne and Bovair, 1997) or using highly engaging experimental tasks to prevent excessive self-monitoring (e.g. Chung and Byrne, 2008).

Ruh et al. (2010) have noted that, in support of error rates, it is also highly useful to investigate step times in the study of routine procedures. These can provide a more fine-grained measure of performance than error rates, because the latter are often too low to statistically analyse or their distributions are skewed and non-normal. Ruh et al. (2010) argue that errors "can provide only indirect clues to the mechanisms that support the typically error-free performance of routine action". Indeed, using step times as an additional measure "provides the more direct and sensitive source of information about the underlying processes".

Ruh et al. (2010) collected a large volume of data of performance on a routine, hierarchically organised task. They found that step times were longer at branch points, compared to within-subtask step times. Moreover, this effect became more pronounced when a secondary task was added: the step times at branch points increased further while within-subtask step times remained stable. Interestingly, error rates, on the other hand, were not affected significantly by the increased load of the secondary task. Ruh et al. (2010) interpret this dissociation between step times and error data as indicating that step times are the more sensitive and fine-grained measure. Moreover, in relation to task structure, they argue that the links between within-subtask steps must be stronger, leading to faster performance. At the branchpoints, more deliberate, higher-level processes are involved, which are more time-consuming. This work nicely illustrates how the additional measure of step times can be used to gain further insights into routine procedural performance than by error rates alone.

In short, it is highly beneficial to collect step time data in addition to error data, for practical purposes related to data analysis, as well as for theoretical purposes. As such, the experimental work presented in this thesis will consider both.

Having discussed the theoretical accounts of how routine procedures are executed correctly and how errors are made, the chapter will now discuss how this can be applied to the robust error patterns frequently observed on routine procedural tasks. The next section discusses a number of basic error mechanisms found in the cognitive models, and how these can give rise to a persistent pattern of slip errors.

2.6 Error Mechanisms

The current thesis aims to explain why some steps in a task appear to be more prone to errors than others. As discussed in the introduction (and in more detail in the next chapter), it is hypothesised that device-oriented steps are more prone to errors than task-oriented ones. However, before a hypothesis can be developed about the underlying causes of these errors, it is important to consider the different, basic ways in which errors can occur. As discussed in the previous section on the cognitive models, there are a number of basic mechanisms through which slip errors can occur. This section briefly highlights these. They are further developed in the next chapter, which investigates the cause of the errors in more detail.

Low Activation Levels

In the M-f-G model (Altmann and Trafton, 2002; Trafton et al., 2011) and the IAN model (Cooper and Shallice, 2000, 2006), errors can occur because the target goal or schema's activation level is too low. This means it is less able to compete with others, and occasionally loses out. As a result, it is unable to direct behaviour, and an error occurs.

There are several ways in which a goal or schema's activation can be too low. For instance, it may receive too little activation from its parent or the environment. Alternatively, a goal or schema may initially be sufficiently activated, but lose this activation before the associated action can be executed. This can occur for instance through interruptions (see section 2.5.2), or a delay between the formation of the goal and its subsequent execution. In the IAN model, it is also possible that the parameters that govern the balance of activation sources are skewed, which means that inappropriate activation sources are over-represented.

High Activation Levels on Inappropriate Steps

Another way in which errors can occur in the M-f-G and IAN models is when an incorrect schema or goal gains too much activation. This means that the inappropriate action 'overtakes' the correct action, and is executed instead of the correct action. This approach is highly similar to the previous one in that the activation of the target step is lower than that of the incorrect step. However, instead of the correct action, the erroneous one is the cause of the error in this scenario.

There are a number of more detailed mechanisms that can lead to this scenario, which are very similar to those that lead to low activation levels. For instance, the incorrect step may receive too much activation from the various activation sources. Alternatively, the balance between the activation sources may favour those that lead to incorrect activation patterns.

Degraded Context Representations

The SRN model (Botvinick and Plaut, 2004) takes a fundamentally different approach to errors. In this model, errors occur because the representation of the current context is degraded. This sometimes leads to a lapse into a different context that is similar to the current one, resulting in an error. The main mechanism through which such degradation occurs is the application of noise to the system.

These three mechanisms describe the main causes of errors in the cognitive models discussed in this thesis. Note that this is not intended to be an exhaustive analysis of all possible error mechanisms, but merely a consideration of the most common ones, that may be able to account for the errors observed in the Doughnut task. Moreover, an issue that this chapter does not consider in detail is the underlying causes of errors (as opposed to the *mechanism* through which they occur). This is addressed in detail in the next chapter on device- and task-oriented steps.

2.7 Conclusion

The current chapter has discussed the existing literature on human error, and the correct execution of skilled action sequences. The type of error this thesis is concerned with is the slip error, which is defined as an error in the execution of a task. Slips occur despite the person having the correct knowledge. They are persistent and cannot easily be eliminated. As such, they are deeply interesting to human factors specialists and cognitive scientists alike.

This chapter analysed three different cognitive models that account for skilled sequential action. For each of the models, the mechanisms for the correct and erroneous execution of action sequences were discussed, and its strengths and weaknesses were analysed. The M-f-G model explains routine behaviour as a chain of associatively linked goals or steps, in which each provides activation for the next. Errors can occur when the activation level of the correct goal is not high enough to surpass the interference level. The M-f-G model is capable of explaining how goal-directed behaviour occurs, and a later version also accounts for routine sequential behaviour.

The IAN model argues that routine sequential behaviour is governed by hierarchically organised schemas, which are interactively activated through self, lateral, top-down and environmental influences. The schemas are ordered by triggering and pre- and postconditions, which allow the gating of activation sources. As in the M-f-G model, errors can occur when the activation value of a schema is not high enough, or when objects or resources are unavailable at the time of schema selection. The IAN model is capable of showing flexible behaviour, and can explain a wide variety of errors. As such, it is well suited to account for routine sequential behaviour.

The SRN model is unique in that it uses distributed, graded patterns to account for sequential action, rather than explicit representations of goals. Action sequences are produced by the changing patterns of active and inactive interconnected units. Errors occur when these patterns are degraded and another learned action sequence takes over. Strengths of the SRN model include the ability to share information, its intrinsic ability to account for memory load, and its elegant mechanism for capture errors. However, the SRN has some serious shortcomings: it does not generally execute sequences that it has not learned before (including making errors that were not part of the training sequence), and it currently does not account for the timing of actions. As such, while it is a promising and interesting approach, this model needs further

development before it can accurately predict behaviour.

This chapter has provided a background for the current thesis by discussing how routine procedures are executed, and how errors can be made. However, as discussed in the introduction, this thesis is concerned not just with how errors can occur, but also why some steps in a procedure are more prone to errors than others. More specifically, it addresses the hypothesis that steps that are not related to the task goal are more likely to be forgotten than steps that are directly relevant to the goal. This has not been addressed in the current chapter. Rather, the next chapter investigates these device- and task-oriented steps, and proposes a hypothesis to explain why they may be more prone to error.

Take-home message:

The current work focusses on slip errors in routine procedural tasks. Three cognitive models of routine action, the M-f-G model, the IAN model and the SRN model, are capable of making a variety of errors. The main hypothesis tested in the current thesis is that device-oriented steps are more problematic because they have lower activation levels than their task-oriented counterparts.

Chapter 3

Device- and Task-Oriented Actions

In this chapter:

- Previous research on device and task-oriented steps, and related concepts, is discussed.
- Based on this, a working definition is developed.
- This definition is further developed in a small qualitative study.
- Based on the previous research, the predictions for the experimental work are discussed.

3.1 Overview

The main focus of this thesis is on why some steps in a procedure are more prone to errors than others. More specifically, it looks at one such factor that has been argued to play an important role in the occurrence of slip errors: some steps in a task are crucial to the achievement of the task goal, while others are merely "extra steps imposed by the device" (Li et al., 2008). This chapter looks at this factor in more detail.

While there has been some prior work on concepts similar to this, the notions of device- and task-relevant steps are inherently ill-defined. There is no single, agreed definition, and the terminology used differs from study to study. Therefore, the first aim of this chapter is to remedy this, by analysing the concepts in more detail and proposing a unified definition.

The second aim of this chapter is to investigate how these concepts can be used to explain the robust error patterns found on some experimental tasks (e.g. Li et al., 2008). The previous chapter discussed how errors occur. Three models of routine procedural action were introduced, that each make predictions about the origin of slip errors. Memory representations of an incorrect action can have an activation level that is too high, providing too much competition for the correct goal. Alternatively, an action may have an activation level that is too low to compete with other memory representations. Another error mechanism is the degradation of context representations, which leads to the occasional lapse into a different context. The knowledge gained from the previous chapter will be applied to the concepts of device- and task-oriented

steps. This leads to a hypothesis of why device-oriented steps may be more problematic than their task-oriented counterparts. It is argued that the most suitable and likely explanation of how errors occur on these steps is that they have lower activation levels.

This chapter first presents a thorough review of the different perspectives taken by different lines of work. Based on this, it develops a working definition. This working definition is then further developed using structured interviews with a number of experts in the field of HCI. This definition is then validated in the next chapter, and forms the basis of the hypotheses tested in the experimental work. The chapter closes with a discussion of how the different error mechanisms can explain the hypothesised differences in performance on device- and task-oriented steps, and presents a number of predictions that are addressed in the experimental work in later chapters.

3.2 Previous Research

The distinction between device and task-oriented steps as presented in this thesis is not a novel one: several previous studies have discussed similar ideas. However, there is little agreement in the existing literature on what is at the core of these concepts. Some studies have approached the issue from a mental model perspective, while others have focussed on concepts such as cognitive salience. The current section discusses these, and other approaches, in detail.

3.2.1 Mental Models

Mental models have long been acknowledged to be important in the successful interaction with devices. One of the earliest studies to specifically investigate mental models comes from Young (1981). He used different types of pocket calculators to demonstrate the different forms mental models can take. He noted that the concept of a mental model is rather hazy, and has not been well-defined in the literature. Young (1981) argued that

> "Central to [a conceptual model] is the assumption that the user will adopt some more or less definite representation or metaphor which guides his actions and helps him interpret the device's behaviour."

Indeed, the role of mental models is not always clear, and there may be several different types (see also Young, 1983). In particular, Young (1981) further investigated two in more detail: those that describe how a device works (further referred to here as device models), and those that link knowledge about the device with knowledge about the task (referred to as mapping models).

3.2.1.1 Device Models

Device models describe knowledge about how a device works (e.g. Kieras and Bovair, 1984; Cox and Young, 2000). For instance, in the case of an RPN pocket calculator as described by Young (1981), this type of model may include a representation of a stack of four registers. The model further explains that numbers can be pushed onto the stack by pressing enter, or popped

by using the operators. This simple model allows the user to understand the behaviour of the calculator when it is used for simple calculations.

Kieras and Bovair (1984) argued that having an accurate device model can aid performance, and makes it easier for people to learn how to use the device. They presented an experiment in which participants either learned a device model of a fictional task, or acquired the task sequence by 'rote' learning. They demonstrated that the participants who were given the device model learned the task faster, were more successful in executing it and were better able to infer novel procedures. Kieras and Bovair (1984) argued that device models are beneficial because they give 'meaning' to the actions in the task sequence.

However, other studies have found no beneficial effects of device models on performance. For instance, Hiltz et al. (2010) describe an experiment on the Doughnut task, in which they attempt to reduce the occurrence of slip errors by providing participants with a device model, amongst others. They found that having a device model did not improve task performance. It should be noted that these two studies used different approaches in measuring performance: Kieras and Bovair (1984) assessed overall task performance, and how well participants learned how to operate the device, while Hiltz et al. (2010) studied error rates on a particular step.

The idea that device models provide 'meaning' for actions in a task sequence is highly relevant to the concepts of device- and task-oriented steps. When the meaning of an action (in a particular task context) is clear, it is easier to learn and better remembered than actions that do not have such meaning (Howes and Young, 1996, amongst others). Meaningful actions are closely related to the concept of task-oriented steps, while 'meaningless' actions are related to device-oriented ones.

However, Hiltz et al. (2010) demonstrated that simply understanding the role of an action in a task sequence does not guarantee that performance on that step is improved. Their device model showed how a particularly error-prone step is relevant to the operation of the device, but it did not provide an account of how it related to the main task goal. Indeed, no improvement in performance on this step was found. As such, it could be argued that the meaning must relate to the task goal rather than to the device.

Another issue to consider is the properties of the device model itself, and how appropriate it is to the task. For instance, one could conceive of a device that is very simple in its structure, but is described by an extremely complex device model. It seems reasonable to assume that such a highly complex model would do little to improve task performance. In other words, the device model should allow for an easy translation of device knowledge into the action sequence that will execute the task. This approach is closely related to the argument put forward in the next section on task-action mappings.

3.2.1.2 Task-Action Mappings

A second type of mental model described by Young (1981, 1983) is the 'mapping' model. These models map knowledge about the workings of the device onto knowledge about what is required

Abstract Machine Arena		**Task Arena**		**Action Arena**
"Execution cycle" of machine	↔	Basic calculation	↔	⟨Terminated-calculation⟩
Function register	↔	Operator	↔	⟨Operation⟩
1st argument	↔	1st operand	↔	⟨1st-number⟩
2nd argument	↔	2nd operand	↔	⟨2nd-number⟩
Execution of computational unit	↔	Evaluation	↔	"="
Result	↔	Value	↔	Answer on display

Table 3.1: *Task-action mapping for a simple calculator, reproduced from Young (1981).*

to do the task, and what actions are necessary to complete it. Young (1981) describes three 'arenas', which represent different perspectives on the use of a device. He distinguishes between the task arena, which describes the task; the action arena, which concerns the action sequence that accomplishes the task; and the abstract machine arena, which describes how the device works. In addition, Young (1981) argues that there are mappings between the arenas, and the mapping between the action arena and the task arena determines how the device is used. He demonstrated the mapping between the different arenas using an example of a calculator. Table 3.1 shows this mapping for a simple calculation using this device, as described by Young (1981).

Young (1981) argues that when the mapping is straightforward, the device will be easy to use and few problems in the task's execution are expected. This is the case for simple calculations performed on the calculator: the table shows that each of the three arenas map neatly onto one another. However, this is clearly not always the case, as mappings can be quite complex and not always clear-cut. If the task includes entities that do not map onto the device, then essentially the device is unsuitable for the task. Conversely, the device may also include additional entities that do not map onto any of the task's entities. These 'superfluous' steps can be seen as device-oriented: they are required for the operation of the device but do not directly contribute to the main task goal.

This approach was further built on by Moran (1983) and Payne et al. (1990), for instance. They developed the notions of 'device-space' and 'task-space', which are very similar to the 'arenas' used by Young (1981). Moran (1983) introduced the ETIT (External Task Internal Task) analysis, which is primarily used to assess system design and to analyse how well knowledge of the system transfers to others. It consists of three parts:

- External task space: all possible tasks that the user could bring to the system.
- Internal task space: all possible tasks that can be carried out on the computer system.
- A mapping between the two, which translates the external task into the internal one.

The better and simpler the mapping between the spaces, the easier the system is to use, and the easier it is to transfer the knowledge to another system.

A more elaborate model comes from Payne et al. (1990) in the form of their Yoked State Space (YSS) hypothesis. This model uses concepts similar to those of Moran (1983) and Young (1981): a goal space is the set of possible states of the external world (the 'real world') that can be manipulated with the device, and a device space is the set of possible states of the device that can achieve those goal states. The states in these two spaces can be mapped onto each other using a semantic (meaningful) mapping, so that the device states represent the goal states. The YSS hypothesis argues that the two state spaces must be 'yoked' to allow this translation.

Device operators are used to transform states in the device space, and consist of one or more basic interactions with the device. Payne et al. (1990) give the example of deleting a string of text in word processing software. To accomplish such a goal, the user must first select the string of text, and then press a key or select a menu option to delete it. Many such basic operations do not affect the 'minimal device space', which represents the minimum states necessary to be able to execute the task. This means that these interactions may not have a 'meaning' in terms of their role in the task. In other words, they only afford an operational account, so they are perceived simply as sequences of steps, in which each individual step does not have a particular meaning. They are generally acquired by rote learning. Conversely, operators that do specify a transformation in the minimal device space allow a figurative account, which means it provides meaning for the steps. This has previously been shown to be important in the acquisition of interactive skill: meaningful steps are easier to learn than meaningless steps (Howes and Young, 1996). Payne et al. (1990) argue that it is also possible for users to elaborate the minimal device space, by introducing new concepts or adjusting their understanding of the device. This allows them to 'make sense' of actions that previously only allowed an operational account. This elaboration then allows a figurative account, and as such can make the action more meaningful for the user.

It could be argued that actions that allow only an operational account are closely related to device-oriented actions. They do not affect the device space, and as such do not make a contribution towards the main goal of the task. Actions that allow a figurative account, on the other hand, are closely related to task-oriented steps. They have autonomous 'meaning' for the user, and the transformation they make in the device space bring the user directly closer to their main goal. Payne et al. (1990) argue that operators that only allow an operational account may be more 'clumsy'.

Payne et al. (1990) stress that each user constructs their own YSS, and therefore the status of individual steps can differ between individuals. They argue that this can be seen as part of procedural skill, so that a more experienced user will have a more developed YSS than a novice user.

In summary, the type of mental model described in this section provides an interesting account of the representation of steps. These task-action mapping models could be very useful

in determining whether steps are device or task-oriented. Moreover, Payne et al. (1990) argued that an action's representation depends on whether it is meaningful with regards to the task. This is closely related to the argument put forward by several other researchers, who approached actions in terms of their contribution towards the main goal. This is discussed in the next section.

3.2.2 Contribution to the Task Goal

Goals have long been recognised to play an extremely important role in the execution of tasks (e.g. Newell and Simon, 1972; Norman and Shallice, 1986; Altmann and Trafton, 2002). When performing a task, people often work towards achieving the main goal of the task, and indeed, the absence of such a goal sometimes leads to errors (e.g. Byrne and Bovair, 1997). However, it has been argued that not all steps contribute equally to the main goal (e.g. Kirschenbaum et al., 1996; Li et al., 2008; Hiltz et al., 2010). This section discusses the contribution steps make towards the main goal, and how this relates to the distinction made in the current thesis.

In his paper on errors in VCR programming, Gray (2000) discussed the notion of display-based difference reduction, also known as hill-climbing. It reflects the tendency of humans to attempt to reduce the difference between the current state of the task and the goal state they wish to achieve. Gray (2000) argued that this can be a useful strategy when operating devices with a highly visual display, as it allows the minimal use of memory and planning. While the current work does not make claims about how big a role hill-climbing plays in the execution of routine procedural tasks, the strategy demonstrates the tendency of humans to favour steps that make the largest (perceived) contribution towards their main goal.

Building on the prior work on goal space and tool space to study usability, Kirschenbaum et al. (1996) introduced the concepts of 'tool-only' and 'task-tool' steps. Tool-only steps are defined as "operations that have no direct relationship to accomplishing the task" but that are more concerned with "fiddling with the tool" (Kirschenbaum et al., 1996). Clearly, the concept of a tool-only step is highly similar to the concept of a device-oriented step. This contrasts with task-tool steps, which work towards the goal of the task, and as such can be seen as related to task-oriented steps. Kirschenbaum et al. (1996) argue that the more tool-only steps in a task, the worse usability, especially for several tool-only steps in a row.

Li et al. (2008) presented a similar argument, arguing that some steps in a task sequence are "extra" steps that are imposed by the device. They contrast these with steps that are "integral to the 'natural' task sequence". Like the tool-only and task-tool steps presented above, these are closely related to the device- and task-oriented concepts presented in the current work. Li et al. (2008) further argue that such 'extra' steps may be represented in the mental context in a different manner. Building on work by Cox and Young (2000), they propose that these steps rely on device knowledge, while the steps integral to the task sequence rely on task knowledge instead. Cox and Young (2000) describe device knowledge as "a collection of facts about what the device as a whole (or parts of it) do". Note that this is highly similar to the notion of a device model described in the previous section on mental models. In turn, they describe task

knowledge as "knowledge about how to complete a task using a particular device".

Li et al. (2008) further argue that the way these steps are remembered is different. They argue that task-oriented steps rely mostly on associative cuing, as described by Altmann and Trafton (2002). Each step in a well-learned procedure provides activation for the next, through associative links. This allows routine procedures to be executed rapidly and without much conscious control. However, Li et al. (2008) argue that this associative cuing mechanism may not apply to steps that are not integral to the task sequence. Instead, these must rely on "a more deliberate and less automatic system".

3.2.2.1 Cognitive Salience

Back et al. (2007) first proposed the idea that actions can "spring-to-mind". While they do not provide a clear definition, these actions can be seen as capturing attention. They argue that an inappropriate step 'springing to mind' can lead to errors. While Back et al. (2007) do not discuss in detail what makes a step spring to mind, they focus mostly on visual features in the task interface. For instance, they mention the example of a text entry box, which can prompt a user to enter text into it even though this may not currently be the appropriate action.

Later work by Hiltz et al. (2010) related the springs-to-mind concept to cognitive features of the task as well. They argue that how much an action springs to mind determines its 'cognitive salience'. This is a rather hazy concept, for which no clear definition exists as yet. Hiltz et al. (2010) draw an analogy with visual salience, which refers to how well an object stands out from its background, and how distinct or obvious it is. In the same way, steps that are cognitively salient can be thought of as having a distinct meaning and being important to the task. In this context, task-oriented steps can be seen as having high cognitive salience, because they are important to the task goal. Device-oriented steps, on the other hand, can be seen as having low cognitive salience, because they are not meaningful in relation to the task goal.

Hiltz et al. (2010) equate a lack of cognitive salience with a lack of relevance to the main task goal. Moreover, they argue that the low cognitive salience on some steps in a task procedure can lead to errors, such as those that "do not directly [move] one closer to their main goal, but [are] a necessary step imposed by the device design". Hiltz et al. (2010) present examples, such as the device-initialisation error, which occur because of the step's lack of relevance to the primary task goal, which in turn leads to it having low cognitive salience. Other examples of errors that they argue are related to a lack of relevance to the main goal are PCEs.

Huang et al. (2011) further build on the notion of cognitive salience, arguing that those actions that are directly related to the main task goal have higher cognitive salience than those that are "only incidental actions required by the specific device". Huang et al. (2011) developed a formal model of the Doughnut task (see section 4.2.1 for a description of this task), and showed that cognitive salience played an important role in determining which steps were prone to errors and which were not.

3.2.2.2 Communication Goals

Blandford and Young (1998) introduced the concept of communication goals, which they define as "a task dependent mental list of information the user knows they must communicate to the device". They describe the example of using an online banking system. The user of such a system knows they have to identify themselves to the system, for instance by entering their account number and password. Therefore, in this task, two of the communication goals are to convey the account number and PIN. Similarly, when using an infusion pump, the communication goals would be the amount of drugs to be infused, and the infusion rate, for instance.

This idea is closely related to the notion of goal relevance. Communication goals can be seen as relevant to the main goal of the task. The user knows they have to accomplish the communication goal, as without it the task is not complete. Moreover, these communication goals are generally independent of the device used, though their order and format may differ between different devices. As such, actions that correspond to a communication goal are likely to be easily remembered.

3.2.2.3 Enabling States Analysis

May et al. (1993) introduced the concepts of "enabling goals" and "work goals" in relation the design of usable systems (see also May et al., 1992a,b; Whitefield et al., 1993). They define a work goal as "a particular desired state of the attributes of the domain objects" (Whitefield et al., 1993). Similarly, tasks that achieve these work goals must " change the states of the work domain objects closer towards their states as specified in the work goal". Enabling goals, on the other hand, define a desired state of the *system*, rather than of the work that needs to be achieved. As such, enabling tasks "are not directly associated with the work goal, since none of them changes the states of the work domain objects" (Whitefield et al., 1993). Instead, they put the system into a state in which progress towards the goal can be made. Simply put, enabling tasks require the user to do things "for the sake of the computer rather than to achieve their real task goals" (Whitefield et al., 1993).

It is clear that the concepts of enabling tasks and work tasks are highly similar to the distinction between device- and task-oriented steps made in this thesis. Enabling tasks, by definition, do not do any 'real work', and thus do not make a direct contribution towards the task goal. Work tasks, on the other hand, are directly related to the main goal of the task, and as such are highly similar to task-oriented steps. However, May et al. (1993) offer a novel perspective on what is meant by 'goal relevance'. In the Enabling States Analysis, both work goals and enabling goals are considered to be goals, as demonstrated by their terminology. Thus, enabling tasks are still relevant to a goal, but the difference is that this is an *enabling (or device)* goal, rather than the main task (or work) goal. While this does not contradict the concepts of device- and task-oriented steps as discussed in this thesis, it is somewhat at odds with the notion of goal relevance. In this light, the term 'goal relevance' should be interpreted as relevance to the *main* goal or a work goal.

To illustrate their distinction, May et al. (1993) present the example of getting cash from an ATM machine. The interactions required to achieve the task goal are represented in figure 3.1. The diagram indicates that only the final accomplishment of the main goal, getting the cash, is considered to be a work goal, while all other goals are considered enabling goals. Note that this is at odds with the very similar example given by Blandford and Young (1998) in the previous section on communication goals. Many of the enabling goals, such as specifying the amount of money desired or entering a PIN number to verify one's identity, are considered by Blandford and Young (1998) to fulfil communication goals, and therefore relate directly to the main goal. While indeed these actions do help to get the system into a state in which it will dispense the money, they are highly relevant to the task, are associated with a clear communication goal and are not just required because of the particular design of the device. Thus, it is not entirely clear how the example provided by May et al. (1993) supports their argument. Nevertheless, it does highlight an important issue: what is considered to be part of the task goal is of great importance to the classification of steps as enabling or work tasks (or device- and task-oriented steps).

Figure 3.1: *Enabling States analysis of the interactions required when using an ATM machine. The goals are represented in italics, while actions are underlined (S indicates a system action, U indicates a user action). Reproduced from May et al. (1993).*

In short, this section described the relevance of actions to the main goal in relation to the concepts of device- and task-oriented steps. It was argued that device-oriented steps have low relevance to the main goal, and as such their cognitive salience is lower. Task-oriented steps are directly relevant to the main goal of the task, and many of these accomplish communication goals. As such, their cognitive salience is higher. The next section describes the effects this has on the performance on routine tasks.

3.2.3 Behavioural Effects

An important claim tested by this thesis is that device-oriented steps are more problematic in their execution than their task-oriented counterparts. While these steps have not previously been compared directly, there is some previous work that experimentally investigates related issues. These are reviewed in the current section.

3.2.3.1 Mode Errors

One class of slip errors that has been studied experimentally is the mode error (Monk, 1986; Sellen et al., 1992). This error occurs when a person forgets to appropriately set the mode of the device to their current mode of operation. This can happen when the person misclassifies the mode of the device, or they can simply forget to change it. Examples include forgetting that the caps lock key is on when typing in a password or other text, or trying to type in text into a UNIX text editor while it is in command mode.

Arguably, changing the mode of a device can be seen as a device-oriented step. After all, it is only required for the operation of the device, and does not make a direct contribution towards the main goal. Moreover, modes are generally specific to particular devices and are often not present in physical tasks. As such, a mode error can be seen as involving the omission of a device-oriented step.

Monk (1986) and Sellen et al. (1992) present experimental work on mode errors. Monk (1986) reports on an experiment that aimed to reduce mode errors using sounds. Participants played a simple game, and the different modes in the game were indicated with sounds of a different pitch. He found that this approach was successful in reducing mode errors. This finding indicates that the awareness of the current mode plays an important role in whether or not an error is made. However, while effective, using sounds as an indicator may not be the most practical solution, since this can be disruptive in the workplace.

Sellen et al. (1992) investigated different kinds of feedback, to assess which type of mode indicator is the most effective. First, they compared visual feedback with kinaesthetic feedback, in which participants used a foot pedal to change the mode of the system. They found that both types of feedback reduced the occurrence of mode errors, although kinaesthetic feedback was more effective than visual feedback. In a second experiment, they studied the kinaesthetic feedback in more detail. They found that actively maintaining the mode by keeping the foot pedal pressed down (or up) was more effective than switching by a simple click of the foot pedal. Sellen et al. (1992) explained their findings by arguing that the active feedback is user-

maintained, rather than system-maintained, and as such the user is more aware of the modes.

Closely related to a mode error is the unselected window problem, which involves forgetting to select the appropriate window in a graphical computer interface before interacting with it. This is arguably an instance of the mode error, and indeed Tesler (1981) has argued that "windows are only modes in sheep's clothing". According to Rieman et al. (1994), the unselected window problem is a common occurrence in modern computer interfaces. They argue that the error occurs because the re-selecting subgoal is never formed, or loses its activation before it can be executed. In this case, it is unable to compete with a more strongly activated goal of "doing real work" (Rieman et al., 1994). This illustrates how the unselected window problem relates to the contribution to the task goal discussed in the previous section: selecting the target window is not relevant to 'real work' or the main goal of the task. As such, this can be seen as a device-oriented step.

3.2.3.2 Task and Tool Errors

In his work on VCR programming, Gray (2000) argued that "device-specific goals are the hardest to remember". Moreover, Rasmussen (1987) argued that a frequent source of error are actions that are not a logical part of the task sequence, but are instead isolated. However, there is little experimental work that directly addresses these claims, and as such this is a major aim of this thesis.

Kirschenbaum et al. (1996) presented one of the first studies that explicitly studied errors on task steps and tool-only steps. As argued in section 3.2.2, task-tool steps are analogous to task-oriented steps, and tool-only steps are analogous to device-oriented steps. Kirschenbaum et al. (1996) presented an experiment on the distribution ("how the tool-only operations are distributed among the tool-task operations") and ratio ("the relative effort that the person spends on managing the tool rather than using it to perform the task") of tool-only steps in a routine procedural task. They hypothesised that the more tool-only steps, the weaker performance. Moreover, they argued that performance would be affected by more tool-only operations *in a row*, because it reduces the attention available for the execution of the task. While presenting some data on the tool-only steps in a realistic nuclear submarine task, their results provide little insight into their hypotheses. Moreover, Kirschenbaum et al. (1996) do not make any further predictions about performance on single tool-only steps.

In their experiment on post-completion errors, Li et al. (2008) also observed another frequent error, which involved the omission of a device-oriented step at the start of a subgoal. They termed this a subgoal-initialisation error. However, another error was possible at the same point in the task sequence: the incorrect ordering of the subgoals of entering data into the widgets. Li et al. (2008) termed this a subgoal-sequence error. It can be argued that the subgoal-initialisation error is a device-oriented error, because it involves the omission of a step that is not integral to the task sequence. Conversely, the subgoal-sequence error, while occurring at a point in the task where a device-oriented step must be executed, can be seen as a task-

oriented error, because participants seem to have forgotten the correct task sequence, rather than the device operation. Indeed, subtask-initialization errors were far more frequent than subtask-sequence errors. Interestingly, however, Li et al. (2008) found a differential effect of interruptions on these two types of errors. An interruption just before the step in question did not affect omission errors on this step, while it did lead to an increase in the number of sequence errors made. This is an extremely interesting finding that indicates that these errors may arise from different underlying causes. Li et al. (2008) explain that the interruption disrupts associative links between steps. Since the correct execution of the task sequence relies upon these links being intact, interruptions increase the chance that the task sequence is disrupted. However, steps that are not integral to the task sequence do not rely on these associative links as much, but instead depend upon more deliberate processes. As such, a disruption of the associative links will not affect their execution to the same extent. In short, Li et al.'s (2008) study provides evidence for the idea that different types of knowledge are responsible for different types of errors.

Hiltz et al. (2010) presented an experiment using the same experimental task as used by Li et al. (2008). They investigated the role of goal relevance in the occurrence of the device-initialisation error, which has previously been shown to be very frequent and persistent. Hiltz et al. (2010) tested different approaches in making this action more cognitively salient. In the first condition, they provided participants with a device model similar to the one used by Kieras and Bovair (1984, see also section 3.2.1.1). In the second condition, they gave participants a new goal, to which the device-initialisation step was directly relevant (tester condition). In an additional third condition, participants were also given this new goal, and were explicitly told to ignore other task goals (tester-enhanced condition). In the control condition, participants received the basic instructions for the task, but no device model or altered goal. Hiltz et al. (2010) found that providing participants with a device model was not effective in reducing the error rate on the device-initialisation step in comparison to the control condition. They argued that the beneficial effect of being provided with a detailed device model is limited to learning the task and problem solving situations. They also found that the altered task goal alone was not successful in increasing performance, while in the tester-enhanced condition error rates *were* lower than in the control condition. However, it should be noted that in this additional condition, the working memory load of the task was likely to be much lower, because participants did not have to enter the correct data. It is possible that this caused the reduction in error rates, rather than the altered task goal. Nevertheless, Hiltz et al. (2010) interpreted this as evidence that the goal that is being focussed on plays an important role in task performance.

3.2.4 Cognitive Modelling

The previous chapter introduced a number of cognitive models of routine procedural action, and discussed how they explain the occurrence of errors in routine procedural tasks. As such, it will be beneficial to discuss any modelling attempts that relate to goal relevance, cognitive salience

and related ideas. While the models discussed in chapter 2 do not discuss these ideas, there have been a small number of related modelling studies on the subject. This section discusses these few existing modelling approaches.

3.2.4.1 Soar Model

Churchill and Young (1991) developed a Soar model of calculator use. They provided the model with device knowledge and task knowledge, respectively, and tested the effects this would have on task performance. They found that giving the model knowledge about how to do the task allowed it to successfully execute it, but the model was unable to finish the task when starting at an arbitrary point in the sequence. On the other hand, providing the model with device knowledge, or instructions about how the device works, allowed the model to complete the task successfully, regardless of whether it started at the beginning or half-way through. This model was able to represent the current state of the device, rather than the current action, allowing it to be more flexible.

At first glance, these findings may seem to contradict the argument put forward in this thesis, as the device knowledge leads to more successful performance than the task knowledge. However, the task knowledge essentially trained the model by rote, while the device model allowed it to develop problem solving knowledge of the task. Churchill and Young's (1991) findings are similar to those by Kieras and Bovair (1984). As discussed in section 3.2.1.1, Kieras and Bovair (1984) provided half of their participants with 'rote' training, while the other half were given a device model that taught them how the device worked. Kieras and Bovair (1984) argued that the device model was more successful because it provided meaning to the steps in the procedure, while the rote instructions did not.

3.2.4.2 The IAN Model

Another cognitive model that addresses similar concepts comes from Cooper and Shallice (2000). They describe the IAN model (discussed in detail in section 2.4.2.3), a model of routine procedural action that is capable of explaining a variety of errors. In a subsequent paper, Cooper and Shallice (2006) describe the concepts of crux and enabling/tidying actions. Crux actions are described as "critical behaviors" that are "more important to successful completion of the routine than others". These actions are essential to the completion of the task, and together the crux actions achieve the main goal. As such, crux actions cannot be omitted. The other actions serve to enable the execution of the crux action, or they may serve a "subsidiary" function such as tidying up the task environment to enable further actions. Enabling and tidying actions can occasionally be omitted if the situation allows. Indeed, these descriptions are highly similar to the concepts of device- and task-oriented steps described in the current work.

However, Cooper and Shallice (2006) do not make an attempt to discuss how these different actions are represented and how they may influence error rates. Their main function in the IAN model is to explain how it can operate more efficiently, by omitting unnecessary actions. For instance, Cooper and Shallice (2006) present the example of adding sugar to tea, which requires

removing the lid from the sugar jar, using a spoon to add a scoop of sugar to the tea, and replacing the lid. Removing and replacing the lid can be seen as enabling and tidying actions, while the adding of the sugar to the tea is a crux action. When adding two scoops of sugar to the tea, the lid does not have to be replaced and subsequently removed again after the first scoop. Instead, the IAN model is flexible enough to omit these enabling and tidying actions when they are not necessary. As such, it appears that the concepts of crux and enabling/tidying actions are introduced to make the model operate more efficiently, rather than to represent the different types of steps in a task sequence.

3.2.4.3 Activation-Based Approach

The previous chapter discussed the role of activation in a number of prominent cognitive models of routine procedural action. It argued that low activation levels can lead to the omission of steps. A number of other studies have made similar arguments. For instance, Byrne and Bovair (1997) developed a cognitive model of the post-completion error, and argued that this slip occurs because the PC step has lost its activation before it can be carried out. Indeed, their model is capable of making PCEs based on this mechanism.

Rieman et al. (1994) studied the unselected window problem, which they see as a failure to execute a subgoal that is not involved in doing "real work". They argued that the error occurs because the subgoal is never formed, or because it loses its activation before it can be executed. This makes it difficult for the window-selection subgoal to compete with other goals of doing 'real work', which are more strongly activated. Indeed, the argument Rieman et al. (1994) are making is very similar to the argument made in the current thesis for the activation of device- and task-oriented steps. Steps or goals that are not related to the main goal of doing 'real work', have lower activation levels than those that do, making it harder to compete.

This is also supported by Hiltz et al. (2010), who argued that the frequent device-initialisation errors found in their experiment are caused by a low activation level on these steps. Moreover, they argued that this low activation level is due to the low cognitive salience associated with these steps, which in turn relates back to their lack of contribution towards the main goal of the task.

3.2.5 Summary and Working Definitions

The previous sections described the existing work on device- and task-oriented steps and their related issues. The different approaches taken vary widely, and as such the concepts of device- and task-oriented steps, and what is at their core, are hazy and ill-defined. However, it is crucial for the current thesis to build upon a clear, well-grounded definition. The current section summarises the discussion of previous work, and proposes a working definition.

Despite the lack of unity between the different lines of work, there are a number of points on which the various lines of work seem to converge:

Some steps in a task sequence are not relevant to the task goal. The different previous studies have approached this in different ways. For instance, the work on task-action

mapping showed that some steps necessary to operate the device do not map onto a goal state (Young, 1981; Payne et al., 1990). They do not have a meaning in terms of the task goal (Howes and Young, 1996). Others have argued that some steps do not form an integral part of the task sequence (Li et al., 2008), or do not fulfil any communication goals (Blandford and Young, 1998). Still others have directly argued that some steps do not contribute to the main goal (Kirschenbaum et al., 1996; Hiltz et al., 2010), and do not do any 'real work' (Rieman et al., 1994).

These steps will be more problematic. Steps that do not have a meaning in the task sequence can only be acquired by rote learning (Payne et al., 1990), which has been found to be less effective (Kieras and Bovair, 1984). They will be more difficult to remember (Gray, 2000), and do not 'spring to mind' (Back et al., 2007; Hiltz et al., 2010), resulting in low cognitive salience (Hiltz et al., 2010; Huang et al., 2011). More deliberate processes may be necessary to remember them (Li et al., 2008). They can be disruptive to task performance (Kirschenbaum et al., 1996; Monk, 1986; Sellen et al., 1992), and are prone to errors (Li et al., 2008; Hiltz et al., 2010). More theoretical work has argued that such actions have low activation levels (Rieman et al., 1994; Hiltz et al., 2010), making it more difficult to compete with other goals (Back et al., 2007).

Based on this analysis, the following working definitions are proposed:

> *"Device-oriented steps are those that are more concerned with the correct operation of the device than with the achievement of the main goal."*

and:

> *"Task-oriented steps are those that are more concerned with the achievement of the main goal than with the operation of the device."*

3.3 Qualitative Study

A brief qualitative study was conducted to further support the development of the definitions. The study took the form of semi-structured interviews with a small number of experts in the field of HCI. The definitions proposed in the previous section are evaluated by a number of experts in the field. This has two benefits: first, the definitions serve as a vehicle for experts to discuss their views and opinions about the concepts at hand, and second, they are evaluated and validated in order to develop more grounded, definitive definitions. The next section presents this expert study.

3.3.1 Approach

Semi-structured interviews were conducted with four expert participants. All participants were male and were researchers or academics. While all participants were familiar with the concepts 'device-oriented' and 'task-oriented', their expertise on the subject varied somewhat. Two participants were familiar with the concepts, but were not actively engaged in research in the

area. One participant had published and evaluated some work in the area, whereas the last had published extensively on related concepts.

It should be noted that the terms 'device-*specific*' and 'task-*specific*' were used throughout the interviews, instead of the 'device-*oriented*' and 'task-*oriented*' generally used in this thesis. The change in terminology was one of the outcomes of this study, and as such will be discussed in detail in the discussion.

Participants were each asked the following questions:

- How would you, in your own words, describe your views on the distinction between device- and task-specific steps?
- Can you come up with examples for both?
- Can you come up with an example of a step that is ambiguous? Can you explain why this step is ambiguous?
- How would you go about classifying steps as one or the other? How would you decide which is which? What criteria would you use?
- How would you classify the following steps and why?
  - Switching on the oven before baking a cake
  - Confirming your entry after putting quantities into the dough port
  - Using the Windows XP start button to bring up a word processor
  - Removing the original after making copies
  - Cleaning the Wicket Doughnut machine after making doughnuts
  - Taking in the landing gear after taking off with a plane
  - Activating the compartments on the doughnut task? How is this different from other power buttons?
- Do you think there are any other factors that may affect how steps are classified?
- [When given the definitions:] Do you agree with these definitions? How are they different from your own views? How would you change/improve them?
- Is there anything else you would like to add?

The interviews were recorded using a laptop with voice recording software. The screen of the laptop remained blank throughout the study so as not to distract participants (they were aware their voices were being recorded). The total duration of the interviews varied between 20 and 40 minutes.

3.3.2 Analysis and Results

The interviews were transcribed and then coded. Codes were then grouped into categories, and the results are divided into a number of main themes.

3.3.2.1 Device-Oriented

The first major theme concerns participants' views about the device-oriented or device-specific concept. Two major viewpoints were identified, and an additional one proposed and subsequently refuted by one participant.

The first main viewpoint is that the main factor in device-oriented steps is that they are specific to a particular device. One participant argued that if a step is found on multiple devices, it should be considered task-specific, whereas if it is found on only one device, it should be considered device-specific:

> *"[The step] wouldn't be there if you weren't doing it with this particular device."*

The same participant discussed further scenarios in which device-specific steps may arise, for instance when a step is done automatically on one device, but manually on another:

> *"You could imagine a scenario where the aircraft would do it itself once it reaches a certain altitude, [...] which would suggest that it is a bit devicey."*

A second main viewpoint is that device-oriented steps are those that do not directly bring the user closer towards their main goal. Instead, this type of step is required for the operation of the device only. Participants argued that device-specific steps are outside of the main task goal, or outside of the golden path.

> *"That's device, because it's outside of the main task."*

> *"It is quite a clean device-specific step, because it's not really related to performing the task."*

> *"A device-specific step [is] something that you have to do to the device, but isn't necessarily part of the golden path to complete the task."*

This viewpoint seems to be the predominant one amongst the experts: when asked specifically which view they favoured, three out of four participants argued for this view.

A third view is that device-oriented steps are those at the lowest level in a hierarchical analysis of a task. This is because the actual interaction techniques may often differ between different devices, even if they accomplish the same (sub)goal.

> *"[...] specific to the actual interaction techniques of that particular device."*

For instance, entering a number into a device can be done by pressing number buttons, turning a dial or clicking '+' or '-' buttons using a mouse. The way this action is carried out can be seen as device-specific. It should be noted that the participant expressing this view added that this is probably not an appropriate way of looking at device-oriented steps, as it would make the prediction that the vast majority of steps are device-oriented.

Participants had some difficulty coming up with examples of device-oriented steps. This is perhaps illustrative of the idea that there is not an 'iconic' example of a device-oriented step,

like there is for the PC step (e.g. removing the original after making copies). A probable reason for this is that device-oriented steps, by nature, are more likely to occur on a single type of device only.

Device-oriented steps are contrasted with task-oriented steps, and most of these issues are reversed for those. The next section will discuss the issues involved in task-oriented steps, as identified by participants.

3.3.2.2 Task-Oriented

Similar to device-oriented steps, two major viewpoints on the task-oriented concepts were identified.

Using the first perspective, task-specific steps are required on all devices, and are thus independent of the device. One participant strongly linked this to automaticity: if a step is done automatically on one device, and must be done 'manually' on another, then it is device-specific, even if the step makes a direct contribution towards the goal.

> *"If in say 20 years we make planes that do this automatically, so you don't need to [execute the action] yourself, but then the pilot gets into a plane where you do have to [execute the action] yourself, then that seems much more device-specific, because it's particular to that device."*

Using the second perspective, task-oriented steps make a direct contribution towards the task goal, and are concerned with the achievement of the main goal.

> *"If a particular action is tied closely to achieving a goal, so it moves you towards a goal state directly, then I would suggest that that's more likely to be a task-specific step."*

> *"It's strongly relevant to a task goal."*

One participant argued that task-oriented steps are on the 'golden path', which is the sequence of steps that are critical to completion of the main goal.

> *"Task-specific steps seem easier to define, when I think about them as being restricted to the golden path between you starting a task and finishing it."*

Apart from these two major viewpoints, a number of other factors and characteristics were also brought up by the participants. They were highlighted as properties that are often observed on device- or task-oriented steps. The next section discusses these.

3.3.2.3 Factors and Strategies

Participants discussed a number of factors that they saw as important features of device- and task-oriented steps. They related this directly to their strategies for determining whether a step is device- or task-oriented.

One strategy that emerged is to compare the procedure for doing the task on a specific device to a procedure for doing the same task by hand. If a step is also required when doing the task by hand, it is task-oriented, but if not, it is device-oriented.

> *"If it corresponds clearly to something you would do in the task domain, even if you weren't doing it by computer [...], then I tend to think of it as task."*

> *"If you can do it yourself, then all those steps are kind of tasky steps."*

One participant argued that this approach works as a rough guide, but perhaps not in all cases. It may be problematic for those tasks that do not have a 'real world' equivalent. For instance, the type of task that is done using modern software packages such as Adobe Flash or Illustrator cannot be done 'by hand'.

Another strategy is to determine whether the step makes a difference in the real world. If it does, it is task-oriented, and if it does not, it is device-oriented.

> *"There's [a] notion of having an effect in the world, it's very tightly aligned with this kind of tasky side of devices."*

> *"It only makes a difference on the interface to the device, so therefore it's devicey and not tasky."*

Some steps have a clear effect in the real world, such as clicking the 'Print now' button after filling out the printing dialogue box, because this action actually makes the printer print:

> *"But certainly that clicking on that final print thing says ... make it print! So it's doing your task."*

> *"It's that final OK that actually gets the sprinkles or the whatever it is or the cutting done."*

A further strategy that one of the participants mentioned is to consider whether the designer of the device had a choice of including a particular step. If the step could have been omitted if different design choices were made, then the step is device-specific.

> *"If there are conceivably alternative ways, whether it doesn't have to be there or not. [...] Whether the designer has a choice, I think that should maybe be more defining."*

Participants also noted that whether or not a step is required, or necessary, may or may not play a role in the classification of steps. They acknowledged that this may be a factor, where task-oriented steps are associated with being necessary, and device-oriented steps with being optional.

> *"There are necessary steps and optional steps."*

However, they further argued that device-oriented steps, too, can be required.

"Often it's very necessary, it's just that it's necessary because of the device, not necessarily because of the task."

"They're necessary to get the task accomplished, on the device, but they don't actually do the task."

As such, the participants argued that perhaps this is not a strong factor in determining whether a step is device- or task-oriented.

It was also argued that the experience a user has with a task may influence the way the steps are perceived.

"Your view of the task can be influenced by the device you use."

However, participants differed in the extent to which they thought experience plays a role. Some participants argued that device-oriented steps only emerge when users have used more than one device to accomplish a certain task.

"You need a variety of devices in order to get the kind of basis for the distinction, that seems extremely plausible to me."

"If you only use one [device] then you might get used to that interaction sequence, and think Oh that's just the way it's done, whereas if you use lots, then you realise that there are different options and they're particular to certain devices."

Moreover, as one participant noted, it may not so much be the *level* of experience, but rather the *breadth* of experience.

Two participants made the observation that the level of analysis may play a role. They argued that at higher levels in a goal hierarchy, goals are more task-oriented, whereas on the lowest level, the actual steps that are executed are device-oriented. On this lowest level, the way task-oriented goals are achieved differs between devices, and thus this is device-specific.

"Different devices will have their device-specific way of entering that device-specific step."

"Maybe that lower level of granularity seems more like device-specific. It depends on how finely grained you want to make the distinction."

"There could be variability around the golden path in terms of how you can interact with the device and achieve the same sort of sequence of actions."

However, it was added that this may not be a fruitful approach, because this often leads to different classifications than all other classification strategies.

As these factors and strategies illustrate, a step can have both device-oriented and task-oriented properties. This introduces a certain amount of ambiguity to the concepts. The next section will discuss this in more detail.

3.3.2.4 Distinction and Ambiguity

One of the main findings of the expert evaluations is that the concepts are not binary or clear-cut, as demonstrated by the following quotes.

> *"It is not necessarily a clear binary distinction."*

> *"It's a real distinction, well it can be a binary one, but maybe there's sort of a bit more graded one."*

> *"I think it might be a relative concept, so a comparative one."*

This was further demonstrated by several findings.

First, participants came up with various factors that affect a step's classification, and mentioned that steps may have properties that are characteristic of the opposite sides.

> *"Preparatory actions are indeed where one might have problems, because [...] it's something that you have to do to the device, but it's to enable it in order to achieve the goal you want."*

> *"I would say that an on-button would be an example of an action which could be device- and task-specific."*

Second, the language participants used to describe certain steps very clearly illustrates that they do not see them as binary. This is illustrated by phrases such as:

> *"[It] is kind of almost a bit devicey."*

> *"It's beginning to edge towards the devicey thing."*

> *"It was a bit tasky as well."*

Third, participants had some difficulty classifying certain steps.

> *"My instinct is to say device-specific, but it's a bit of both, because you can't make any progress unless you do that."*

> *"I would say that wasn't just devicey, I'd say that it was a bit tasky as well."*

Even if participants were able to provide a firm classification of a given step, this often took a considerable amount of time and effort. That said, it should be noted that the steps participants were asked about were chosen because it was anticipated that they were difficult to classify.

Additionally, the classification of steps was not always consistent across participants. For instance, one participant classified 'confirming the Dough port entry' as device-oriented:

> *"I would suggest [that] was a device-specific step, because it's complementary to entering data, so you could imagine a device where, once you finish completing the field, it would automatically recognise that that step had been completed."*

whereas another classified it as task-oriented:

> *"It can be seen as consummatory, and therefore VERY tasky."*

One participant argued that the ambiguity is partly due to the two possible interpretations of the terms 'device-specific' and 'task-specific'.

> *"You compare those [hierarchical task] analyses for each device, and see how they differ. And see if there are overlapping elements. [...] but this is where the shades of grey come out, because you end up having device-specific steps which just so happen fortuitously to be common across several devices. [...] There could be another task that has just an additional step that actually is quite core to the task, but somehow is overlooked in the other designs."*

Interestingly, this seems to assume that the step in question is firmly device-oriented, and the different classification strategies are tested on this. This raises the question how this step was classified as device-oriented in the first place, since the participant argued that other strategies may wrongly indicate that a step is task-oriented.

The previous themes were all identified by participants, when discussing device- and task-oriented steps. Taking all these factors into account, participants were then asked to comment on the working definitions presented to them; this is discussed in the next subsection.

3.3.2.5 Definitions

Participants were given the working definitions as described in section 3.2.5. All participants explicitly indicated that they agreed with the definitions, even the participant who had argued for a definition based on a step being specific to a particular device. However, some participants argued that the exact wording of the working definition could be improved. The word 'more' was highlighted as a positive addition, because it highlights the non-binary nature of the concepts.

> *"First of all I would pick out the fact that you've put in the word more. You haven't said that it's concerned with, you said more concerned with. Which is, that's fine, you're hedging yourself with this idea that it's not a binary distinction, I'm fine with that."*

> *"More concerned with, because it is almost an admission that it's not a clear distinction, but a distinction of emphasis."*

One expert argued that the word 'main' was not necessary and possibly even wrong, because a step can also contribute to a subgoal and still be task-oriented.

> *"Maybe you should drop the word main. So then it's more like with the achievement of a task goal. Because then it could be a goal at any level, it doesn't have to be the top level."*

Also, the same participant argued that the word 'direct' should be added to the definition, as it is possible for steps to indirectly contribute to the goal, for instance by putting the device in a state in which direct progress can be made.

> *"The word that was not here but it was beginning to sort of bounce up in my mind, was the word directly. [...] I think a step can be very devicey, even though it's strongly relevant to a task goal; it's just it's not* directly *concerned with [the task goal]."*

Participants also noted that the label 'device-specific' may not be accurate, since it seems to infer that the step is specific to a particular device, rather than concerned with the operation of the device.

> *"I think the label device-specific tends to infer that it's an interaction property with whatever tool you're using."*

The participants also pointed out that the labels do matter, as they influence the way people perceive the definitions.

> *"The labels I think bear an influence on people's understanding."*

Overall, the expert evaluation identified a number of themes and factors that the experts believed were important for the concepts. On most points experts seemed to agree with one another, and differences of opinion manifested themselves mostly as different emphasis on different factors. The next section links the expert evaluations with the existing literature, and provides an overall discussion. The working definitions are then adjusted to reflect this discussion.

3.3.3 Discussion and Revised Definition

This section presents a detailed analysis of the issues that arose in the interviews with the experts. In doing so, it integrates these findings with those from the literature presented at the beginning of this chapter.

3.3.3.1 Different Types of Steps

Two major viewpoints were identified in the expert study: device-oriented steps are specific to a particular device, or they are only required for the operation of the device.

It can be argued that the first approach is problematic in several ways. First, it is not clear how a step would be classified if it is required on some devices but not others. To what extent can a step be classified as device-oriented if it is required on half of the devices that are used for a certain task, but not on the other half? It is not clear whether the line should be drawn at two devices, or all but one, or somewhere in the middle. Moreover, this approach implies that the classification of a step may shift when new devices are encountered that have alternative sets of steps. If a certain step was previously required only on device A, and now a new device B also requires it, the step in question may no longer be considered device-oriented, but is now task-oriented.

Second, it is not clear how this approach fits into theories of cognition. There is no reason to believe that steps that occur on one device only are represented in the mind in a different manner than those that occur on more devices. It could be argued that those steps that occur on one device only are less well-learned, and therefore represented less strongly. However, this would predict that, over time, training can eliminate this difference, which is a view not generally supported (e.g. Byrne and Bovair, 1997; Rieman et al., 1994).

Third, it predicts that a difference between device- and task-oriented steps only arises when a person has experience with more than one device. When doing a novel task, steps are neither device- nor task-oriented, and should as such result in equal performance. This is not generally what is found in experimental settings using novel tasks, such as the Phaser task (e.g. Byrne and Bovair, 1997) and the Doughnut task (e.g. Li et al., 2008). Of course, this does not take into account other factors that may affect performance, such as the properties of the task interface. This issue is addressed in the experimental work presented in this thesis, particularly work using the Frankenstein task (see chapter 7).

Given these difficulties with this approach, the specificity to the device is not thought to be a major factor in the distinction between device- and task-oriented steps. Indeed, none of the existing literature mentions the specificity to the device as a factor that affects steps. It is possible that this viewpoint was elicited by the previous terminology used, which described steps as 'device-specific' and 'task-specific'. These terms seem to imply that the specificity is at the core of the concepts. As such, more suitable terms will be used instead; this is further discussed below (see page 65).

Nevertheless, the approach of specificity may be useful in the classification of steps. While not a core concept, it cannot be denied that many device-oriented steps *are* in fact specific to a particular device. As such, this approach may be useful for a rapid classification of steps, although further analysis using other criteria would be necessary.

The second approach is a better supported one. It argues that steps that make a direct contribution towards the main goal are task-oriented, and those that do not are device-oriented. This approach is more concerned with the nature of the actions rather than with the device. An advantage of this approach is that it is not dependent on a comparison of multiple devices; instead, each individual step can be assessed without having to check if this step is also present on another device. As a result, this approach allows for the classification of steps on tasks where only a single device is available, or for situations in which users have only ever used one device. Moreover, the classification is unlikely to change substantially if users gain experience with additional devices.

Another advantage of this approach is that it can be linked with current theories of cognition. Goals play an important role in cognition, as illustrated by many models of goal-directed cognition, such as the M-f-G model (Altmann and Trafton, 2002) or the IAN model (Cooper and Shallice, 2000, 2006). It is thus not surprising that steps that are goal-directed

may be different from those that are not. Moreover, the second viewpoint is also highly compatible with the previous literature on these concepts. As the analysis of the literature showed (see section 3.2), the vast majority of previous work recognised the significance of a step's contribution towards the main goal as a major factor (e.g. Li et al., 2008; Kirschenbaum et al., 1996; Hiltz et al., 2010; Rieman et al., 1994).

However, a disadvantage of this approach is that it is more subjective. The classification of a step may differ depending on the user's perception of the task. Therefore, this approach may not give a 'definitive' classification for any given step, but there is a certain amount of subjectivity that must be taken into account. This issue is also addressed in chapter 4, which studies how well the definitions developed in this chapter can be used to classify steps in the experimental tasks.

3.3.3.2 Strategies and Factors

The expert study identified several strategies participants used for deciding whether a step is device- or task-oriented. First, they argued that doing the task 'by hand' can reveal which steps are device- and which are task-oriented. While this may work for some tasks, it becomes problematic with tasks that do not have a 'manual' variant (such as printing a document), or for tasks in which the use of a device dramatically alters the procedure or the steps that need to be completed (such as sewing by hand, which uses a single thread, and using a sewing machine, which uses two). Similarly, participants argued that whether or not a step makes a difference in the real world is also a useful guide in the classification of a step. This approach is closely related to work on task-action mappings (e.g. Young, 1981): if a step specifies a transformation in the real world, then it is task-oriented. Again, this approach is subject to the same difficulties: it becomes problematic when the 'real world' and the device are not closely aligned or when the task does not have a real-world equivalent. Another issue that arises with this approach is the timing of the effect in the real world. Using the example of printing a document, specifying the number of printouts is a very task-oriented step. However, this step does not have an effect in the real world until the final 'print' button is clicked. While this does not have to be a problem per se, it may be a source of confusion and as such should be a consideration.

Second, another strategy is to consider whether the designer of the device had a choice of including a particular step or not. Essentially, this strategy checks whether a step is crucial for task completion, or whether it is only required for the operation of the device. It can be seen as a practical way of assessing whether or not a step makes a direct contribution towards the main goal. However, like the previous approaches, there may be steps in which this approach becomes problematic. For instance, switching on a device cannot be avoided in the majority of cases, yet whether this type of step is device- or task-oriented is subject to debate. Another difficulty with this approach is that the designer often has a choice of including additional functionality. For instance, some automatic breadmakers have the option to set the colour of the bread (light, medium or dark), whereas others do not. This is directly concerned with the main goal of

making bread, and not so much with operating the device. However, the designer does have a choice in whether to include this 'colour of the bread' option, even though it would not be required for successful completion of the bread. Another problem with this approach is that some steps can be done automatically, even if they are directly concerned with achieving the task goal. For instance, one can imagine an advanced version of the Doughnut task, in which the recipe is automatically entered into the device, making the data entry steps no longer necessary. In this case, the designer had a choice of whether or not to include the data entry steps, even though these are rather goal-oriented according to the four expert participants. Therefore, while this 'designer choice' approach can be a useful strategy in classifying steps, it has its limitations.

Third, participants also noted that whether or not a step is required, or necessary, may influence the classification of a step. However, this approach is deeply problematic, because often steps that are only concerned with the operation of the device, and have no real-world equivalent or effect, are nevertheless required for successful task completion. However, rather than being required for the goal, it is required by the device, which simply does not allow progress until that step has been completed. Moreover, other steps may be optional, but can still be concerned more with the task goal than with the operation of the device. An example is specifying the number of copies when printing off a document: this step can be omitted if the default of one printout is required, but it is directly concerned with the task goal.

In short, while these approaches may be useful for the majority of steps, they are all problematic for some cases. Thus, instead of using just one, it may be beneficial to use a combination of different approaches when deciding on the classification of a step.

3.3.3.3 Ambiguous Distinction

Most previous studies on the concepts of device- and task-oriented steps have treated them as binary (e.g. Kirschenbaum et al., 1996; Back et al., 2007; Li et al., 2008). However, it is not clear whether this is intentional, or whether it is simply not discussed for clarity's sake. Indeed, all four experts argued that the distinction is not black-and-white, and that a gradient or spectrum may be a more accurate representation. However, the question remains where this ambiguity stems from. First, the contribution towards the goal is partially a subjective experience. As one of the experts mentioned, cleaning the Doughnut machine after making a batch of doughnuts does not directly contribute to your current batch, but it may affect future batches. Moreover, steps such as switching on the device are technically concerned with the operation of the device, but arguments were also made that it directly contributes to the main goal.

Second, users may have varying levels of experience with different devices. A user who has experience with a single device only may be more likely to perceive that every step contributes towards the main goal. In contrast, a user who has experience with a number of different devices may be more likely to feel that the steps common to all devices contribute towards the main goal whereas those that are specific to a particular device do not.

Third, steps may have both properties that are frequently found on device-oriented and on

task-oriented steps. For instance, task-oriented steps frequently occur on many different devices, and can generally not be omitted. However, certain steps may not possess these characteristics, even though they make a direct contribution towards the main goal. An example is again the specification of the bread colour in an automated breadmaker: this is a feature present on only some devices, and can be omitted. Nevertheless, if used, it makes a difference to the actual bread, and as such can be argued to be task-oriented.

These ambiguities highlight the importance of taking all these factors into account when classifying and discussing device- and task-oriented steps. It should be noted that the empirical work in this thesis aims to use steps at the extreme ends of the spectrum as much as possible. This will ensure that the experimental data is clear, and that the results are unambiguous. This is further addressed in the next chapter.

3.3.3.4 Definition and Terminology

The terminology used in the various previous papers describing concepts similar to device-and task-oriented steps has been inconsistent at best. This section critically examines the validity of the terminology, and proposes changes to better reflect their meaning.

Some of the experts argued that the word 'specific' (as used during the study) may be misleading. It gives the impression that the most important aspect of the definitions is that the step is specific to the device or the task. However, as discussed in previous sections, this may not be the most accurate approach to the concepts. Instead, the word 'oriented' may be more suitable, as this does not have the same connotations, and is closer in meaning to 'concerned with'. Moreover, this is the same terminology as used by Cox and Young (2000).

As such, the following terms and definitions are proposed:

> ***Device-oriented steps*** *are those that are more concerned with the operation of the device than with achieving the task goal.*
>
> ***Task-oriented steps*** *are those that are more concerned with achieving the task goal than with the operation of the device.*

These labels and definitions will be used for the remainder of this thesis. The next section connects the findings of the literature review on routine action with the definitions developed in the current chapter. It sets out the hypothesis put forward and the predictions tested in the experimental work.

3.4 Hypothesis and Experimental Predictions

The previous chapter discussed how errors can occur in routine procedural tasks. It was argued that there are a number of general mechanisms in the cognitive models that can lead to errors:

- Low activation levels on the target step
- High activation levels on a competing step

- Degraded context representations

This section discusses in detail how each of these could apply to device- and task-oriented steps, and potentially explain the error patterns on tasks containing such steps.

3.4.1 Low Activation Levels

A common and widely accepted error mechanism is the low activation level on target steps or schemas. This is a plausible explanation for the hypothesised problems on device-oriented steps. Indeed, Hiltz et al. (2010) argued that low cognitive salience on device-oriented steps can be represented by low activation levels. Moreover, Rieman et al. (1994) argued that steps or goals that are not related to the main goal of doing 'real work' (device-oriented steps in the current work), have lower activation levels than those that do (task-oriented in the current work).

This hypothesis leads to several predictions. First, error rates should be higher on device-oriented steps. As discussed before, lower activation levels make it more likely that a step inadvertently falls below the activation level of a competitor. This, in turn, can lead to the step being omitted, and as such an error will occur.

Second, the time it takes to execute a step should be longer on device-oriented steps. The M-f-G model predicts that goals or chunks with a lower activation level take longer to be retrieved than those with a higher activation level (e.g. Anderson et al., 2004; Altmann and Trafton, 2002). Moreover, activation accumulates over time, and this is thought to be slower for device-oriented steps than for task-oriented ones. As such, this hypothesis predicts that device-oriented steps will take longer to execute.

Third, if a device-oriented action leads to an error, this error is more likely to be an omission than if it occurred on a task-oriented step. Activation levels are noisy, and therefore if a device-oriented step has a low activation value, it is expected to occasionally fall below the level of a competitor. The result is an omission error. On task-oriented steps, on the other hand, the activation levels are thought to be relatively high, so they are more robust to this noise. Instead, more has to 'go wrong' for these steps to lead to an error, such as the intrusion of an incorrect step. In this scenario, a capture error may be more likely to occur rather than an omission. Li et al. (2008) presented some preliminary support for this prediction. They found that on a particular device-oriented step, the vast majority of errors were omissions, rather than sequence errors.

Fourth, a high load on working memory should affect device-oriented steps more severely than task-oriented ones. Working memory load can be represented by the number and activity of other items in working memory. According to the IAN model, these are in direct competition with one another. As such, the more items competing for activation, the more difficult it is for a single item to become the most active. Schemas or goals that have a lower activation level will be at a disadvantage, and may be forgotten even more often under a high load.

The current thesis argues that the hypothesis of lower activation levels on device-oriented steps provides a highly plausible account for the difficulties on these steps. Nevertheless, the

alternatives must also be investigated. The next section considers the alternative hypothesis that the difficulties on device-oriented steps are caused by higher activation levels on inappropriate steps.

3.4.2 High Activation Levels on Incorrect Steps

Back et al. (2007) argued that steps that spring-to-mind can capture attention, and if they do so at the wrong moment in a task sequence, this can lead to errors. In terms of activation levels, this can be explained by increased activation levels on the steps that spring-to-mind. Similar to the previous hypothesis, this leads to situations in which an erroneous step has a higher activation level than the target step. However, the difference is that in this case, the error is caused by the highly active step, rather than by the inactive step.

In relation to the predictions described above, this alternative mechanism makes similar predictions about the error rates, step times and effect of working memory load. The high activation on the incorrect step is expected to lead to more errors on the correct step, because it will have to compete with this 'intruder', and may occasionally lose out. Back et al. (2007) argued that this error mechanism can explain the slips observed in their experimental task, since the error-prone steps were followed by a highly salient step. Indeed, this mechanism could also account for the error patterns observed on the Doughnut task, as the majority of the error-prone, device-oriented steps are followed by a highly salient (task-oriented) step.

Moreover, this approach makes similar predictions about step times and the effect of working memory load. If an incorrect step gains too much activation, the current, correct step must become relatively more active to overcome this than if there was no such intruding step. Therefore, it predicts that step times are longer when there is a highly active competing step. Similarly, the highly active incorrect step will be better able to compete with other goals under a high working memory load than steps that have a more moderate activation level.

This approach does not make any strong predictions about the type of error that will be made, because this depends on the intruding step. If the next step in the task sequence is the one that gains too much activation, it is expected to result in an omission error. If *another* step in the task sequence is the trouble maker, a sequence error or other type of error may occur. Note that this is not dependent on the current, correct step.

While this approach seems plausible for explaining certain error patterns, it is problematic as an explanation for errors on device-oriented steps. It predicts that whenever a device-oriented step occurs, *another* step suddenly gains too much activation. It is important to note that this is caused by the other step, rather than by the device-oriented step. This seems highly unlikely: there is no obvious reason why these intruding steps should occur specifically on device-oriented steps. Nevertheless, it is possible that this mechanism can explain the error patterns on individual tasks such as the Doughnut task, as the device-oriented steps on this task are followed by highly salient task-oriented ones. Therefore, this mechanism should be investigated further.

In short, while an approach from higher activation levels on incorrect steps is a plausible account for explaining some error patterns, it is problematic as a general account of why device-oriented steps would be more problematic than task-oriented steps.

3.4.3 Degraded Context Representations

An alternative approach is implemented by the SRN model (Botvinick and Plaut, 2004), and involves degraded context representations. Recall that this model uses graded patterns of active and inactive neuron-like units to simulate routine procedural action. Errors occur because these patterns can be degraded and become more 'vague', for instance through noise. A degraded pattern can come to resemble another pattern, which can lead to a lapse into the other pattern.

In relation to the current work on goal relevance, device-oriented steps can be represented by patterns that are not as strong or clear as those representing task-oriented steps. As argued by Botvinick and Plaut (2004), the more degraded a pattern is, the more likely it is to result in an error. In addition, a high load on working memory can be represented by increased noise on the system, which in turn further degrades the patterns. As such, it is expected that device-oriented steps become more problematic under a high load. However, it is not clear whether task-oriented steps, that are hypothesised to have more pronounced patterns, suffer from this noise to the same extent, or whether, due to graceful degradation, the effect may be less severe.

Contrary to the low-activation level hypothesis, the approach from degraded representations predicts that the majority of errors are capture errors. This is not specific to device- or task-oriented steps, but is a general prediction about the errors the SRN model makes. Note that this capture mechanism can also generate omission errors if the capturing context is the one representing the next step. Also, this approach does not make any predictions about the time it takes to execute steps, as this is not currently relevant to the SRN model.

In short, it is argued in the current thesis that the most likely explanation for errors on device-oriented steps is that they (or their associated goals or schemas) have relatively low activation levels. This approach is thought to make the most plausible predictions about errors, step times and working memory load. Since it is difficult to directly investigate the activation levels on a step (without building a cognitive model), the experimental work presented in this thesis assesses the four hypotheses put forward above. In short:

- Device-oriented steps are expected to have higher error rates than task-oriented steps.
- Device-oriented steps are thought to take longer to execute than task-oriented ones.
- The proportion of omission errors is predicted to be higher on device-oriented than on task-oriented steps.
- Working memory load is expected to affect these three measures more on device-oriented than on task-oriented steps.

Chapters 5, 6 and 7 present the experimental studies that address these predictions. The overall discussion (chapter 8) considers *why* activation levels may be lower on device-oriented

steps, and what the underlying cause of this is. Moreover, it will discuss how the cognitive models presented in the literature review can account for this, by looking at how these models deal with goal-directed action.

3.5 Conclusion

Not all steps in a task are created equal. Some steps do not make a direct contribution towards the goal of the task. Instead, they are imposed by the device, and necessary for its successful operation. These steps are termed 'device-oriented'. They contrast with steps that are integral to the task sequence, and make a direct contribution towards the achievement of the goal. These are termed 'task-oriented'. An important finding of the expert evaluation study is that the distinction is not necessarily binary: some steps have properties of both device- and task-oriented steps.

This chapter argued that at the core of the distinction between device- and task-oriented steps is the notion of 'goal relevance'. In this thesis, the term goal-relevance refers to *relevance* of a step to the *goal* of the task. There is some evidence that steps with a low relevance to the main goal are more problematic. The main hypothesis put forward in this thesis is that this is caused by lower activation levels on these steps. The three experimental chapters (5, 6 and 7) present the empirical work that addresses this.

Take-home message:

Task-oriented steps make a direct contribution towards the completion of the task goal, while device-oriented steps do not - these are concerned only with the operation of the device. Device-oriented steps are thought to be more problematic, because they have lower activation levels. This is expected to lead to higher error rates, longer step times, proportionally more omission errors, and higher susceptibility to working memory load.

Chapter 4

Study on the Classification of Device- and Task-Oriented Steps

In this chapter:

- The experimental tasks used in this thesis are introduced.
- A study is presented in which experts and novices classify each task's steps as device- or task-oriented.
- The classifications are used to validate the definitions proposed in the previous chapter, as well as the experimental tasks.
- The findings show that participants are largely consistent in categorising the different steps.

4.1 Overview

Chapter 3 reviewed previous work related to device- and task-oriented steps, and developed a definition of the concepts. One issue that became apparent during the discussion with the four experts was that the distinction between the two types of steps is not necessarily binary. Some steps have characteristics of both device-oriented *and* task-oriented steps, and their contribution toward the main goal is not always clear.

Indeed, this issue affects even a well-defined type of step like the post-completion step. While many PC steps are clearly identified as such, there are a number of examples that are not as straightforward. For instance, Ratwani and Trafton (2009) describe an example of a post-completion error in which a person forgets to attach a promised file to an email. While this step frequently occurs towards the end of the task and after the email text has been entered, the main goal will not have been completed until *after* the user hits 'Send'. Therefore, the error of omitting the attachment does not fit the description of an error that occurs *after* the main goal has been completed. This example shows that any classification of steps is open to discussion to a certain extent, and thus this issue is not limited to the distinction between device-

and task-oriented steps. Nevertheless, it is important to study this matter in more detail, for two reasons. First, if such a reliable classification is not possible and the distinction proves to be merely theoretical, this may be problematic for the practical application of the findings of this thesis. Second, a reliable classification is important for the tasks that are used in the experimental work presented in this thesis.

There are currently few reliable methods or frameworks for deciding whether a step is device- or task-oriented. Indeed, there have been few previous studies that even explicitly classified steps as such (or related concepts). The only study to do this is by Kirschenbaum et al. (1996), who asked two participants to rate a large number of steps on a real-life submarine operating task. However, the purpose of their study was not to assess how well steps can be classified per se, but to study the ratio and distribution of 'tool-only' steps in their task. Nevertheless, they found that the agreement between the two participants was moderately high.

The current chapter presents a study on the classification of steps into device- and task-oriented categories. The aim of this study is twofold. First, it investigates whether steps can be reliably classified as device- or task-oriented. This serves as further validation for definitions and terminology proposed in the previous chapter. Second, the study focusses on the tasks used in the experimental work of this thesis: the Doughnut task (see chapter 5), the Spy task (chapter 6), and the Frankenstein task (chapter 7). It is important that the steps of these tasks are accurately and reliably classified; this is addressed by the current study.

The chapter starts with a detailed description of each of the three tasks used in this thesis, as these are used in the classification study. The classification study is then presented, in which two experts and two novices independently categorise each step. The chapter concludes with a discussion of the reliability of the ratings, and the success of the tasks.

4.2 The Experimental Tasks

The three tasks used in this thesis are aimed at testing routine performance. Participants practice until they approach expert performance and are very proficient in executing the task. They perform it correctly most of the time and have the correct knowledge to do so. As a result, errors made on these tasks are generally slip errors: the failures occur in the execution of the task.

This section discusses the three primary tasks used in the experimental studies presented in this thesis[1]. The procedures to execute each task are explained in detail, and screen-shots of the task interfaces are shown.

4.2.1 The Doughnut Task

The doughnut task is a procedural task in which participants have to follow a certain procedure to make virtual doughnuts. It was originally developed by Li (2006) to study post-completion errors, and it has since been used in a variety of studies and has seen a number of adaptations.

[1]The secondary tasks used in some of the studies will be discussed in their respective chapters.

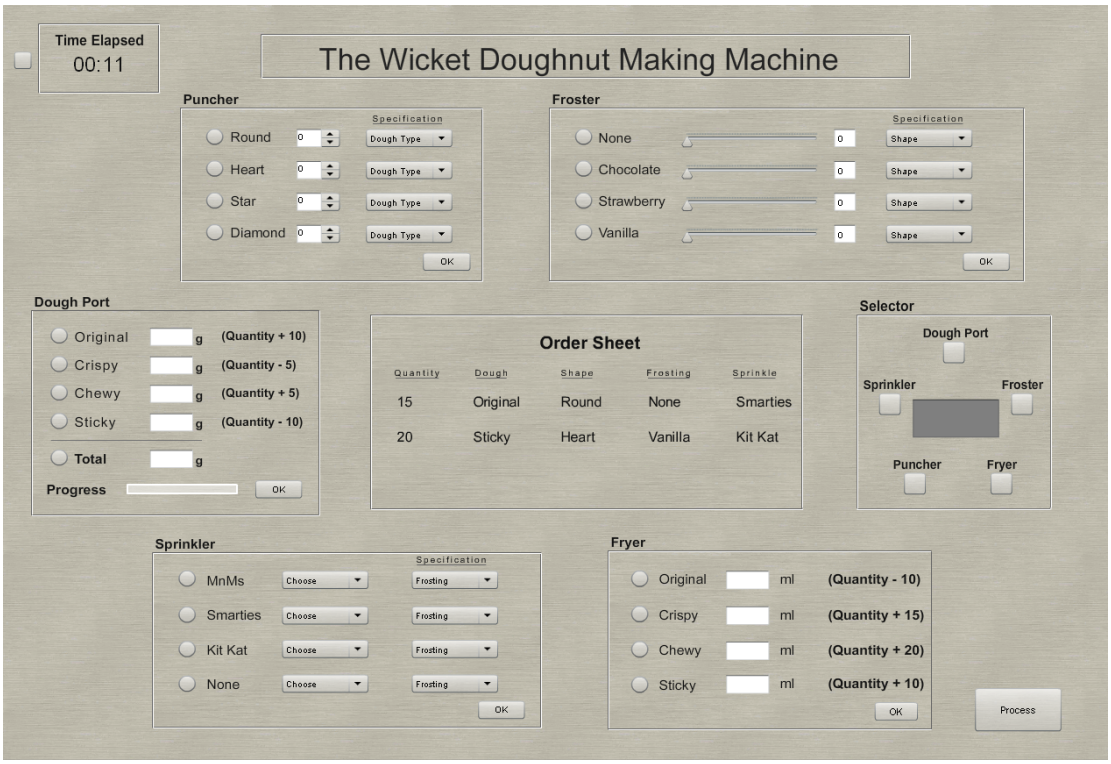


Figure 4.1: *the Wicket Doughnut Making Machine as adapted from Li (2006).*

For instance, Ratwani et al. (2008) used the doughnut task to predict post-completion errors, but adapted the task so that participants made sea vessels instead of doughnuts. Another example is a study by Hiltz et al. (2010), who used the same interface of the doughnut task, but changed the instructions so that participants were testing the machine rather than making doughnuts. Chan (2008) used the original doughnut task to study the influence of motivation on human error.

Figure 4.1 shows the main doughnut machine interface as adapted from Li's (2006) original machine. The doughnut task consists of five compartments, or widgets, in which participants have to enter information about the doughnuts to be made. Information about the doughnuts they have to make is displayed on the order sheet in the centre. There are 5 different compartments, each of which needs different input from the order sheet. The widgets need to be operated in a certain order:

Dough Port → Puncher → Froster → Sprinkler → Fryer

Any other order is not allowed and counted as an error. Before data can be entered, the compartment needs to be activated by clicking the appropriate selector button on the selector panel on the right-hand side. Once all the widgets have been completed, clicking the Process/Clean button will process and make the virtual doughnuts. A pop-up screen then appears, notifying the participant that the doughnuts are ready. This pop-up gives a false-completion signal. After dismissing the pop-up, the machine needs to be cleaned by clicking the appropriate button. A competing signal is also present in the form of a flashing message notifying the subject of the

next call. For an overview of all steps required to complete the doughnut task in the right order, see table 4.1.

In between doughnut-making trials, participants have to obtain the next order by using the call centre. This is a separate task done on a separate computer terminal. During this task, participants receive a call from one of the doughnut shops and have to get the next order from them (figure 4.2). They have to select the doughnut shop from a list, and subsequently on a map. After confirming, the order is then transferred to the doughnut machine. The purpose of the call centre task is mainly to separate trials on the doughnut making machine. It ensures that the post-completion step is indeed seen as the last step in the task and the device-initialisation step is seen as the first step. Note that recent work by Ratwani et al. (2008) and Ratwani and Trafton (2009) did not use the call centre, and they did not find this to be of influence on post-completion or device-initialisation error rates. To further clarify the Doughnut task's structure, figure 4.3 shows a hierarchical task analysis of the doughnut and call centre tasks.

Task Step	**Action Required**	**Type**
0	Respond to a call at the Call Centre	Not classified
1	Show the next order by clicking Next Order	Task
2	Select the Dough Port compartment	Device
3	Enter the data into the Dough Port	Task
4	Confirm the input by clicking OK	Task
5	Wait until the progress bar fills	Device
6	Select the Puncher compartment	Device
7	Enter the data into the Puncher	Task
8	Confirm the input by clicking OK	Task
9	Select the Froster compartment	Device
10	Enter the data into the Froster	Task
11	Confirm the input by clicking OK	Task
12	Select the Sprinkler compartment	Device
13	Enter the data into the Sprinkler	Task
14	Confirm the input by clicking OK	Task
15	Select the Fryer compartment	Device
16	Enter the data into the Fryer	Task
17	Confirm the input by clicking OK	Task
18	Process the doughnuts by clicking Process/Clean	Task
19	Dismiss the report by clicking OK	Not classified
20	Clean the machine by clicking Process/Clean	Device

Table 4.1: *All steps required to complete one trial on the doughnut machine. The last column shows whether a step is device- or task-oriented; this is further discussed in section 4.3.3. Also note that steps 1 and 19 are not classified, this is discussed in chapter 5.*

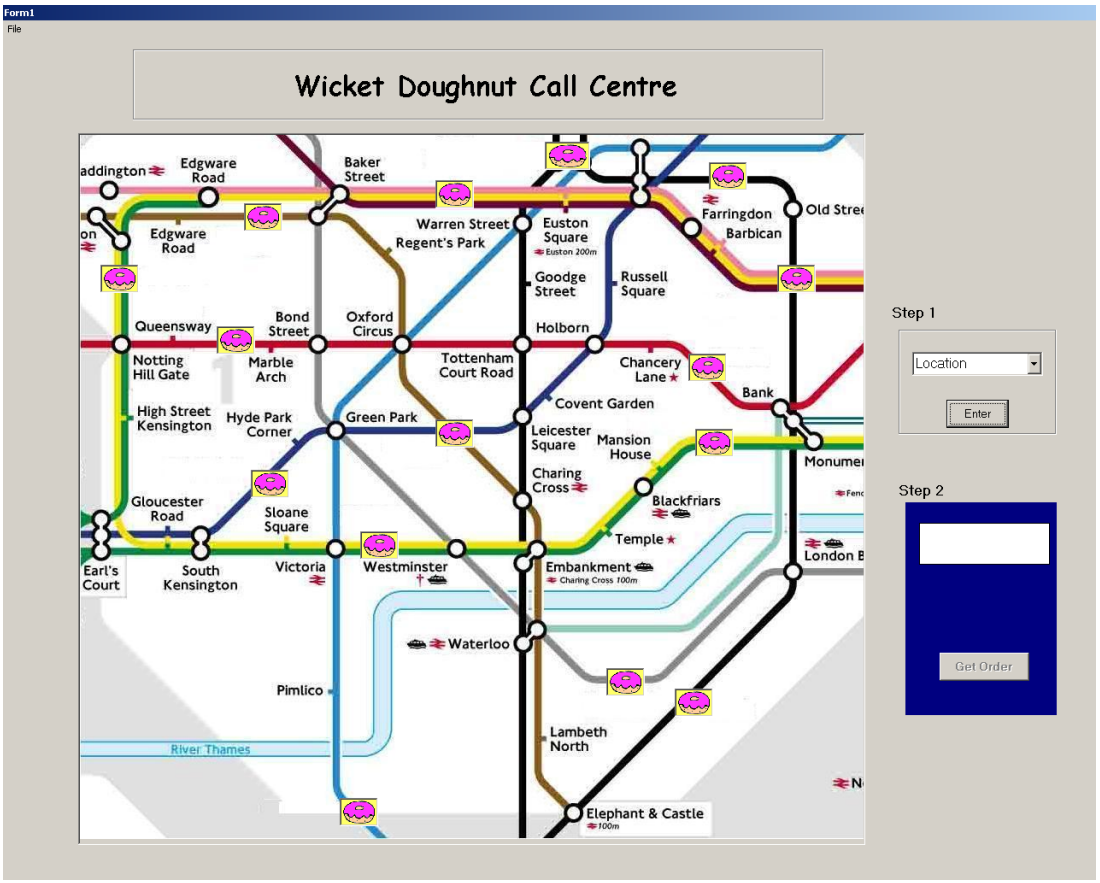


Figure 4.2: *the Call Centre, as developed by Li (2006). In Step 1, subjects select the required location from the drop-down list and click Enter. In Step 2, subjects select the required location on the map and drag it into the white box. Clicking the Get Order button then sends the order to the doughnut machine.*

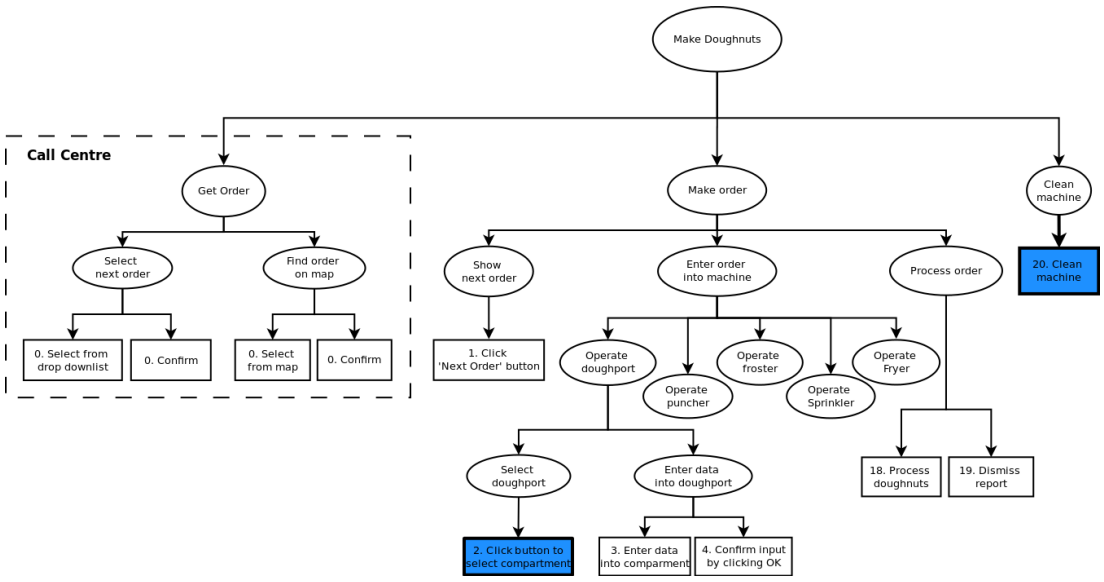


Figure 4.3: *Hierarchical task analysis of the Doughnut task. Steps 2 and 20 are considered device-oriented (see section 4.2.1); both are shaded. Note that the Operate Puncher, Operate Froster, Operate Sprinkler and Operate Fryer subgoals are not defined further to save space; they are identical in structure to Operate Doughport and as such also contain a device-oriented step at the beginning.*

4.2.2 The Spy Task

In the Spy Task, participants assume the role of a spy who must fly a plane to deliver secret messages to fellow spies around the world. The task consists of a sequence of steps that must be executed in the right order. The Spy task has two related parts: flying the plane and delivering the message. The Spy task was developed to incorporate a wide variety of different steps, to counteract any negative effects of the repetitive nature of the Doughnut task (see chapter 5).

The task begins with flying the plane to a specified destination; figure 4.4 shows the plane cockpit. First, participants must prepare for take-off by turning on the cockpit, entering the required destination, and contacting Air Traffic Control with a request for take-off. This last action is in turn divided into three further actions: turning on radio transmission, transmitting the message, and turning radio transmission off again.

Second, participants must fly the plane to its destination. Take-off starts with releasing the brakes, increasing thrust to gain speed, and extending the wing-flaps. After lift-off, the landing gear has to be taken in. The landing procedure starts by decreasing thrust to slow down, and speaking to Air Traffic Control once again. The wing-flaps are then extended, and the landing gear is deployed. After the plane touches down, the thrust must be reversed to slow the plane down. Landing is complete when the plane has come to a full stop and the brakes are applied.

Figure 4.4: *the Spy task interface. The secret compartment for sending messages is shown on top of the cockpit window; it is hidden during the first part of the task.*

Third, a 'clean-up' procedure requires participants to switch off the plane, and switch the thrust back to forward again.

The second part of the task involves delivering the secret message. To avoid detection, the interface for the delivery is hidden during the flight. First, the secret compartment must be activated. The encrypted signal must then be turned on, after which the message can be transmitted. After completion, the participant receives a new message and destination. The encrypted signal must then be turned off, and the secret compartment must be hidden. The task then continues with the next trial. Table 4.2 shows an overview of all the steps needed to complete the task. In addition, figure 4.5 shows a hierarchical representation of the task.

Task Step	**Action Required**	**Type of step**
	Flying part	
1	Turn on the main display	Device
2	Enter the destination by selecting from the list	Task
3	Confirm the destination	Task
4	Turn on radio transmission	Device
5	Indicate type of request for ATC (take-off)	Task
6	Send the ATC request	Task
7	Turn off radio transmission	Device
8	Release handbrake	Device
9	Increase engine thrust to full	Task
10	Extend wing-flaps	Task
11	Take in landing gear after lift-off	Task
12	Slow down by decreasing engine thrust to zero	Task
13	Turn on radio transmission	Device
14	Indicate type of request for ATC (landing)	Task
15	Send the ATC request	Task
16	Turn off radio transmission	Device
17	Retract wing-flaps	Task
18	Extend landing gear	Task
19	After touchdown, switch engines to reverse thrust	Task
20	Stop at the secret spy terminal and apply handbrake	Task
21	Switch main display off	Device
22	Switch engines back into forward thrust	Device
	Delivery part	
23	Activate secret transmission compartment	Device
24	Establish an encrypted transmission channel	Task
25	Type in message	Task
26	Confirm to receive new message and destination	Task
27	Turn off encrypted transmission channel	Device
28	Deactivate secret transmission compartment	Device

Table 4.2: *All steps required to complete one trial of the Spy game. The third column indicates whether the step is task-oriented or device-oriented.*

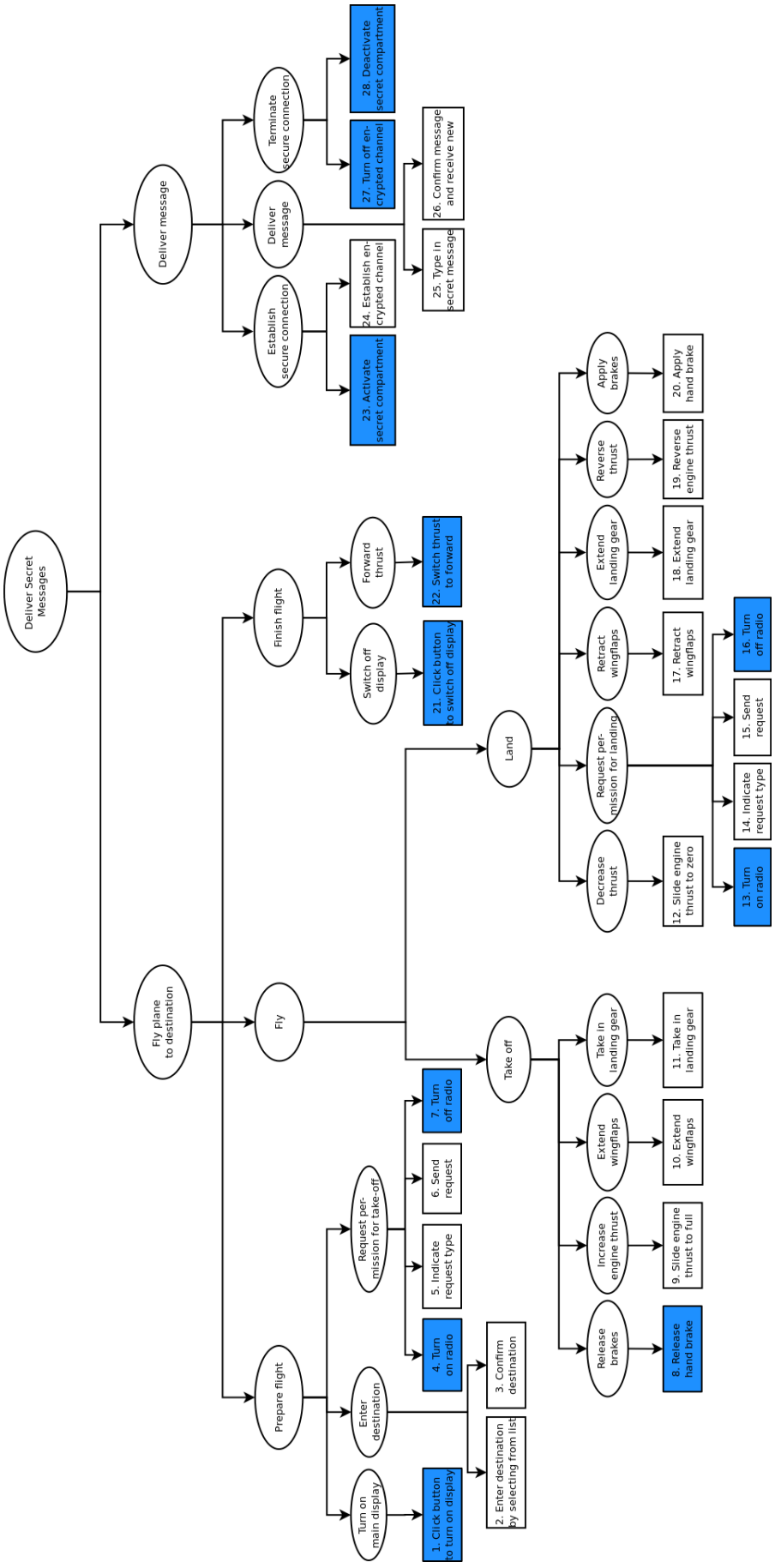


Figure 4.5: *A hierarchical task analysis of the Spy task. Device-oriented steps are indicated in blue.*

4.2.3 The Frankenstein Task

The Frankenstein task is a routine procedural task that was specifically developed to study device- and task-oriented steps. It has two conditions: in the first, participants create monsters according to a set of specifications, and in the second, participants disassemble them instead.

A number of target steps are included that are directly relevant to the task goal in one condition and related only to the device in the other, and vice versa. For instance, in the 'creating' condition, participants have to activate each body widget before it can be used. This is considered a device-oriented step. In the 'disassembling' condition, on the other hand, the same step does not activate a widget, but instead removes the relevant body part from the monster. This step, in turn, is considered task-oriented. Crucially, in both conditions, the same interface is used[2], and the order in which steps have to be carried out is the same. The Frankenstein task contains 10 of these target steps, and for both conditions there are a number of device-oriented and task-oriented target steps. The remaining steps are non-targets, and act as 'fillers' to enhance the task experience and improve the story.

Figure 4.6 shows a screen-shot of the task interface. All task steps necessary to complete the Frankenstein task are summarised in table 4.3. The target steps are highlighted in the middle two columns, and indicate whether a step is device- or task-oriented.

Figure 4.6: *Screen-shot of the Frankenstein task. On the left is the locator, which allows participants to find monsters on the map. On the right is the main Frankenstein interface, which allows them to create or destroy the monsters.*

4.3 Classification Study

4.3.1 Introduction

There is currently no reliable way to determine whether a given step is device-oriented or task-oriented, or indeed if steps can be reliably classified at all as one or the other. Because this is important for the work presented in this thesis, the current study addresses these issues. Using the definitions developed in the previous chapter, the first aim of this study is to assess whether

[2]With the exception of different button labels, and a number of animations were different to fit the relevant story.

	Subgoal	Condition 1	DS/*TS*	DS/*TS*	Condition 2
1		Pick up phone			Pick up phone
2		Get coordinates			Get coordinates
3	Get next monster	**Activate slider**	**DS**	***TS***	**Slide**
4		**Slide**	***TS***	**DS**	**Deactivate slider**
5		Get blueprint at co-ordinates			Get monster at coor-dinates
6		**Activate**	**DS**	***TS***	**Remove**
7	Body / Head	Specify			Classify
8		Confirm			Confirm
9		**Activate**	**DS**	***TS***	**Remove**
10	Legs / Arms	Specify			Classify
11		Confirm			Confirm
12		**Activate**	**DS**	***TS***	**Remove**
13	Arms / Legs	Specify			Classify
14		Confirm			Confirm
15		**Activate**	**DS**	***TS***	**Remove**
16	Head / Body	Specify			Classify
17		Confirm			Confirm
18		**Paint monster**	***TS***	**DS**	**Activate widget**
19		Select from menu			Select from menu
20	Program / Wipe	Debug and program			Confirm choice
21		**Deactivate widget**	**DS**	***TS***	**Wipe monster brain**
22		**Position for blast**	***TS***	**DS**	**Activate**
23	Laser	Operate			Operate
24		**Release monster**	***TS***	**DS**	**Deactivate**

Table 4.3: *All steps required to complete the Frankenstein task. Target steps are marked bold, and the middle columns denote whether the target step is task- or device-oriented for each condition. Note that steps 3 and 4 are repeated twice, once for each coordinate. The remaining steps are not classified, because they were not manipulated experimentally.*

steps can be reliably classified into device- and task-oriented steps. The current study focusses on a small selection of tasks: those used in the experimental work presented in this thesis. While this limits the generalisability of the findings, developing a full framework to classify any given procedural step did not fit within the time-frame of this PhD. The use of the experimental tasks also addresses the second aim of this study: it is important to develop a reliable classification of their steps, as this will help strengthen the experiments and verify that the distinctions made are valid and reliable. Moreover, these tasks have been studied extensively, and as such they are well understood.

To assess whether steps can be reliably classified as device- or task-oriented, the current study looks at inter-rater reliability. Participants are asked to 'rate' steps, and statistical analysis reveals whether these classifications are reliable across participants. Both expert and novice

participants participated in the study. The experts were included to reduce any effects of a misunderstanding of device- and task-oriented steps. Inter-rater reliability not only measures how good the 'rating system' is (i.e. the two categories of steps), but it also measures the ability of the raters to apply it. As such, it was important to minimise any effects of the latter by using participants who were experts in both the concepts being rated as well as the experimental tasks used. This method was also employed by Kirschenbaum et al. (1996), who used two submarine experts to rate steps as 'tool-only' or 'task-tool'.

Two novice participants were also included, who had no experience with the concepts of device- and task-oriented steps or with the experimental tasks. This was done to test how well the concepts of device- and task-oriented steps can be applied by non-experts. If the findings of the current thesis are to have any impact on the design of new devices, it is important that device designers will be able to accurately recognise and classify task steps. Moreover, in case of any potential conflicts between the expert ratings, the novice ratings can be used to resolve them.

4.3.2 Method

4.3.2.1 Participants

Four participants were selected: two novices from among psychology undergraduate students at UCL, and two experts in the field of Human-Computer Interaction. One of these experts also participated in the qualitative study reported in the previous chapter. The novice participants were paid £10 for their time and effort.

4.3.2.2 Materials

The classification of steps as tool- or task-oriented was studied using a card sorting task. Each step was represented on a single card using a screen-shot of the corresponding interaction element. For instance, if a participant was to classify the 'Clean' step on the Doughnut task, the card showed a picture of the 'Clean' button. This visual card sorting task was used to minimise any effects a verbal description may have on participants' classification. While in theory the description of the steps should not matter, it is anticipated that, in practice, a verbal description such as 'click button X' may bias especially the novice participants towards the view that the step is only required for the operation of the device.

Participants were asked to sort the cards into three piles:

- Device-oriented steps
- Task-oriented steps
- Not sure

Each pile consisted of a sheet of paper with the relevant definition and characteristics written on it. The sheets are included in appendix A.1.

The steps to be classified were taken from the tasks used in the experimental work in this thesis. All steps on the Doughnut and Spy tasks were included, whereas only the target steps of the Frankenstein task were included (the filler steps were not classified). It should be noted that participants classified the Frankenstein task's steps twice: once for the *create* condition and once for the *destroy* condition.

4.3.2.3 Design

All participants completed the same tasks. The dependent variable was the classification of a step as device- or task-oriented or 'not sure'. Participants' responses for all target steps were recorded manually. For both pairs of participants, novice and expert, the inter-rater reliability was determined.

4.3.2.4 Procedure

All participants executed the study individually. First, participants were briefed on the experiment, and were given the definitions along with an example of each category. They were then asked to classify a small number of steps to ensure they had accurately understood the concepts and were able to make judgements about them. If necessary, participants were given additional examples or further explanations. The instructions given to participants can be found in appendix A.1.

Participants were trained on each of the experimental tasks. They first read an instruction sheet (see appendices A.2, A.3 and A.4 for the instructions for the Doughnut task, Spy task and Frankenstein task, respectively), after which they observed the experimenter execute one trial of the task. They were then allowed to practice until they had performed two trials without any errors. This ensured that they had a good understanding of what was involved in doing the task, and what the role of each step was. After training on a task was completed, participants were then asked to classify each step, by placing the cards in one of the three piles. The order of the cards did not follow the natural task sequence but was pseudo-random, to ensure steps were considered individually and not compared to the step before or after. The only constraint on the order was that similar steps (such as activating the doughport and activating the puncher on the Doughnut task) were not done consecutively.

Each of the tasks was studied separately from start to finish, so a participant learned the task and classified the steps before moving on to the next task. The order of the tasks was counterbalanced pseudo-randomly between participants; the only constraint was that the two conditions of the Frankenstein task were not done consecutively. This was done to avoid mix-ups between step pairs that used the same button. The total duration of the experiment was approximately 2 hours for the novice participants, and 1.5 hours for the experts.

4.3.3 Results

A total of 19 (Doughnut task) + 28 (Spy task) + 10 * 2 (Frankenstein Create and Destroy) = 67 ratings were recorded for each subject. The 'agreed' ratings are included in tables 4.1, 4.2,

		Participant Pair	
Task	**Measure**	**Experts**	**Novices**
Overall	Percent agreement	79%	70%
	Cohen's κ	$0.61 (p < 0.001)$	$0.50, p < 0.001$
Doughnut Task	Percent agreement	74%	63%
	Cohen's κ	$0.58, p < 0.001$	$0.40, p < 0.01$
Spy Task	Percent agreement	75%	68%
	Cohen's κ	$0.48, p < 0.01$	$0.13, p < 0.005$
Frankenstein Task, condition 1	Percent agreement	90%	80%
	Cohen's κ	$0.78, p < 0.05$	$0.59, p < 0.05$
Frankenstein Task, condition 2	Percent agreement	90%	80%
	Cohen's κ	$0.80, p < 0.01$	$0.60, p < 0.05$

Table 4.4: *Agreement and Kappa values for all four tasks, for both participant pairs.*

and 4.3, as shown above in the descriptions of the task. Moreover, the individual ratings for each step can be found in appendix B.

Raw percent agreement and Cohen's Kappa values (agreement adjusted for chance matches) were computed for each participant pair for each task, shown in table 4.4.

4.3.4 Discussion

This brief study investigated the classifications of steps on the experimental tasks. The aims of this study were to validate the classifications used in the experiments in this thesis, and to determine whether steps can be reliably classified as device- or task-oriented.

Overall, the agreement within participant pairs was moderate to substantial, according to Landis and Koch's (1977) evaluation of κ values. The following sections will discuss each task in more detail.

4.3.4.1 Doughnut task

At a κ value of 0.58, the agreement of experts on the Doughnut Task can be classified as moderate. Interestingly, the disagreement between the experts was only on the 'Confirm widget' steps (4, 8, 11, 14 and 17): one expert classified these steps as task-oriented, whereas the other was not sure. Because there are five of these steps (out of a total of 20), this will naturally affect the κ value quite strongly. Indeed, the experts agreed on all other steps. Interestingly, the novices also disagreed on the 'Confirm widget' steps: one classified them as task-oriented, but the other classified two as device-oriented and was not sure about the other three. This latter result is somewhat surprising, because the five confirmation steps are extremely similar. This finding may further reflect the relative ambiguity of the confirmation step, and is in line with the findings of the previous chapter, in which participants highlighted confirmation steps as potentially ambiguous. In the current thesis, the 'Confirm widget' steps are considered to be task-oriented, because this rating received the most 'votes' overall.

While the Doughnut task was not originally designed to study how goal relevance might impact error rates, the current study demonstrates that the classifications are generally robust. Therefore, from a classification perspective it is suitable for future experiments and can be used with confidence.

4.3.4.2 Spy task

The κ values on the Spy task were lower than those on the Doughnut task, for both the experts and the novices.

On one of the steps, establishing the secure, encrypted channel, no consensus was reached. One expert and one novice classified the step as device-oriented, and the other expert and other novice classified it as task-oriented. As such, the author had to make the final decision about the classification of the step. It was determined to be task-oriented, because the main goal of the task was to transmit a top secret message to a fellow spy. The secrecy of the mission was made abundantly clear throughout the instructions. As such, it was argued that establishing the secure channel was directly relevant to the task, and brought participants closer to their main goal of transmitting the message.

Two further steps were borderline ambiguous. First, confirming the destination was rated as task-oriented by one expert and device-oriented by the other, while the novices classified the step as task-oriented and not sure, respectively. Note that this step is similar to the confirmation steps on the Doughnut task, on which the participants also did not fully agree. As on the Doughnut task, the final classification of this step is task-oriented, because two participants assigned this rating, against only one device-oriented rating. Second, closing the encrypted channel was rated device- and task-oriented by the two experts, respectively, while the novices classified the step as device-oriented and not sure. By 'popular vote', the step's final classification is therefore device-oriented. This contrasts with the 'establish encrypted connection' step, which was classified as task-oriented, even though it operates on the same subsystem, albeit in the opposite direction. Similar to a post-completion step, closing the encrypted channel does not directly help the participant get closer to their main goal, in this case because it has already been completed.

A reason for the lower agreement on the Spy task may be the atypical representation of a plane as a device. It could be argued that a plane consists of several devices, such as a radio, an altimeter, and a navigation system. Aggregating these separate devices into one 'meta-device' may be unnatural, and hence lead to more variation in the classification of the individual steps[3].

Nevertheless, the agreement amongst all participants is well above chance level, and for all but one step an agreed classification could be determined. Therefore, the Spy task can be seen as a moderately successful task to study device- and task-oriented steps.

[3]In hindsight, it is likely that the inexperience of the author at the time of development contributed to the issues encountered on the Spy task. The task was developed very early on in this PhD, and as such, the literature review on device- and task-oriented steps had not yet been fully completed. Therefore, a number of the steps, for instance as discussed in the previous two paragraphs, are not as strongly device-oriented or task-oriented as intended. However, rather than redesigning the task, it was decided to develop a new task (the Frankenstein task), to also control for the task interface.

4.3.4.3 Frankenstein task

For both the experts and the novices, agreement was highest on the Frankenstein task, for both *create* and *destroy* conditions. The κ values suggest substantial agreement, according to Landis and Koch's (1977) guidelines. It should be noted that in both conditions, the experts disagreed on only a single step, yet this was sufficient to bring down the κ value from 1 (perfect agreement) to 0.78. The sensitivity of Cohen's κ value is likely due to the fact that there were only 10 steps to rate in each condition. In the *create* condition, the step in question was step 22, 'Position the laser'. The novices rated this step as task-oriented and not sure, respectively, and therefore the agreed rating was task-oriented. In the *destroy* condition, the disagreement between experts related to step 21, 'Wipe monster brain'. The final classification of this step was also task-oriented, because both novices rated the step as such.

The Frankenstein task was specifically designed to study the effect of goal relevance, and therefore a high agreement was expected. Unlike on the Spy task, each step was scrutinised in detail using the findings of chapter 3 before programming on the task commenced. Each step was designed to be as strongly task-oriented or device-oriented as possible, while keeping in mind the coherence of the cover story. In short, these findings show that the Frankenstein task can be used in the experimental studies with confidence.

4.3.4.4 Overall Ability to Classify Steps

As expected, agreement was higher between expert participants than between novice participants. This likely reflects the familiarity of the experts with the concepts of device- and task-oriented steps, as well as their greater knowledge of the experimental tasks. The agreement between raters reflects both how appropriate the rating system is and how well the raters are able to apply it. Since the experts' agreement was higher, their performance more closely reflects how well the rating system works, and the influence of their ability to apply it is limited. On the other hand, the agreement of the novice participants was lower, indicating that their ratings also reflected how well they were able to apply them. Nevertheless, they were quite successful at classifying the steps, especially considering they had no prior knowledge of the concepts of device- and task-oriented steps or of the tasks.

One type of step that seemed to be more difficult to classify than others is a confirmation step. In both the Doughnut and Spy tasks, confirmation steps were included, and indeed the participants did not always agree on their classification. A possible explanation is that participants can have different mental models of these steps. The 'Confirm widget' step on the Doughnut task, for instance, could be seen from different perspectives. It could be argued that the step achieves the input of the data into the machine, and as such brings the user directly closer to their main goal. Alternatively, it seems equally valid to argue that the data input is achieved in the previous step, and the confirm step is merely imposed by the device. Accordingly, both a task-oriented and a device-oriented classification could be appropriate, depending on the mental representation the participant has. While this could be problematic for the reliable

classification of this type of step, it is in fact perfectly in line with the theory underlying device- and task-oriented steps. If a step is perceived as making a direct contribution towards the main goal, regardless of whether this is consistent with the inner workings of the device, it is likely to be classified as task-oriented (and vice versa). The issue with confirmation steps is that they inherently afford both accounts. As such, the inconsistency may therefore be in the perception of the device and how it works, and not necessarily in the distinction between device- and task-oriented steps. That said, specific instances of confirmation buttons may be more likely to be classified as device-oriented (or task-oriented) than others. For instance, the 'Print' button on a printing dialogue can be argued to be highly task-relevant, because it directly achieves the main goal of printing a document. Going back to the 'Confirm widget' example on the Doughnut task, the final consensus was that the step is task-oriented. This may reflect the erasing of data from the widget, which simulates that the device has successfully accepted the data, and thus the step has achieved a tangible goal.

The current study used only a small number of participants. However, this is not unusual in inter-rater reliability studies. Indeed, Kirschenbaum et al. (1996) used only two expert participants. In fact, the Cohen's κ statistic is limited to only two raters. However, given that the κ values found all reached statistical significance, this is unlikely to be a problem. Indeed, even with this small number of participants, relatively high rates of agreement were achieved.

4.4 Conclusion

In this chapter, the experimental tasks used in this thesis were introduced, and these were subsequently used to study the classification of each individual step as device- and task-oriented. A study was presented that found that participants were generally successful in categorising steps.

From a classification point of view, both the Doughnut and Frankenstein tasks are highly suitable for the purpose of studying device- and task-oriented steps. With a small number of exceptions, the steps were reliably classified by both the expert and novice participants. Agreement on the Spy task was also substantial, although it includes a number of steps that were more difficult to classify. As such, some caution should be exercised when interpreting experimental results on the Spy task. Overall, the current study demonstrates that the three tasks are suitable for their purpose and provided a validation of the different steps.

The study also highlighted that the classification of certain steps is less straightforward than others. A particular type of step that participants agreed on less was the confirmation step. These steps can be approached from different perspectives, which result in different classifications. As such, there may be an element of individual differences to the classification of these steps, as it depends on the mental representation a person has.

The agreement between participants was generally substantial. Of course, these results cannot be taken to mean that all steps and tasks are easy to classify, and that any person will

be able to do so without problems. To be able to conclude this, a much larger study is needed that investigates a number of different tasks beyond the artificial toy tasks used in the current study, and using a larger number of (novice) participants. Specifically, interface and device designers could be tested, since they are a potential target audience for the findings of this PhD. Moreover, a framework that allows the more systematic classification of steps can be developed. However, this is beyond the scope of the current thesis. Nevertheless, the current findings are encouraging and provide an interesting and valuable topic for future research.

Having validated the classification of steps on the three experimental tasks, the thesis will now present the experimental work done. The next three chapters will present six empirical studies that use these tasks, and investigate the hypotheses presented in the previous chapters. They are organised by task: chapter 5 presents two studies that use the Doughnut task, chapter 6 uses the Spy task, and chapter 7 presents the final two experiments that use the Frankenstein task.

Take-home message:

Three experimental tasks are introduced that are used in this thesis: the Doughnut task, the Spy task and the Frankenstein task. A classification study is presented that validates the definitions of device- and task-oriented steps proposed in this thesis. Moreover, the study demonstrates that the steps on the three experimental tasks are classified reliably, and thus the tasks are suitable for their purpose.

Chapter 5

Exploration of the Error Pattern on the Doughnut Task: Experiments 1 and 2

In this chapter:

- The consistent error pattern found on the Doughnut task is studied in more detail in two experiments.
- The results of experiment 1 demonstrate that this error pattern is robust even without interruptions.
- The experiment also finds that the distinction between device- and task-oriented steps is successful in explaining the error pattern.
- Experiment 2 manipulates working memory load, and finds that device-oriented steps are more affected by a high load than task-oriented ones.
- The shortcomings of the Doughnut task as a task of device- and task-oriented steps are discussed.

5.1 Overview

A basic assumption in the current thesis is that errors in routine procedural tasks do not occur at random. The central question asked in this work is whether goal relevance is an important factor that affects the likelihood that an error is made.

One experimental task that has shown a particularly consistent error pattern is the Doughnut task (Li, 2006, see section 4.2.1). This task has mainly been used to study the post-completion error, and has been deployed by a number different researchers (e.g. Li et al., 2008; Ratwani et al., 2008; Chan, 2008; Hiltz et al., 2010; Ament et al., 2011). Figure 5.1 shows the error patterns obtained on three different studies, all using the Doughnut task. Particularly high error rates are seen on the post-completion step (step 20, clean the Doughnut machine) and on the device-initialisation step (step 2, select Doughport). Moreover, the other selector steps (6, 9, 12 and 15) also have error rates higher than most other steps, while steps such as entering data into each widget (e.g. 7 and 10) and confirming it (e.g. 4 and 8) have error rates

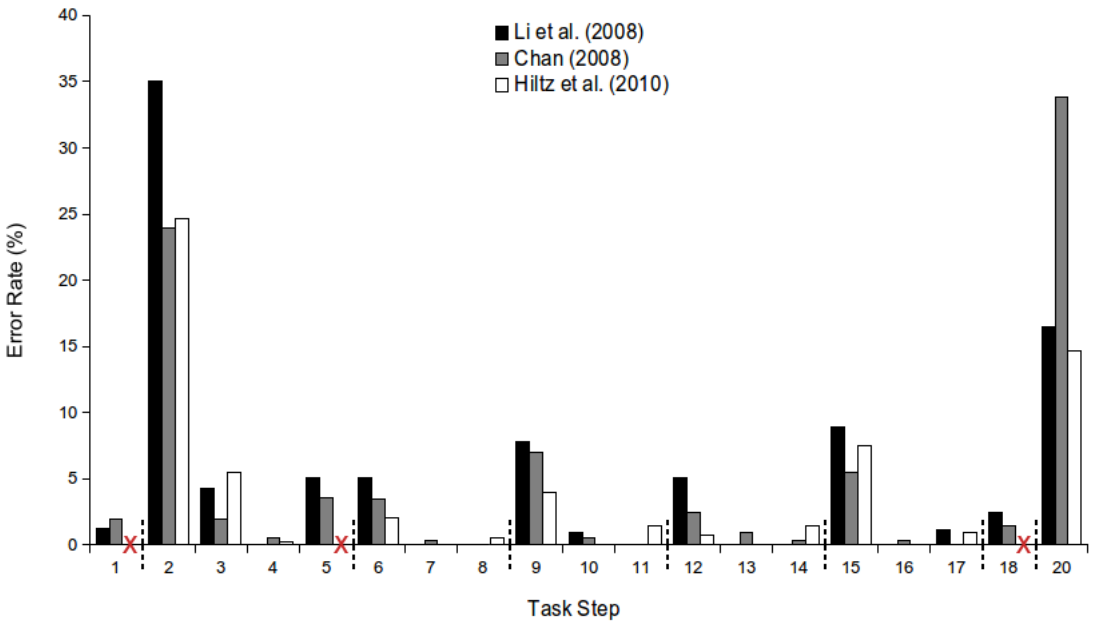


Figure 5.1: *Persistent error patterns on the Doughnut task. The black bars represent Li et al.'s (2008) findings, the grey bars represent Chan's (2008) findings, and the white bars represent Hiltz et al.'s (2010) findings. Note that step 19, dismissing the pop-up screen, is not included. Moreover, Hiltz et al. (2010) did not include the 'new order' step (1), the 'wait for progress bar' step (5) and the 'process doughnuts' step (18); these are marked on the graph with a red cross. Dotted lines on the X-axis indicate subtask boundaries.*

close to 0. While the absolute error rates may be different for each study, the general pattern of more and less error prone steps appears to be present in all three studies.

Li et al. (2008) noted that on his Doughnut task, many of the steps that were most prone to errors were "extra steps imposed by the device", and rely on knowledge of the device rather than on task knowledge. Moreover, they argued that these steps may not be represented in participants' task representations. Indeed, Li et al.'s (2008) characterisation of these steps reveals that they closely fit the description of device-oriented steps as discussed in the previous chapters. The current chapter investigates whether the error pattern observed can be explained by whether or not each step is relevant to the task goal.

The classification study presented in the previous chapter revealed an additional two device-oriented steps in the Doughnut task: the 'wait for progress bar' step, and the 'clean' (PC) step. While Li et al. (2008) did not include these in their discussion, it can be argued that they are also device-oriented, because they fit the description discussed in chapter 3. Neither step makes a direct contribution towards the main goal, and each is required by the device rather than the task. Indeed, the raters in the classification study (see section 4.3) consistently marked these steps as device-oriented.

The central question addressed in this thesis is whether, and why, device-oriented steps are indeed more problematic than task-oriented ones. As such, the first experiment explores whether this approach is plausible and useful, using the Doughnut task as developed by Li (2006). This task's consistent error pattern makes it an interesting object for investigation. Moreover, since

there are, as yet, no tasks that are specifically aimed at studying the goal relevance of steps[1], the Doughnut task is a useful starting point in exploring the topic.

Experiment 1 uses a basic version of the Doughnut task to explore the errors made on device- and task-oriented steps. It aims to investigate whether the error pattern so frequently observed is robust, and how well the distinction between the two types of steps can explain it. Experiment 2 examines the errors in more detail by looking at how they are affected by working memory load. Chapters 2 and 3 argued that device-oriented errors are due to working memory failures. As such, they should be sensitive to changes in working memory load. Experiment 2 further explores this issue using the Doughnut task. The Doughnut task has been used extensively to study PCEs, but it has never been employed in the study of the relevance of steps to the task goal. As such, the chapter concludes with an evaluation of the Doughnut task as a task for studying device- and task-oriented steps.

5.2 Experiment 1

5.2.1 Introduction

The main purpose of this first study is to explore in detail the error pattern that has been observed consistently on the Doughnut task. While this task has been used extensively in the study of post-completion errors (e.g. Li et al., 2008; Hiltz et al., 2010; Ratwani et al., 2008), it has not been applied to the study of errors due to low goal relevance. Therefore, the current study aims to compare error rates on the different types of steps, and investigates whether making a distinction between them can help in explaining the error pattern found.

An important point that the experiment addresses is the frequent use of interruptions in previous studies using the Doughnut task. These consistently occurred before each of the steps that are identified as device-oriented in this thesis (with the exception of the 'select doughport' and 'wait for progress bar' steps). As such, it is possible that the observed error pattern is, in part, caused by the interruptions, rather than by the steps being device-oriented. It is important to investigate whether the error pattern is robust even without these interruptions.

Two measures are explored in the current study: error rates are compared, and the correlation between performance on the two types of steps is investigated. It is hypothesised that device-oriented steps give rise to higher error rates, and that the distinction between the different types of steps can account for (part of) the error pattern. The correlation between device- and task-oriented steps is included for exploratory purposes, and as such no strong predictions are made for this measure.

5.2.2 Method

5.2.2.1 Participants

Fifteen participants were recruited from a dedicated psychology subject database. They were aged between 19 and 67 with a mean age of 27.1, and nine were female. The majority of

[1]The Spy and Frankenstein tasks were developed *after* this study was completed.

participants were psychology students, and they were paid £6,- for their time.

5.2.2.2 Materials

The Wicket Doughnut task was used (as described in detail in section 4.2.1). At the beginning of a trial, participants had to take a call on the Call Centre to get the next order, this was then 'transferred' to the Doughnut task interface. Clicking the 'Show Order' button displayed the order on the order sheet. The order data then had to be entered into the 5 widgets. The subtask on each widget consisted of selecting the widget, entering data into it and confirming the input. After all widgets were completed, the order had to be processed by clicking the 'Process' button. A pop-up screen then indicated the completion of the trial, and the number of doughnuts made. At the end of the trial, the machine had to be cleaned by clicking the 'Clean' button. While Li et al. (2008) used interruptions at several points during the task, the current experiment did not.

Two separate computer terminals were used; one for the call centre and one for the doughnut making task. Both screens were operating at a resolution of 1280 x 1024 pixels.

5.2.2.3 Design

A within-participants design was used, with the independent variable being the type of step. This variable had two levels: device- or task-oriented. Seven device-oriented steps were present (as identified in the classification study, see section 4.3, table 4.1): five selector steps, the clean step, and the progress bar step. Twelve task-oriented steps were present: the 'new order' step, the process step, five data entry steps and five widget-confirmation steps. The call centre was not included in the analysis, as it served only to separate trials on the main Doughnut task. Moreover, the 'dismiss pop-up' step was excluded because no errors were possible on this step, as the pop-up screen blocked actions on the main screen.

The dependent variable was the error rate. Errors were counted systematically according to the required steps. An error is defined as any action that deviates from the required action at a certain step. To ensure only inappropriate actions are counted and not each individual inappropriate mouse-click, only one error could be made on each step. Note that step times were not measured on this task, as some steps required additional operations (looking up doughnut information or performing calculations) while others did not, thus skewing the time taken to complete these particular steps. This is further addressed in the general discussion of this chapter.

5.2.2.4 Procedure

Participants carried out the experiment individually. During the training phase, participants were given an instruction sheet that explained in detail what their task was, and all the procedures necessary to complete it (see Appendix A.2 for the full instructions). After reading the instruction sheets, they observed the experimenter doing the task once, after which they were allowed to practice it twice. Any errors made during the training trials were pointed out im-

mediately using the default Windows XP notification sound and were required to be corrected before the participant was allowed to move on.

Participants were instructed to complete the doughnut task as quickly and as accurately as possible. A timer was displayed on the screen throughout the experiment to encourage swift performance; it was reset after each trial. After processing the doughnuts, a pop-up screen notified the participant of the number of doughnuts made. This was sometimes deliberately inaccurate to encourage more accurate performance. Participants were not aware that errors were being studied.

When an error was made, this was pointed out to the participant immediately using the Windows XP notification sound. It was not possible to resume the task until the error was corrected (with the exception of the PCE, which could not be corrected). This was done by simply executing the correct task step. Any deviation from the prescribed task procedure was counted as an error. However, at any given step, only the first erroneous action was counted as an error; any subsequent erroneous actions on the same step were ignored. The reason for this was that participants sometimes rapidly clicked the erroneous button a number of times in a row. Had these actions been counted as errors, it would have resulted in an inflated error rate. While only one error could be made on any given step, any errors on subsequent steps were counted as a new error. Therefore, it was possible to make more than one error per trial.

During the experimental phase, the participants then completed 11 trials, with the opportunity of a short break after 6 trials. The total duration of the experiment was approximately 50 minutes.

5.2.3 Results

The main dependent variable was error rate, which was expected to be higher on device-oriented steps. Correlations between performance on device- and task-oriented steps were also investigated.

Data from 15 participants was recorded. For each, error rates were calculated for the two step types. Only one error was possible on each of the steps, so a total of 19 errors could be made on a single trial. Each participant did 11 trials, and data from 15 participants was analysed, giving a total opportunity for errors of 3135. Across all participants, a total of 169 errors were made, giving an overall error rate of 5.39%. Device-oriented errors were made more often than would be expected if errors occurred randomly throughout the task. If errors occurred completely randomly, then the device-oriented steps should be responsible for $7/19 = 37\%$ of all errors. However, it was found that these steps accounted for 88% of all errors, far exceeding the stochastic expectations.

The average error rate on device-oriented steps was 12.8% (SD = 7.5%), while on task-oriented steps it was 1.1% (SD = 1.3%). These rates were compared using a Wilcoxon signed-rank test, because the data were not normally distributed and the variances were not equal. This revealed a significant difference between the two types of steps, $Z = -3.35, p < 0.005, r = 0.61$.

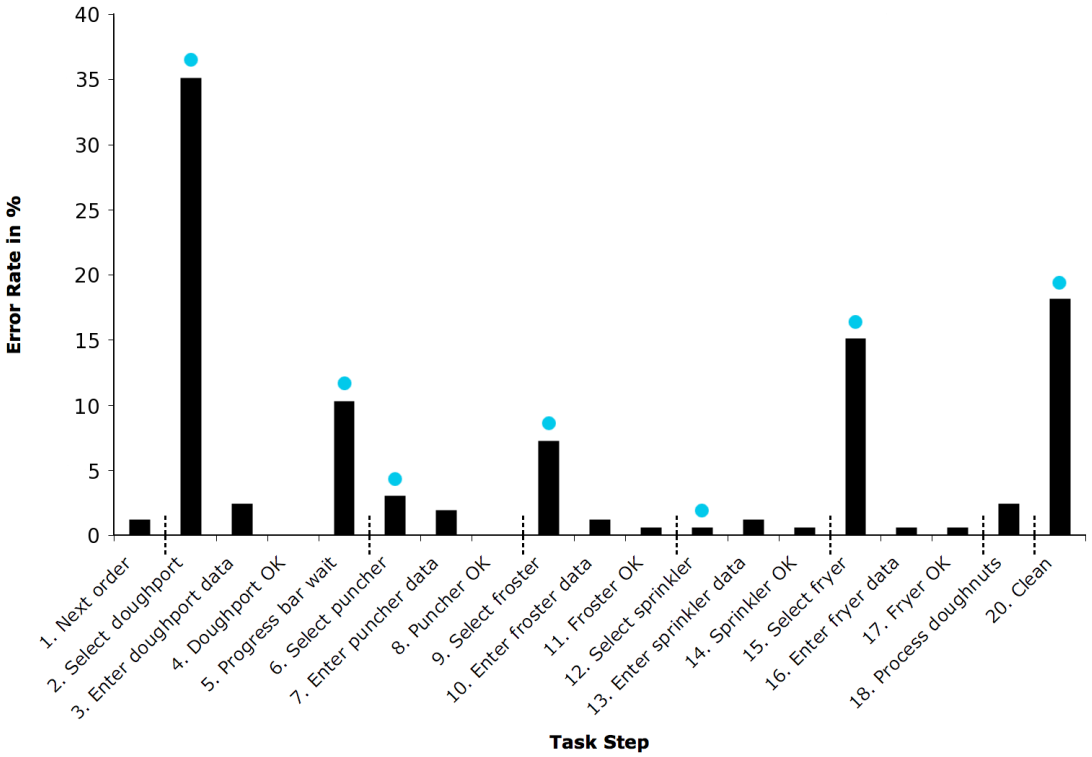


Figure 5.2: *Error pattern observed on the Doughnut task. Device-oriented steps are marked with a circle above the bars. Dotted lines on the X-axis indicate subtask boundaries.*

Figure 5.2 shows the pattern of errors on each of the individual steps, averaged over all participants. Since the number of device- and task-oriented steps in the task was not equal, it was not possible to analyse in detail how much each of the steps contributed to the variation in the error rates. Nevertheless, the graph clearly demonstrates that the six steps with the highest error rates are all device-oriented. Only one out of the seven device-oriented steps has an error rate that is much lower (selecting the Sprinkler). Moreover, five out of the seven device-oriented steps have error rates over the 5% systematicity level (see Byrne and Bovair, 1997), while none of the task-oriented steps reach this level.

Figure 5.3 shows the number of trials for which 0, 1, 2, 3, or 4 errors were made. Four was the largest number of errors on any given trial. For the trials with more than one error, the average number of steps in between two errors was 7.6 (SD = 4.1), and on only two occasions did errors occur on two consecutive steps (0 steps between errors).

The correlation between error rates on device- and task-oriented steps was also investigated using Spearman's rank test. This revealed no significant correlation, $\rho = 0.35, p = 0.21$.

5.2.4 Discussion

The current experiment investigated the error patterns on the Doughnut task in more detail. It was hypothesised that error rates were higher on device-oriented than on task-oriented steps, even when no interruptions were present. Moreover, the current study looked at whether error rates on device- and task-oriented steps were correlated within participants.

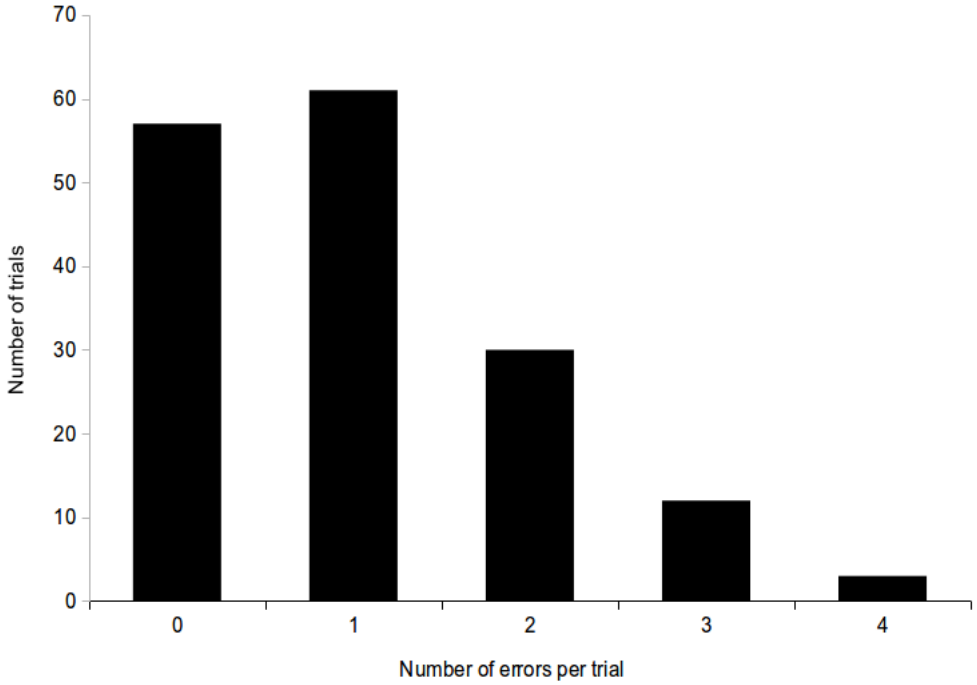


Figure 5.3: *Number of trials on the Doughnut task for which 0, 1, 2, 3, or 4 errors were made.*

5.2.4.1 Overall Error Pattern

The typical error pattern found in earlier studies using the Doughnut task was also present in the current results. The selector steps showed relatively high error rates, with the strongest effect on the first one (the dough port selector). This is consistent with findings from earlier studies, as discussed in section 5.1. The 'clean' step also exhibited a relatively high error rate, again in line with earlier findings on the Doughnut task, whereas most of the others were well below the 5% systematicity level. As such, the error pattern is extremely robust across experiments.

This error pattern held despite the fact that the current study did not make use of interruptions. Previous work by Li et al. (2008) has shown that interruptions increase error rates. The interruptions in this original study were all placed just before the clean step and the selector steps, and therefore it was possible that the error patterns were caused mostly by their deleterious effects. However, the finding of the current study that the pattern holds even without interruptions demonstrates that other factors are at work as well, such as the error-prone steps being less relevant to the task goal.

It was found that error rates on device-oriented steps were indeed much higher than those on task-oriented steps, confirming the predictions. Moreover, the distinction between device- and task-oriented steps was able to explain a sizable part of the error pattern observed. This means that an approach from the relevance of steps to the task goal is highly useful in explaining the error pattern on the Doughnut task, and this is a promising avenue for future research.

The average error rate of 5.39% indicates that on average, approximately one error is made per trial of 20 steps. A concern is that these errors may not be independent, but instead occur

in clusters. To investigate this, the distribution of errors across trials was investigated. It was found that on the majority of trials, participants made zero or one error, while in approximately a quarter of trials two or more errors were made. To investigate whether the errors were independent of one another or occurred in clusters, the number of steps between two errors in trials with more than one error was investigated. It was found that there was a relatively large number of steps between most errors, with very few clusters of consecutive errors occurring. This indicates that the errors are likely to be largely independent of one another.

Interestingly, no correlation was found between error rates on device- and on task-oriented steps within participants. This means that performance on one type of step was not related to performance on the other. This is an interesting finding, as it indicates that the errors could arise from different underlying causes. If the same cognitive processes were responsible for both types of errors, it would be expected that higher error rates on one would be associated with higher error rates on the other, but this does not appear to be the case. Moreover, if found to be robust, this finding could be relevant to an applied domain, because it could indicate that a person who is very experienced in a particular task may still make a disproportionate number of device-oriented errors. This could be problematic in safety-critical domains in which complex devices are used. However, this finding needs to be investigated further before any such conclusions can be drawn. This will be addressed further in the studies using the Spy task (chapter 6) and the Frankenstein task (chapter 7).

5.2.4.2 Individual Steps

Figure 5.2 clearly shows that the error rate on the first selector step (doughport) is much higher than on the remaining four selector steps. This is surprising, since these steps are very similar in form and function, and therefore similar error rates would be expected. The steps differ only in their position in the task sequence, but it is not known why being the first would lead to such a large increase in error rate (e.g. Li, 2006; Hiltz et al., 2010). Conversely, error rates on the fourth selector step (sprinkler) were much lower than the others. Again, this is surprising. One explanation is that, within each subtask, the selector step has to compete with the highly task-oriented data entry step. Occasionally, a selector step may lose this competition, leading to an error. It is possible that the first selector step not only competes with the subsequent step, but also with the highly task-relevant overall goal of entering all the data into the widgets. However, this explanation does not account for the lower error rate found on the fourth selector step. While this issue may be specific to the Doughnut task, it presents an interesting point. As such, it will further be explored in experiments 5 and 6 (see chapter 7), which test whether this effect is also present in a similarly structured task.

The purpose of the current study was to explore the Doughnut task *as is*, without any modifications or manipulations. This means that the different steps were not controlled for. There is one step, the 'clean' step, on which this may be problematic. This step was classified as device-oriented in the previous chapter. However, in addition to this, it is also a

post-completion step (Li et al., 2008). It is well-known that PC steps are relatively error-prone (e.g. Byrne and Bovair, 1997; Chung and Byrne, 2008), and therefore the high error rate on the clean step could, in part, also be due to its post-completion structure. It is important to note the relationship between goal relevance and post-completion task structure. Some device-oriented steps are post-completion steps, and most post-completion steps are also device-oriented[2]. Indeed, the clean-up step on the Doughnut task is device-oriented, *as well as* being a post-completion step. Therefore, it is likely that both factors contributed to the relatively high error rate.

The main aim of this study was to investigate the error patterns on the Doughnut task in more detail. The current experiment showed that the error patterns previously observed on the Doughnut task are robust, and that they hold even without the inclusion of interruptions. Moreover, dividing the steps according to their relevance to the task goal is a successful approach in explaining the error pattern. The findings of this study are important for the current thesis, as they confirm the premise on which the main argument is based: that a distinction between device- and task-oriented steps can be a useful approach in explaining robust error patterns. However, as intended, these results cannot easily be generalised; this will be discussed further in the overall chapter discussion (see section 5.4).

The next experiment will investigate in further detail the underlying causes of device- and task-oriented steps. One of the hypotheses put forward by the current thesis is that they are differentially affected by working memory load. As such, the next study will manipulate the working memory load on the Doughnut task, to investigate the effect on device- and task-oriented steps.

5.3 Experiment 2

5.3.1 Introduction

Byrne and Bovair (1997), amongst others, argued that slip errors in routine procedures are generally due to failures in working memory. After all, they are not due to a lack of knowledge, and as such are unlikely to be due to long-term memory forgetting. Indeed, Byrne and Bovair (1997) showed that one such error, the PCE, is sensitive to changes in working memory load. Their argument is based on the loss of the main goal as an activation source for the PC step. While this specific mechanism does not directly apply to device-oriented steps in general, their reasoning that higher working memory load leads to increased competition between memory items also extends to other steps. Indeed, if device-oriented steps have lower activation levels as hypothesised, then the increased competition that results from an increase in working memory load should disproportionately affect these steps.

[2]Though it can be argued that not all PC steps are device-oriented. For instance, the action of removing the original after making copies on a copy machine is a typical post-completion step. However, it can be argued that it is not device-oriented, as the step is not just required by the device, but without it the task cannot reasonably be said to be complete. This is an issue that requires further investigation in future work, but is outside the scope of this work.

The reason why working memory load is studied is not only theoretical, however. Many safety-critical tasks take place in high load environments, such as busy hospital wards or while flying a fighter jet. Therefore, it is important to understand how working memory load affects errors, and particularly if this is different for device- and task-oriented steps. If device-oriented steps are indeed more severely affected by a high load, then these steps should be designed out of the relevant devices as much as possible.

The current experiment investigates whether device-oriented steps are differentially affected by working memory load. The Doughnut task is used for this, since the previous experiment showed that it produces a strong effect of goal relevance. A preliminary analysis of this study was presented as a talk at CogSci 2010 (Ament et al., 2010).

5.3.2 Method

5.3.2.1 Participants

Forty participants were recruited from a dedicated psychology subject database. They were aged between 18 and 33 with a mean age of 22.0, and 27 were female. The majority of participants were students, and they were paid £6 for their time.

5.3.2.2 Materials

Like experiment 1, the current experiment used the Doughnut task, again without interruptions. For the details of this task, please refer to sections 4.2.1 and 5.2.2.2.

In addition to the main Doughnut task, a monitoring task called the Doughnut Live Feed was used to vary working memory load (see figure 5.4). In this task, participants had to count the number of doughnuts sold in the shops. The Doughnut Live Feed was shown at the bottom of the screen below the main Doughnut task interface, where occasionally a description of a doughnut was shown. Participants had to attend to a specific characteristic of the doughnut (such as dough type, hole shape or frosting) and keep count of how many with that characteristic were sold. In the low working memory load condition, participants were asked to attend to and keep track of doughnuts with a specific dough type, for instance Crispy. In the high working memory load condition, participants were asked to attend to and separately keep track of doughnuts with a specific dough type and those with a specific hole shape. In both conditions, once a participant had counted 20 doughnuts of the specified type, they had to click the button on the left of the live feed and start counting from zero again. This allowed the experimenter to assess whether a participant was successfully monitoring the live feed.

To ensure effective monitoring, new items on the live feed did not capture visual attention. This was achieved by using a background that changed from grey to white and back in continuous cycles. Each doughnut description faded in on top of that from white to black, and faded out again after a random number of cycles. Figure 5.4 shows the progression through one cycle. Each cycle took three seconds, and items remained visible for between 2 and 4 cycles. This randomness made it impossible for participants to predict when a new doughnut description

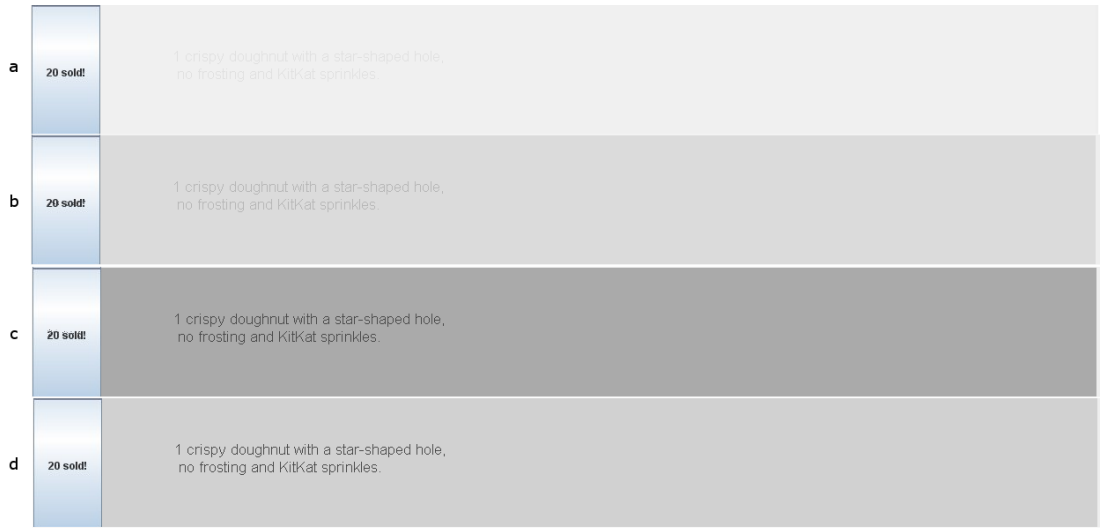


Figure 5.4: *the Doughnut Live Feed. (a) A cycle starts out completely white. (b) The background then quickly fades to grey, while the item fades from white to black. (c) Halfway through the cycle, the background and the item are at its darkest, and the item is clearly visible. (d) At the end of the cycle, the background fades to white again while the item may either stay visible or fade as well.*

would be shown, which would have made the monitoring less effective. The monitoring task and primary tasks were carried out simultaneously.

As in the previous experiment, two separate computer terminals were used; one for the call centre and one for the doughnut making task and live feed. Both screens were operating at a resolution of 1280 x 1024 pixels.

5.3.2.3 Design

A mixed design was used, with two independent variables. The first independent variable was working memory load; this was varied between participants. This variable had two levels: low load and high load. The second independent variable was the type of step; this was varied within participant. This variable also had two levels: device-oriented and task-oriented.

The dependent variable was the error rate. Errors were counted systematically according to the required steps. An error was defined as any action that deviates from the required action at a certain step. To ensure only inappropriate actions are counted and not each individual inappropriate click, only one error could be made on each step.

5.3.2.4 Procedure

Participants carried out the experiment individually. During the training phase, participants were given an instruction sheet that explained in detail what their task was, and all the procedures necessary to complete the task (see appendix A.2). They then observed the experimenter doing the task once, after which they were allowed to practice it twice. Any errors made during the training trials were pointed out immediately using the default Windows XP notification sound and were required to be corrected before the participant was allowed to move on. After each practice trial, the experimenter asked the participant how many doughnuts they had counted on the live feed, and encouraged more accurate performance if necessary.

Participants were instructed to complete the doughnut task as quickly and as accurately as possible. A timer was displayed on the screen throughout the experiment to encourage swift performance; it was reset after each trial. After processing the doughnuts, a pop-up screen notified the participant of the number of doughnuts made. Participants were also told to count the doughnuts in the live feed as accurately as possible; this was further encouraged by the '20 doughnuts' button. Participants were not aware that errors were being studied.

During the experimental phase, the participants completed 11 trials, with the opportunity of a short break after 6 trials. Any errors were pointed out immediately and had to be corrected before the participant was allowed to carry on. The total duration of the experiment was approximately 60 minutes.

5.3.3 Results

Data from 12 participants was excluded from the analysis. The reasons for excluding participants varied. Three participants failed to follow the instructions to monitor the live feed correctly, which meant their working memory load could not be guaranteed. One participant's data sheet was lost. Eight participants were excluded because they made omission errors at a particular step on more than 65% of trials. The reason for excluding these error-prone participants is that such high error rates likely indicate that the participant has not correctly learnt how to perform the task correctly (Chung and Byrne, 2008). Analysis of error rates for the remaining twenty-eight participants is presented.

Due to the failure of a relatively large number of participants to perform the task to criterion, it was first examined whether error rates decreased as participants gained more experience at performing the task. Figure 5.5 shows the error rates on each consecutive trial. A Page's Trend test was done to determine whether performance improved after more trials were completed. This revealed a significant reduction in error rates as a function of trial number, $L = 11444, p < 0.05$. This suggests that a learning effect was present in the data. Moreover, a learning curve was fitted to the graph (see figure 5.5). The best fit came from a linear regression line, although at $R^2 = 0.16$ the fit was poor.

Of primary interest were the error rates on device- and task-oriented steps. Error rates were calculated for each participant for the relevant step types. Only one error was possible on each of the steps on each trial. As in experiment 1, step 19 (dismissing the pop-up screen) was removed from further discussion, because no error was possible on this step. Thus, a total of 19 errors could be made on a single trial. Each participant did 11 trials, and data from 28 participants was analysed, giving a total opportunity for errors of 19 x 11 x 28 = 5852. Across all participants, a total of 292 errors were made, giving an overall error rate of 4.99%.

It was hypothesised that error rates were higher on device- than on task-oriented steps. Table 5.1 shows the average error rates across all participants on the different types of steps. Working memory load was also manipulated, and it was hypothesised that a high load had a greater effect on device- than on task-oriented steps. Figure 5.6 shows the error rates on the

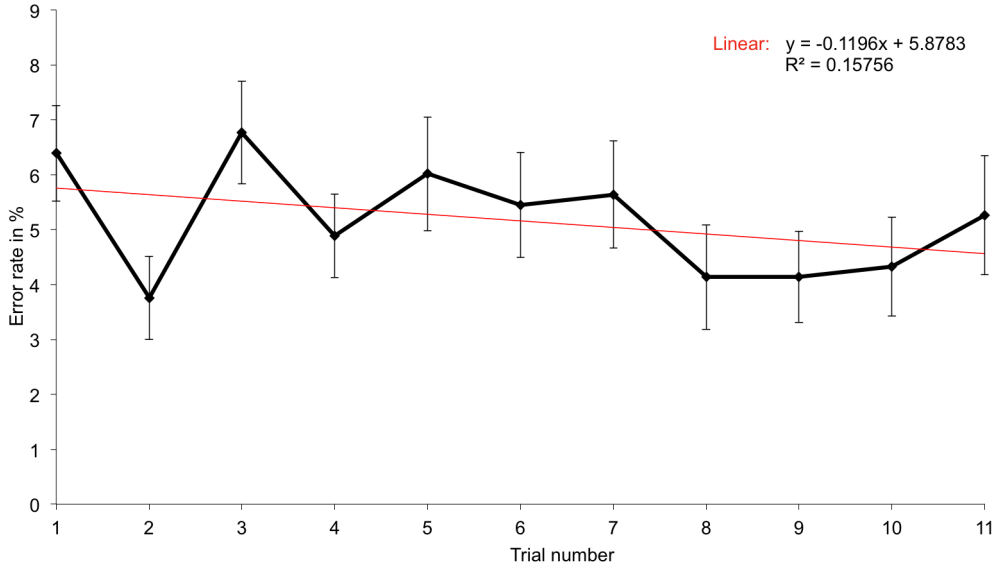


Figure 5.5: *Error rates for each trial on the Doughnut task.*

different working memory load levels, for both device- and task-oriented steps. Error rates on task-oriented steps remained stable across all conditions, while error rates on device-oriented steps increased under high working memory load. A 2 x 2 mixed-design ANOVA with type of step as the within-subjects variable and working memory load as the between-subjects variable was used (see section 5.3.4 for a discussion of the use of parametric tests on error data). This revealed significant main effects of working memory load, $F(1, 26) = 8.10, p < 0.01$, and of step type, $F(1, 26) = 81.90, p < 0.001$. An interaction effect was also found, $F(1, 26) = 6.68, p < 0.05$. Furthermore, simple effects analysis showed that there was no simple effect of working memory load on task-oriented steps, $F(1, 26) = 0.95, NS$. On the other hand, there was a significant simple effect of working memory load on device-oriented steps, $F(1, 26) = 7.53, p < 0.05$. Because the data showed a moderate positive skew, the data were transformed using a logarithmic transformation to determine whether this improved the distribution and outcome of the analysis. While this did reduce the skew of the data, it did not affect the outcome of the analysis.

Type of Step	**Error Count (Opportunity)**	**Mean Error Rate (SD), in %**
Total	292 (5852)	4.99 (2.51)
Task-oriented	57 (4004)	1.42 (0.96)
Device-oriented	235 (1848)	12.7 (7.44)

Table 5.1: *Total error counts and mean error rates across all participants and conditions for the different types of steps.*

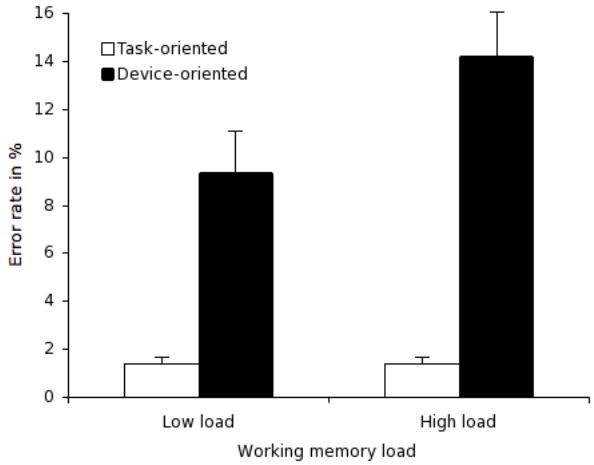


Figure 5.6: *Error rates across working memory load and type of step conditions. Error bars represent the standard error of the mean.*

5.3.4 Discussion

The current experiment investigated the hypothesis that error rates on steps that are not directly relevant to the task goal are higher than those on steps that are, and that working memory load has a differential effect on them. The results of this study show that the error rate observed at device-oriented steps is greater than the error rate observed at task-oriented steps. This is in line with the prediction and with the results of experiment 1.

A high working memory load resulted in higher error rates overall, and an interaction effect of working memory load and type of step was also found. This means that, as predicted, working memory load has a differential effect on device- and task-oriented steps. More specifically, the significant simple effect of working memory load on device-oriented steps, and the absence of such an effect on task-oriented ones indicates that a high working memory load affects device-oriented steps but not task-oriented ones. This confirms the predictions, and is in line with the hypothesis that steps that have low relevance to the task goal have lower activation levels than those with high relevance. The high working memory load leads to increased competition between items in working memory, which makes it more difficult for them to reach the required activation level to direct behaviour. Lower activation levels on device-oriented steps make it even more difficult for these steps to compete with other memory items, and thus lead to more errors.

An issue that arose in this study is the use of parametric statistical tests on error data. These are almost always non-normally distributed, making the analysis of these data problematic in case no widely accepted non-parametric alternative is available. This is the case in the current study, where a mixed design is employed with a within-participants variable and a between-participants variable, but also in experiments 4, 5 and 6 in chapters 6 and 7. There are several options to address this issue. The first is to run two non-parametric tests on the data, one for the within-participants variable and one for the between-participants variable. However,

this does not take into account any interaction between the two variables, and as such is not a suitable option. The second is to use parametric tests, but with caution. This method is employed by the majority of studies of human error, such as Li (2006),Chung and Byrne (2008), Byrne and Bovair (1997), Ratwani et al. (2008), Back et al. (2010), Byrne and Davis (2006), Byrne (2008) and Li et al. (2008). Another option that can be applied is the transformation of data using a logarithmic or square root function, for instance. This could especially be effective if the non-normality is caused by skewness, as is often the case in error data. However, for the current experiment, such a transformation did not affect the outcome of the analysis. The approach taken in the current thesis is to use non-parametric tests as much as possible whenever the data violate the assumption of normality (or other assumptions) and cannot be corrected by transformation. In case no non-parametric tests are available, parametric tests are used instead.

A shortcoming of the current study is the limited amount of training participants received. This may have contributed to the high error rates observed on the eight excluded participants, and the finding of a learning effect. This is a problem that affects many studies of errors in skilled procedures, such as Back et al. (2007) and Ratwani et al. (2008). Time and financial resources are limited, and therefore it is not always possible to train participants for longer periods of time. Moreover, it is difficult to determine when a participant has become sufficiently skilled in the task. This remains a thorny issue in the study of routine procedural tasks, but has received little attention from other studies using the Doughnut task or similar tasks (e.g. Li, 2006; Hiltz et al., 2010; Chung and Byrne, 2008). Nevertheless, the remaining experiments in this thesis will address this issue by increasing the number of training trials each participant executes before moving on to the experimental phase.

The purpose of the current experiment was to study the effect of working memory load on device- and task-oriented steps. While the experiment generated positive results, it is important to note that these cannot yet be generalised to other tasks. As such, the conclusion of the current experiment is that *on the Doughnut task*, working memory load has a stronger effect on device- than on task-oriented steps.

5.4 General Discussion

The main purpose of the current chapter was to explore whether a distinction between device- and task-oriented steps can be useful in explaining error patterns and whether these two types of steps can be studied in the laboratory in a reliable way. The Doughnut task was used for this, because it has repeatedly generated a robust error pattern, and the task steps on this task have been reliably classified as device- and task-oriented (see section 4.2.1).

The findings of the two experiments presented in this chapter are in line with the predictions that steps with low relevance to the task goal are more prone to errors, and are affected more by a high working memory load than those that have high relevance. The main hypothesis put forward by the current thesis is that this is due to lower activation levels on device-oriented steps. The findings of these two experiments are in line with this: lower activation levels make

it more likely that the step (or its associated schema or goal) occasionally does not reach the required activation level to be carried out. Moreover, a higher load makes it even more difficult for steps with a low activation levels to compete with others. However, the findings from the current study cannot rule out the alternative mechanisms, such as higher activation levels on erroneous steps, and degraded representations (see chapters 2 and 3). Specifically, the task structure of the Doughnut task is such that the majority of device-oriented steps are followed by a strong task-oriented step. For instance, the selector steps are followed by the data entry steps, and the 'clean' step is followed by the next trial. As such, it seems equally likely that the errors on the device-oriented steps are caused by the highly task-relevant steps having activation levels that are comparatively high, and make it more difficult for the previous steps to compete. The task structure of the Doughnut task makes it difficult to distinguish between these two scenarios.

Moreover, the hypothesis that representations of device-oriented steps are more degraded or less strong can also be used to explain the findings. The more degraded the representations are, the higher the error rate will be. Moreover, this is further exacerbated by a high working memory load, which further degrades the patterns, leading to even more errors.

In short, the findings of the current studies can be explained using all three error mechanisms addressed in this work. Therefore, further studies using different experimental tasks must further investigate these different explanations, to determine which can best account for the difficulties associated with low relevance to the task goal.

5.4.1 Evaluation of the Doughnut Task

While the Doughnut task has yielded very strong and positive results in the current two experiments, the results should be interpreted with some caution. They cannot yet be generalised to other tasks, because the Doughnut task has a number of drawbacks.

First, the Doughnut task does not adequately control for factors such as step difficulty and visual salience. It can be argued that certain steps (such as entering the correct data) are more difficult than others (such as showing a new order). Interestingly, using this reasoning, the most difficult steps were in fact task-oriented: Kelley (2007), also using the Doughnut task, noted that some participants found the data entry steps extremely challenging, especially those that included mathematical operations. In the current study, these steps were identified as task-oriented. As such, if step difficulty played a large role in determining the error rates, it would be expected that the error rates were higher on task-oriented steps than on device-oriented ones, which is clearly not supported by the current findings. In addition, the task interface did not adequately control for the different types of steps. While it can be argued that the task interface is fairly uniform and there are no large differences between the interface items for the different steps in terms of visual salience, it is possible that this had an unexpected effect.

Second, step times are often used in conjunction with error rates (e.g. Byrne and Davis, 2006; Ruh et al., 2010), and form part of the main hypothesis of the current work. However, the

Doughnut task cannot produce reliable step time data. The reason for this is that some steps on the task require participants to perform mental operations or find data on the order sheet. This obviously takes additional time, and makes it difficult to compare these steps to others on which such operations are not required.

Third, the Doughnut task uses a false completion signal (the pop-up screen notifying participants that the Doughnuts are ready) and a competing signal (the next telephone call coming in) for the post-completion step. Li (2006) found that these signals play a role in increasing the error rate on the PC step. While the current findings also hold when the PC step is excluded, part of the effects found may be due to these signals and the post-completion structure of the task, rather than purely a distinction between device- and task-oriented steps. As such, future experiments should use tasks in which no strong PC step is present, and no competing and false completion signals are provided.

Fourth, errors on all steps except the PC step had to be corrected before a participant could carry on with the task. While the participants were notified when they made a post-completion error, the lack of remedial action may have provided less reinforcement for this step than on other steps. This issue only affected the post-completion step, but it may have further contributed to the relatively high error rate on this step.

Fifth, the fifth step in the task procedure, which required participants to wait until the progress bar was filled, was qualitatively different from the other steps. On step 5, no action was required; instead participants merely had to wait for the progress bar to fill, until they could continue with the next step. While it can be argued that, like any other step, participants have to *remember* to wait before continuing, it poses an experimental problem. It is possible that, when making an error on this step, participants were simply impatient and clicked the next step even though they knew the machine would not let them. As such, there is not an easy way to distinguish between situations in which participants made a true error on this step or were simply impatient. Moreover, if participants did wait on step 5, it allowed them an extra few seconds to think about the next step (selecting the puncher). This means that the next step may be executed faster, and more accurately. While a substantial error rate was still found on the step in question, it cannot be ruled out that this would have been even higher without the progress bar step. Indeed, alternative versions of the Doughnut task have eliminated the 'Wait for progress bar' step (e.g. Ratwani et al., 2008; Back et al., 2010).

Sixth, as noted above, the Doughnut task cannot easily distinguish between the different mechanisms for errors as offered by the theoretical models. This is due to the specific structure of the Doughnut task, in which most of the device-oriented steps are followed by a highly salient step. Different experimental tasks must be used to determine which explanation fits best.

5.5 Conclusion

The current chapter presented two experiments that showed that the notion of goal relevance is useful in explaining the error patterns observed on the Doughnut task. They demonstrated

that people are more likely to make errors on device-oriented steps than on task-oriented ones. This provides support for an approach based on these concepts and that they can be effective in explaining observed error patterns. Moreover, it has provided some support for the hypothesis that device-oriented steps have lower activation levels, and therefore will be affected more severely by a high working memory load.

However, it is clear that the Doughnut task has significant problems as a task for studying the effect of the relevance of steps to the task goal. It does not adequately control for device- and task-oriented steps, and there are a number of alternative explanations for the current results that cannot be ruled out using this task. As such, the Doughnut task should be considered as a valuable and useful starting point in the study of goal relevance, but is unsuitable for directly studying this. Since the claim that steps with low relevance to the goal in general are more likely to be problematic than ones with high relevance is central to this thesis, the results of the current chapter must be replicated on a different task to ensure the findings are robust and can be generalised. The next two chapters present experiments that address this.

Take-home message:

Two experiments showed that error rates are higher on device-oriented steps than on task-oriented ones, and working memory load affects device-oriented steps more severely. Therefore, the notion of goal relevance is successful in explaining the robust error pattern observed on the Doughnut task. However, the Doughnut task is not suitable for the controlled investigation of the relevance of different steps to the task goal.

Chapter 6

Further Investigation of the Cognitive Mechanisms Underlying Device- and Task-Oriented Steps using the Spy Task: Experiments 3 and 4

In this chapter:

- A new experimental task is introduced, which addresses the shortcomings of the Doughnut task.
- Two experiments are presented that use this Spy task.
- The first experiment compares device- and task-oriented steps, and finds higher error rates, longer step times and different eye movements on the former.
- The second experiment addresses the eye movement patterns in more detail by looking at how visual cues are used.
- The chapter concludes that the most likely explanation for the findings is that device-oriented steps have lower activation levels.

6.1 Overview

The previous chapter showed that an approach from the relevance of steps to the task goal can be successful in explaining the error patterns observed on the Doughnut task. However, the Doughnut task has several problems as a task to study device- and task-oriented steps. It was unable to distinguish between the different theoretical explanations, and a number of confounding factors were present. This means that the results from the previous chapter cannot easily be generalised, and further experiments are necessary to demonstrate that the findings are robust.

The current chapter aims to address this by introducing a new task: the Spy task. A full

description of this task can be found in section 4.2.2. It makes a number of improvements over the Doughnut task:

- The last step on the Doughnut task has been identified as both a post-completion step as well as a device-oriented step. Post-completion steps are known to generate high error rates (e.g. Byrne and Bovair, 1997; Li et al., 2008), and therefore the high error rates observed on the 'Clean' step could, in part, also be due to its position after the 'completion' of the task. However, this adds a layer of complexity to the step, as it is not possible to determine how much of the error rate on this step is due to it being device-oriented and how much is caused by its post-completion structure. To remedy this, the Spy task does not include a post-completion step.

- On the Doughnut task, the post-completion step was accompanied by a competing signal and a false-completion signal, which have been shown to increase error rates on the PC step (Li, 2006). The Spy task does not include a strong post-completion step and therefore also no false-completion signals. However, competing signals are not necessarily specific to the post-completion step, and it is not known to what extent they can affect performance on other steps as well. To minimise any potential effects, the Spy task did not include any of these signals.

- On the Doughnut task, all errors had to be corrected, except for the PCE. The mandatory correction of a step could act as a reminder, and as such participants may be more likely to correctly execute it on subsequent trials. It is possible that this further contributed to the high error rate on the post-completion step. Therefore, on the Spy task, all errors have to be corrected, with no exceptions, to ensure that each step is treated equally.

- The Doughnut task contains a number of repetitive subroutines, which means that there are essentially only a small number of different steps that are repeated several times. This increases the chance that any effects found are due to factors to do with a particular step rather than with it being device- or task-oriented. The Spy task contains a larger number of different steps, only a few of which are repeated more than once, thus greatly reducing this risk. This is a significant improvement over the Doughnut task, as it also reduces the effect of confounding factors in the task interface or step difficulty.

- In addition to error rates, step time data is often used to further support findings in the study of routine procedural performance (e.g. Chung and Byrne, 2008). However, the Doughnut task is not suitable for the analysis of step time data, because it required looking up information or additional mental operations before a number of steps. This means that step times on these actions are likely to be substantially longer than on actions that do not require such operations, making it impossible to directly compare the step times. The Spy task did not include these additional operations and as such is more suitable for the investigation of step times.

- The majority of device-oriented steps on the Doughnut task were followed by a strong task-oriented step. This made it impossible to distinguish between two alternative error mechanisms. It is possible that the device-oriented steps had lower activation levels, but based on the Doughnut task results it seems equally likely that the task-oriented steps were too active, and therefore captured attention away. The task structure on the Spy task was designed not to include such step pairs.

The first experiment in this chapter (referred to as experiment 3) investigates performance on the Spy task in a basic condition with no manipulations (besides the type of step). It aims to establish an idea of base-line performance on the task, and compares the performance of a small number of participants. Error rates and step times are both addressed. Moreover, eye movements are also recorded for exploratory purposes. These reveal an interesting difference between device- and task-oriented steps. This finding is further explored in the second experiment in this chapter (referred to as experiment 4), which looks at how visual cues are used differently for device- and task-oriented steps. A preliminary analysis of the two experiments discussed in this chapter was presented as an oral presentation at CogSci 2009 (see Ament et al., 2009).

6.2 Experiment 3

6.2.1 Introduction

The third experiment of this thesis aims to investigate the difference between device- and task-oriented steps in a more controlled setting, using the Spy task. It aims to determine whether the findings of the previous chapter are robust and apply to other sequential tasks as well. Moreover, additional measures are explored (step times and eye movements) to further investigate the differences between the two types of steps. The main hypothesis of this thesis is that device-oriented steps have lower activation levels, and this predicts that step times will be longer on these actions. It is not known if this has any effect on eye movements, since none of the models and previous work discussed in chapters 2 and 3 make any strong predictions about this. As such, this measure is mainly exploratory, and investigates the basic measures of average fixation length, number of fixations before each step, and the total fixation time.

6.2.2 Method

6.2.2.1 Participants

Twelve undergraduate students took part in the experiment, all were paid £6 for their time. Six were female; age information was not recorded.

6.2.2.2 Stimuli

The Spy task was used in this study; it is described in detail in section 4.2.2. Participants' main task was to fly a plane to a destination and deliver a secret message. A number of device- and task-oriented steps were identified in the task; these can be found in table 4.2.

A Tobii 1750 eye-tracker was used to collect eye movement data, with a sampling frequency

of 50Hz. Its integrated screen was used to present the Spy task, which operated at a resolution of 1280x1024.

6.2.2.3 Design

A within-participants design was used, with all participants completing the same task. The independent variable was the type of step, which had two levels: device-oriented and task-oriented.

There were three (types of) dependent variables. The main measure was error rate; errors were counted systematically according to the individual steps. Each step could yield only one error per trial, to avoid counting multiple erroneous mouse clicks at the same step. Step times were also recorded, these were defined as the period of time from the completion of the previous step until the first mouseclick on the target step. In addition, eye movements were recorded for exploratory purposes. These included average fixation length, number of fixations and the total fixation time.

6.2.2.4 Procedure

The experiment started with an on-screen information sheet that explained the cover story and the steps necessary to complete the task (see appendix A for the full instructions). Participants were then walked through the task step by step: an instruction was shown superimposed on the task interface, until the corresponding step was completed accordingly. Participants then practised without instructions. Any errors made during the training were pointed out immediately using the default Windows XP error sound, and proceeding was not possible until the error was corrected. Participants practised until they completed two trials without making any errors.

The main task then started, on which 11 trials had to be completed, with the option of a short break after 6 trials. Participants were encouraged to work as fast and as accurately as possible. The total duration of the experiment was approximately 50 minutes.

6.2.3 Results

Data from 12 participants was recorded. Step 23 (activating the secret compartment) was excluded from analysis, because it had no visual representation in the task interface. Many previous studies have demonstrated that the task interface plays an important role in the strategies used to execute the task (e.g. Gray et al., 2006). Therefore, it is possible that the absence of a visual representation led to a different strategy on the step in question, which was thought may affect performance on this step.

An important issue that arose from the experiments using the Doughnut task is whether participants had sufficiently learned the task before commencing the experimental phase. The number of practice trials ranged from 6 to 16, with an average of 10.7. Figure 6.1 shows the overall error rate per trial in the experimental phase. A Page's Trend test was done to test whether there was a learning effect present in the data. This revealed a significant downwards trend in the number of errors as a function of trial, $L = 3930, p < 0.05$. This indicates that

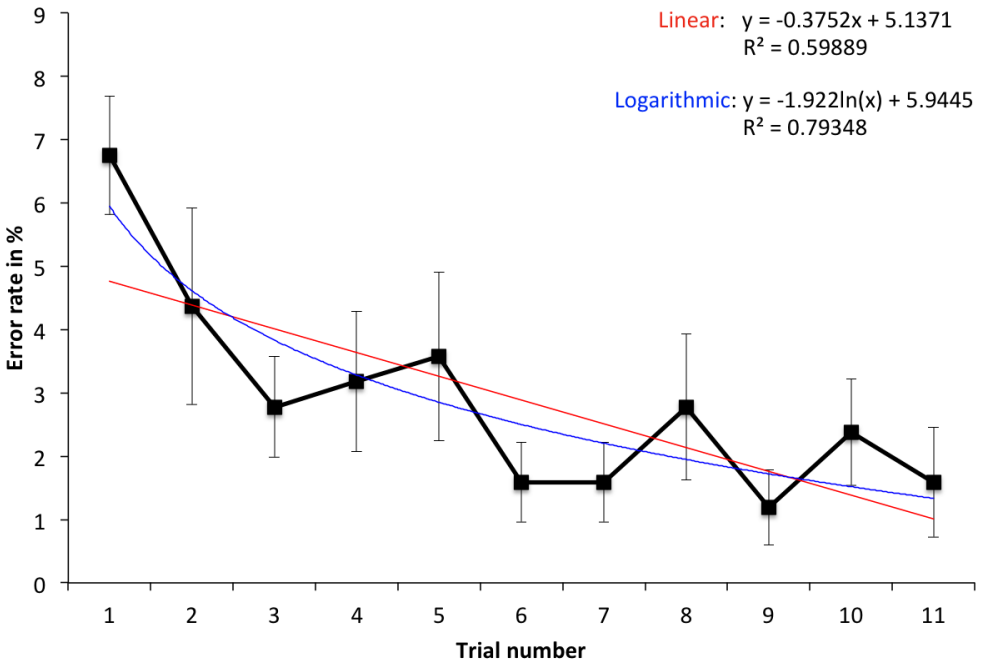


Figure 6.1: *Error rates for each trial on the Spy task. Error bars represent the standard error of the mean.*

participants were still learning the task in the experimental phase. Moreover, to determine which model best describes the learning present in the data, regression lines were fitted to the graph in figure 6.1. With an R^2 of 0.79, the best fit was a logarithmic regression line, which is indicated in blue on the figure. In addition, a linear regression line was fitted, but with an R^2 of 0.60 its fit was poorer.

The error, step time and eye movement data are summarised in table 6.1, and are further analysed in the sections below.

	Type of Step		
Measure	**Device-Oriented**	**Task-Oriented**	**Correlation**
Error Rate	5.1% (3.1%)	1.6% (1.2%)	$\tau = 0.05$
Step Time	2.6s (0.68s)	1.8s (0.38s)	$\tau = 0.56*$
Eye Movements:			
Average Fixation Length	515ms (110ms)	630ms (122ms)	$\tau = 0.60*$
Total Fixation Duration	2262ms (676ms)	1563ms (332ms)	$\tau = 0.64*$
Number of Fixations	4.91 (1.90)	2.98 (0.79)	$\tau = 0.87*$

Table 6.1: *Summary of the results on the Spy task. Average error rates, step times and eye movements (standard deviations in brackets) are shown in rows, for both device- and task-oriented steps. Eye movements are further split into average fixation length, total fixation duration and number of fixations. The last column shows the correlation between device- and task-oriented steps for each measure using Kendall's tau. A * denotes that a correlation is significant.*

6.2.3.1 Error Rates

A total of 106 errors were made against 11 x 12 x 28 = 3696 opportunities, leading to an overall error rate of 2.87%. The error rate on device-oriented steps was 5.1% (SD = 3.1%), whereas on task-oriented steps the error rate was 1.6% (SD = 1.2%). The data was not normally distributed, and therefore a non-parametric test was used. A Wilcoxon signed-rank test revealed that the difference was significant: $Z = -2.36, p < 0.05, r = 0.68$.

There was no significant correlation between performance on device-oriented and on task-oriented steps using Kendall's tau, $\tau = 0.051, p = 0.83$.

6.2.3.2 Step Times

Step times were computed for device- and task-oriented steps. Erroneous cases were excluded, because the hypothesis is concerned with the time it takes for the target goal to reach the required activation level to be executed. When an error is made, the target goal never reaches this level of activation and is as such not relevant. On device-oriented steps, participants took on average 2.55 (SD = 0.68) seconds to complete the step, whereas for task-oriented steps this was shorter at 1.78 (SD = 0.38) seconds. This difference was significant, $Z = -2.80, p < 0.01$.

Interestingly, unlike on the error rates, there *was* a significant correlation between the step times on device- and task-oriented steps, as demonstrated by Kendall's tau, $\tau = 0.56, p < 0.05$. However, no significant correlation was found between overall error rates and overall step times, $\tau = 0.23, p = 0.37$.

As described by Fitts' law (Fitts, 1954), the physical distance to the next interaction element affects the time taken to complete this step. Since distance is not controlled for in the Spy task, this could explain the difference in step time found for device- and task-oriented steps. However, there was no correlation between the step time and the distance to the next interaction element, $\tau = 0.14, NS$. Moreover, it was found that the distance to the next step is not different for device- and task-oriented steps, as demonstrated by a Mann Whitney U test: $Z = -1.48, NS$.

6.2.3.3 Eye Movements

Like the step times, only correct cases were included for all three eye movement measures. Average fixation lengths (AFLs) were computed for device- and task-oriented steps. On device-oriented steps, the AFL was 515ms (SD = 110ms), while on task-oriented steps it was longer at 630ms (SD = 122ms). This difference was significant, $Z = -2.70, p < 0.01$.

The total fixation duration (TFD) per step was also computed, averaged for device-oriented steps and for task-oriented steps respectively. On device-oriented steps, the total fixation duration was 2262ms (SD = 676ms), while on task-oriented steps this was shorter at 1563ms (SD = 332ms). This difference was significant, $Z = -2.80, p < 0.01$.

The average number of fixations (NoF) per step was computed for device- and task-oriented steps. On device-oriented steps, participants took on average 4.91 (SD = 1.90) fixations to complete the step, whereas for task-oriented steps this was less at 2.98 (SD = 0.79) seconds. This difference was significant, $Z = -2.80, p < 0.01$.

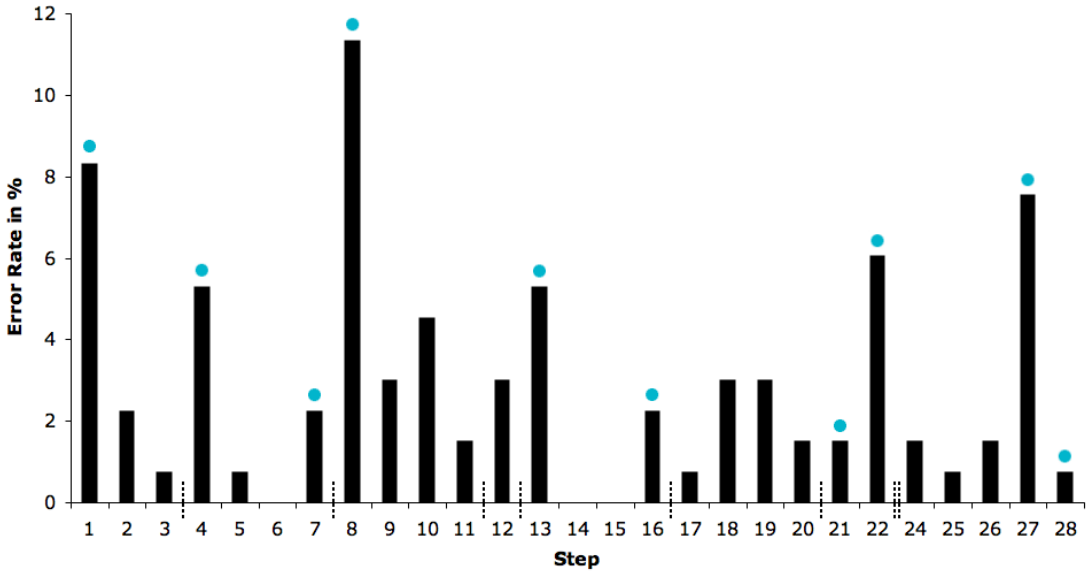


Figure 6.2: *Error pattern observed on the Spy task. Device-oriented steps are marked with a circle above the bars. Dotted lines on the X-axis indicate subtask boundaries, the double dotted line marks the boundary between flying the plane and delivering the message.*

6.2.3.4 Individual Steps

Figure 6.2 shows the pattern of errors on each of the individual steps, averaged over all participants. The graph demonstrates that the six steps with the highest error rates (above the 5% systematicity level (see Byrne and Bovair, 1997)) are all device-oriented, while none of the task-oriented steps reach this level. The graph shows more variation between steps than on the Doughnut task, with a number of task-oriented steps showing relatively high error rates, and a number of device-oriented steps that have error rates below average. The graph also shows that there are no steps with exceptionally high error rates, as was the case on the Doughnut task. Interestingly, steps 4 and 13 are similar in their function (both turn on the radio), as are steps 7 and 16 (radio off), which appears to be reflected in their highly similar error rates.

A number of steps used identical interface items but had different functions and therefore different relevance to the task. Five of these were device-oriented (switch radio on (4, 13), switch radio off (7, 16), release brakes (8)), and five were task-oriented (extend wingflaps (10), take in landing gear (11), retract wingflaps (17), extend landing gear (18), apply brakes (20)). This allows for the direct comparison between device-oriented (M = 5.3%, SD = 3.99%) and task-oriented steps (M = 2.3%, SD = 2.9%). A Wilcoxon signed-rank test reveals a trend but no significant difference in this subset of steps, $Z = -1.81, p = 0.070$.

6.2.4 Discussion

The current experiment studied the difference between device- and task-oriented steps on a new task that addressed many of the issues with the Doughnut task. It aimed to test whether the findings of the first experiment (see chapter 5) are robust, and introduced step times and eye movements as new measures.

6.2.4.1 Error Rates

The experiment found that device-oriented steps are more prone to errors than task-oriented ones on the Spy task. This demonstrates that the difference in error rates on device- and task-oriented steps is robust, and holds even when a number of confounding factors is accounted for.

Analysis of the number of errors per trial showed an improvement over time. Moreover, the logarithmic curve on the graph shows that the downward trend is slowing, but is still present towards the end of the experiment. This indicates that participants were still acquiring the task while executing the experiment. The effect was most pronounced on the first trial of the experiment, after which the error rate dropped off more quickly. This initial rapid improvement can be explained in two ways: first, it is likely that participants improved over time as they gained more experience in doing the task. However, it is also possible that it reflects a 'start-up effect'. Since there was a short period of time between the end of the training and the start of the first experimental trial (as the eyetracker was calibrated), participants may have taken some time to 'get back into it'. The presence of a learning effect is a potentially serious issue for the current experiment, because it is aimed at studying routine, well learned performance. The finding of a learning effect indicates that participants' level of performance may not yet be completely routinised. This is somewhat surprising, because the number of training trials was greatly increased after similar concerns on the experiments using the Doughnut task. Since the Spy task is arguably a more complex task than the Doughnut task, it is likely that participants simply need more time to acquire the task sufficiently.

A factor that the current task did not completely account for is the interface, since there were a number of different interface items that could be used, such as flip switches and simple buttons. The different interface items were all designed to have similar colours and sizes, thereby minimising any differences in visual salience. Moreover, all steps in the Spy task required only a simple mouseclick to execute it (with the exception of the destination input step (2), which required recognition of the destination in the drop-down list), so other than remembering the next step, no additional mental operations were necessary. Nevertheless, it is possible that some unknown factor was responsible for the difference found between device- and task-oriented steps. Therefore, the results also looked at a subset of steps that were executed using identical interface items: those that flipped the switches on the bottom left of the interface. There were ten steps that were executed using these switches, five device-oriented and five task-oriented ones. The findings showed that the difference between device- and task-oriented error rates on these steps was not significant, but showed a clear trend in this direction. This is encouraging, but it also highlights the need of future tasks to fully control for interface and step difficulty issues.

Unlike on the Doughnut task, there were no steps that had exceptionally high error rates. This is likely to reflect the absence of strong post-completion and device-initialisation steps, and indicates that it is less likely that there are such confounding factors that bias the results.

6.2.4.2 Step Times

Step times were longer on average for device-oriented steps than for task-oriented ones. This is in line with expectations, and with the hypothesis that device-oriented steps have lower activation levels than task-oriented ones. However, an obvious difficulty with the step times measure on the Spy task is that the distances between interface items are not controlled for. The results showed that the distances were not correlated with the step times, and the average distances were not different for device- and task-oriented steps. Nevertheless, it cannot be completely ruled out that this played a role.

6.2.4.3 Eye Movements

Participants' eye movements were different for device-oriented and task-oriented steps: average fixation lengths were shorter, total fixation durations were longer and the number of fixations was larger on device-oriented steps. This is extremely interesting, because it could indicate that cognitive processes underlying these steps are different. It is well-known that different cognitive mechanisms can give rise to different eye movements (e.g. Epelboim et al., 1995). However, it is much more difficult to determine the type of cognitive mechanism involved in each step. One possibility is that on device-oriented steps, there is more uncertainty about what the next step is, and therefore participants may spend more time searching the interface for visual cues. On task-oriented steps, on the other hand, they may be more concerned with finding the location of the next interface item, and less with finding cues to determine what the step may be. This is in line with Henderson (2003), who found that fixation lengths may be relatively short when a person is searching a task interface. However, he also notes that fixation lengths are influenced by a large number of factors, including visual features such as luminance and image contrast. As such, it is difficult for the current findings to make any claims about the cognitive processes underlying device- and task-oriented steps. This hypothesis will further be addressed in experiment 4, which tests the influence of cue availability on device- and task-oriented steps.

It should be noted that the number of fixations and total fixation duration are both dependent on the quality of the eye movement data recorded. The Tobii eyetracker employed in this study does not use a head rest to ensure the participant's eyes are always in the same location. While this is beneficial in allowing the participant to relax and thereby reduce self-monitoring, it means that participants can occasionally move beyond the 'window' from which the eyetracker is able to record. This was minimised by the experimenter continuously monitoring the tracker status, and ensuring the participant's eyes were within the eyetracker's boundaries. Nevertheless, it is possible that there is a small margin of error in the number of fixations and total fixation duration data. However, there is no reason to believe that this affects one type of step more than another. Moreover, since a large amount of data is collected for each participant, this margin of error is likely to be very small in comparison to the effects found.

6.2.4.4 Correlation

As in experiment 1, no significant correlation between error rates on device- and task-oriented steps was found. However, there *was* a significant correlation between *step times* on these steps. This dissociation is somewhat puzzling, because the main hypothesis of the current work argues that both error rates and step times are determined by the activation levels. This finding could indicate that different mechanisms are responsible for the errors on device- and on task-oriented steps. In addition, no significant correlation was found between overall error rates and overall step times. Thus, participants who were faster were not necessarily more or less accurate then slower participants. This indicates that there was no significant speed-accuracy trade-off.

In short, the results are encouraging and indicate that the distinction studied in this thesis is real and useful, and extends beyond the findings of the Doughnut task. The findings that error rates are higher and step times are longer on device-oriented steps are in line with the hypothesis that activation levels are lower on these steps. It supports the idea that it is more difficult for device-oriented steps to compete with other memory items. The finding that eye movements are different for device- and task-oriented steps is extremely interesting, and indicates that they may have different cognitive mechanisms or are represented differently in memory. The next experiment will address this finding further.

6.3 Experiment 4

6.3.1 Introduction

The previous experiment demonstrated that the finding that device-oriented steps are more problematic is robust. Moreover, it found that eye movements were different for device- and task-oriented steps. Previous work (as reviewed by Henderson (2003)) has demonstrated that different cognitive processes give rise to different eye movement patterns. Therefore, the previous results could indicate that the underlying cognitive processes may be different for device- and task-oriented steps. However, a difficulty is that it is not directly possible to determine what these processes are from the eye movements.

The main hypothesis of the current thesis is that device-oriented steps are more problematic than task-oriented steps. They are more difficult to remember, and therefore it would be expected that participants make more use of visual cues in the task interface to remember device-oriented steps. This is in line with the soft-constraints hypothesis (Gray et al., 2006), which argues that people often use the fastest strategy to accomplish some goal or determine the next step. Lower activation levels on device-oriented steps means that they take longer to execute (as demonstrated in experiment 3), and as such participants may be more likely to use information from the task interface to help determine the next step.

Moreover, Li et al. (2008) suggested that associative links between procedural steps may not be as strong on some steps in a procedure, for instance the selector steps on the Doughnut

task. Instead, participants have to rely on other cognitive mechanisms to remember these steps. This could further predict that device-oriented steps rely more on external cues in the task interface, rather than on these internal cues.

Therefore, it is predicted that any disruption of the task interface should affect device-oriented steps more severely than task-oriented steps. The current experiment tests this by manipulating the availability of visual cues on device- and on task-oriented steps. This is done by occasionally hiding interaction elements from the task interface. Altmann and Trafton (2002) predicted that "the experimental manipulation of cue availability [...] should affect the ability to retrieve goals from memory". If device-oriented steps are already more difficult to remember than task-oriented ones, it is expected that they will be more affected by a lower cue availability than task-oriented steps. Effects are likely to be seen in higher error rates and longer step times on device-oriented steps.

6.3.2 Method

6.3.2.1 Participants

Twelve undergraduate (seven female) students took part in the experiment; all were paid £5 for their time. Only participants who had not taken part in the previous study were allowed to take part.

6.3.2.2 Stimuli

The same Spy task as in the previous study was used, with an adjustment to allow for the manipulation of cues. It was argued that the visual representation of the step, such as a button, text input field or flip switch, was the most powerful part of the visual cue. Temporarily removing this visual representation from view was therefore thought to reduce the strength of the visual cue on the step.

As such, the availability of visual cues was manipulated by temporarily hiding the relevant interaction elements from the interface. For instance, when the destination input had to be

(a) Confirmation button visible

(b) Confirmation button hidden

Figure 6.3: *Example of how a step is hidden on the Spy task: (a) The destination confirmation button is visible, and (b) hidden from view. Note that the area where the button normally appears is still functional: clicking it will accomplish the confirmation step and subsequently make the button reappear.*

confirmed, the confirmation button was hidden from the interface, instead simply showing the background image (see figure 6.3 for an example).

The removal of the interaction element did not mean that the step could not be carried out: the underlying area on the interface was still functional, so clicking on it completed the step in the same manner that clicking on the actual interaction element did. In all cases, the interface items were sufficiently large that participants did not have to 'click around' to find them.

The hidden interaction elements were changed on each step during the procedure. They were chosen pseudo-randomly: in approximately 50% of steps, the element corresponding to the next correct step was hidden, and in the other 50% of cases it was a random other step. This ensured that the hidden step was not always the correct one, which would have allowed participants to simply look for the empty area on the screen to help them determine the next step.

To stop the disappearance of a step from grabbing visual attention and thereby inadvertently cueing the next step, the interface element was always hidden a step early. As a result, the hiding of an interaction element had to last for two consecutive steps: the current, relevant step and the step directly preceding it. Thus, two steps were hidden at all times, and on each next step only one of them was changed. For instance, take a sequence of steps, $1 \rightarrow 2 \rightarrow 3 \rightarrow 4$, with corresponding interaction elements [A], [B], [C] and [D]. At the start of this sequence (0), there is a 50% chance that elements A and B are hidden, respectively (disregarding for the moment the other 50%, which led to a 1 in 26 chance[1] that any particular step was hidden), but for this example they are both visible.

(0) [A] [B] [C] [D]

When step 1 is executed by clicking [A], the visibility of [C] is determined, again with a 50% chance that it will be hidden (which it will be in this example). At this point (1), element [B] will still be visible, while element [C] is hidden.

(1) [A] [B] [D]

Element [B] is then clicked, and the visibility of [D] is determined (it will be hidden) (2).

(2) [A] [B]

At this point, [C] is still hidden, and [D] is also hidden. Clicking on the hidden C area will then determine E (3), and so on. Crucially, [C] is visible again.

(3) [A] [B] [C]

[1] Two of the 28 steps were not hidden, see the Results section for a discussion.

It should be noted that after the correct completion of a step, the interface element always becomes visible, otherwise participants may become confused about whether they correctly executed the step.

6.3.2.3 Design

A within-participants design was used, with two independent variables. The first independent variable was the type of step, which could be device-oriented or task-oriented. The second independent variable was cue availability. This variable had two levels: the target interaction element was visible, or it was hidden.

There were three dependent variables: error rates, step times and eye movements (average fixation length, total fixation duration, and number of fixations). These were defined in the same manner as in the previous experiment (see section 6.2.2.3).

6.3.2.4 Procedure

The procedure was similar to the procedure of experiment 3, see section 6.2.2.4 for a full description. The instructions were altered slightly to include an explanation of the hidden steps. Participants were informed that interaction elements would occasionally be invisible, but were told that that the underlying area was still fully functional. The first two training trials did not have any invisible steps, to allow participants to get used to the task first. After this, steps' interaction elements disappeared as normal.

6.3.3 Results

Data from 12 participants was recorded. The number of practice trials ranged from 5 to 13, with an average of 10.5 trials before two were completed without any errors. As in experiment 3, step 23 (activate the secret compartment) was excluded from analysis, because it was always hidden. Moreover, two further steps were excluded in the current experiment. Step 28 (dismissing the secret compartment) could not be hidden, because it involved using the operating system window manager (clicking the Windows XP 'Close' button on the top right of the window). Moreover, a bug in the task software caused step 25 (entering the secret message) to always be marked as visible in the data output files, even though it was hidden correctly during the experiment. This made it impossible to determine the status of this interaction element for the purpose of analysis.

All data is summarised in table 6.2, and is further analysed in the sections below.

6.3.3.1 Error Rates

Overall, participants made 121 errors. The number of opportunities for error was 11 x 12 x 25 = 3300, giving an overall error rate of 3.67%.

It was hypothesised that error rates are higher on device-oriented and hidden steps than on task-oriented and visible ones, respectively. Moreover, an interaction between the two was expected. A mixed-design ANOVA was used, although the data were not normally distributed. This was done because no non-parametric alternative test was available, and the

		Type of Step	
Measure	**Item visibility**	**Device-Oriented**	**Task-Oriented**
Error Rate	Visible	5.8% (5.0%)	0.73% (1.1%)
	Hidden	7.2% (7.1%)	2.4% (1.8%)
Step Time	Visible	2855ms (851ms)	1983ms (445ms)
	Hidden	2785ms (596ms)	2072ms (452ms)
Eye Movements:			
Average Fixation Length	Visible	477ms (68ms)	533ms (84ms)
	Hidden	494ms (76ms)	549ms (81ms)
Total Fixation Duration	Visible	2683ms (761ms)	1980ms (488ms)
	Hidden	2455ms (429ms)	1916ms (357)
Number of Fixations	Visible	5.8 (2.3)	3.8 (1.3)
	Hidden	5.1 (1.4)	3.6 (1.0)

Table 6.2: *Summary of the results of the Spy task. Average error rates, step times and eye movements (standard deviations in brackets) are shown for both device- and task-oriented steps. For each measure, the first row indicates visible cases, and the second row indicates hidden cases. Eye movements are further split into average fixation length, total fixation duration and number of fixations.*

majority of previous work on error has successfully used ANOVAs to analyse error rates (see section 5.3.3 for a discussion). The ANOVA showed that there was indeed a significant difference between the two types of steps, with device-oriented steps giving rise to higher error rates (M = 6.45%, SD = 5.70%) than task-oriented steps (M = 1.59%, SD = 1.27%), $F(1, 11) = 8.83, p < 0.05$. There was also a significant difference between hidden and visible steps, with hidden steps having higher error rates (M = 3.99%, SD = 2.65%) than visible ones (M = 2.46%, SD = 2.03%), $F(1, 11) = 5.67, p < 0.05$. However, there was no significant interaction between these two variables, $F(1, 11) = 0.03, NS$.

6.3.3.2 Step Times

Step times were expected to be longer for device-oriented and hidden steps. As in the previous experiment, only correct cases were included. Indeed, they were found to be longer on device-oriented steps (M = 2838ms, SD = 673ms) than on task-oriented ones (M = 2026ms, SD = 428ms), and this difference was significant: $F(1, 11) = 56.2, p < 0.001$. However, there was no significant effect of step visibility, $F(1, 11) = 0.005, NS$, with a mean step time of 2310ms (SD = 541ms) on visible steps, and 2327ms (SD = 496ms) on task-oriented steps. There was also no significant interaction between these two dimensions, $F(1, 11) = 0.79, NS$.

No significant correlation was found between overall error rates and overall step times, $\tau = -0.18, p = 0.94$.

6.3.3.3 Eye Movements

Incorrect cases were excluded. It was expected that fixations were shorter and more numerous on device-oriented steps, but there were no strong prediction on the effect on hidden and

visible steps. There was indeed a significant effect of step type on the average fixation length, $F(1, 11) = 9.29, p < 0.05$, with shorter fixations on device-oriented steps (M = 482, SD = 67) than on task-oriented steps (M = 540ms, SD = 80ms). There was no significant effect of step visibility, $F(1, 11) = 3.51, NS$, with a mean step time of 506ms (SD = 70ms) on visible steps and 523ms (SD = 70ms) on hidden steps. There was also no interaction between these two variables, $F(1, 11) = 0.001, NS$.

The total fixation duration was longer on device-oriented steps (M = 2570ms, SD = 565ms) than on task-oriented steps (M = 1950ms, SD = 420ms). This difference was significant, $F(1, 11) = 29.3, p < 0.001$, There was no significant effect of step visibility, $F(1, 11) = 1.82, NS$, with a total fixation time of 2305ms (SD = 595ms) on visible steps and of 2109ms (SD = 348ms) on hidden steps. There was also no interaction, $F(1, 11) = 1.19, NS$.

A similar pattern was also found on the number of fixations. The number of fixations was significantly higher on device-oriented steps (M = 5.5, SD = 1.8) than on task-oriented ones (M = 3.7, SD = 1.1), $F(1, 11) = 38.7, p < 0.001$, There was no significant effect of step visibility, $F(1, 11) = 2.54, NS$, with the number of fixations at 4.7 (SD = 1.7) on visible steps and at 4.1 (SD = 1.1) on hidden steps. There was also no interaction between the two variables, $F(1, 11) = 1.51, NS$.

6.3.4 Discussion

The current study tested the hypothesis that the absence of visual cues will have a greater impact on device-oriented than on task-oriented steps. It replicated the results of experiment 3, showing that device-oriented steps give rise to higher error rates, longer step times and different eye movements than their task-oriented counterparts. As in the previous experiment, no correlation between the error rates and step times was found, indicating that there was no speed-accuracy trade-off.

The results indicated that cue availability affected error rates, with higher error rates on both types of step when the visual cue was invisible. However, this finding did not extend to the step times nor to the eye movement measures, which were all unaffected by cue availability. This is an interesting finding, that cannot easily be explained. One possibility could be that there was a certain degree of confusion when the relevant interaction element was missing, which could have caused an increase in errors. However, this is not consistent with the lack of an effect on step times. If participants were confused, it would be expected to affect step times in the first instance, because they may hesitate before taking action. This effect would be expected to persist even in correct cases.

In terms of activation levels, it was hypothesised that the visual cue will provide a boost for the activation level of the target step, and so help it to become the highest. However, if such a visual cue is not available, then the step will not receive a boost (or to a lesser extent). This means that it is more likely that it does not reach the required activation level to be executed. However, it also means that the step should take longer to be executed, which was not supported

by the current experiment. In short, these results are not in line with the expectations, but it is not immediately clear how they can be accounted for.

Moreover, it was found that cue availability did not affect device- and task-oriented steps differently on any of the measures, including error rate. Again, this is not in line with the predictions. It was hypothesised that if device-oriented steps have lower activation levels, then the lack of an activation boost may lead to higher error rates and longer step times than on task-oriented steps with hypothesised higher activation levels. However, this was not found.

One explanation for this lack of an effect lies in the assumption that the visual representation of the interaction element is responsible for the boost in activation. This is not necessarily supported by other studies. For instance, Ratwani (2008) reported that location plays an important role in task resumption after an interruption. In the same way, the location where the interaction element is supposed to be may provide the cue, despite the visual absence of the cue. In this scenario, participants may remember *where* the next step is supposed to take place, but may not necessarily remember *what* it is. If location indeed plays a significant role in providing an external cue, then the current experimental manipulation has (partially) failed to achieve its goal.

Another explanation is that participants' strategies for using the information available to them (from memory or from the task interface) are more complex than assumed. For instance, the soft-constraints hypothesis (Gray et al., 2006) argues that people generally balance the strategies they use to minimise the time taken to complete the task. This means that if they readily remember what the next step is, they may be less likely to use the task interface to determine what to do. Conversely, if the interface provides a strong visual cue for the next step, they may refrain from doing a more time-consuming memory retrieval. As for instance Gray and Fu (2004) show, people readily adapt these strategies, depending on what information is easily available to them. In the current study, participants were aware that interface items occasionally disappeared, as this happened in 50% of the steps. This may have caused them to adapt their strategy in ways not anticipated by the current study. An improvement for future studies can be to reduce the frequency with which interface items disappear, as participants may be less likely to adapt their strategy if it happens only occasionally. In any case, it seems clear that the manner in which visual cues are used is more complex than the current study has accounted for.

Another explanation is related to Li et al.'s (2008) findings on the Doughnut task. They argued that certain device-oriented steps may not rely on procedural links to the same extent as task-oriented steps. Instead, Li et al. (2008) argue that their execution relies on a more deliberate and less automatic mechanism. For the current experiment, it was hypothesised that these mechanisms could include using information from the task interface. However, since this hypothesis was not confirmed, it is possible that other strategies were used instead on which the experimental manipulation had no effect. Perhaps participants employed strategies such as

problem solving or retrieving declarative knowledge about the correct order of steps to determine what the next step must be. However, the current study may not have been able to capture these processes.

In short, it appears that the current study has a number of problems that cannot easily be resolved. While the manipulation is exceedingly simple, in hindsight, the effect it has on participants is likely to be far more complex than originally expected. First, it is not clear whether simply hiding a step from the task interface reduces the cue availability of the step. Second, even if it does, the implicit assumption of the current study that participants' strategies remain stable throughout the experiment is problematic. It is not clear how to address these issues. Cues from the location (or other unanticipated sources) cannot easily be made less available without making drastic changes to the task interface. Moreover, any such changes may have further effects on the strategies participants use to determine the next task step. As such, the current thesis will move away from this line of work. It was originally intended to further explore an interesting finding from one of the core studies, but is in itself not crucial for the main hypotheses presented in this work. Therefore, the two experiments presented in the next chapter focus on the original hypothesis of lower activation levels on device-oriented steps.

6.4 Overall Discussion

The current chapter presented two studies that used the Spy task to study device- and task-oriented steps. Both found that device-oriented steps have higher error rates, longer step times and different eye movements than task-oriented steps. The second study, however, was unsuccessful in further studying the cognitive processes underlying the two types of steps.

The findings that device-oriented steps have higher error rates and longer step times are in line with the predictions. They are likely to reflect that these steps are more difficult to remember. Moreover, these results support the hypothesis that device-oriented steps have lower activation levels than their task-oriented counterparts. This lower activation leads to a higher chance that the step will not be active enough to be executed. Moreover, even if the step gains sufficient activation, it will take longer to do so.

Unlike on the Doughnut task, device-oriented steps on the Spy task were not immediately followed by a strong task-oriented step. As such, the alternative hypothesis that the errors are caused by erroneous steps gaining too much activation is less likely to apply here. After all, while theoretically possible, there seems to be little reason to believe that such highly active steps occur selectively after device-oriented steps.

Another error mechanism relies on the degradation of task representations: the more degraded the representation, the greater the chance that it loses out to a similar representation. If device-oriented steps have weaker or more degraded representations, then this explanation could account for the increase in error rates on these steps. However, as discussed in section 2.4.3, this approach does not appear to address the timing of steps, and can thus not account for the findings on the step times.

6.4.1 The Spy task

The Spy task was designed to address some of the shortcomings of the Doughnut task as a task to study device- and task-oriented steps. An important improvement of the Spy task is that it includes a wide variety of different device- and task-oriented steps. While it does not directly control for the different types of steps, it is thought that the inclusion of a wide variety of these steps provides a reasonable approximation of such more direct control.

The error pattern on the Spy task is less pronounced than the one found on the Doughnut task. This may reflect, in part, the absence of a strong post-completion step (with false completion and competing signals) on the Spy task. However, while the steps with the highest error rates are all device-oriented, and the majority of task-oriented steps have low error rates, there are also a number of device- *and* task-oriented with an 'intermediate' error rate. This is likely to reflect that, besides a step's relevance to the task goal, there are other factors that further determine the likelihood of an error on a given step.

A shortcoming of the Spy task is that the interface is not completely controlled for. While the analysis of experiment 3 demonstrated that the distance between steps does not affect performance, the size and visual salience of the buttons is not taken into account. The interface was designed to minimise these effects, but it cannot be completely ruled out that this has an effect on performance. As such, an ideal task uses identical interaction elements and distances, while varying only the relevance of steps to the task goal. Such a task is developed for the next two experiments, which are presented in chapter 7.

6.5 Conclusion

The experiments presented in the current chapter provide support for the hypothesis that device-oriented steps are more problematic because they have lower activation levels. Both main findings that error rates are higher and step times are longer on device-oriented steps are in line with this. The alternative mechanisms for error generation can only partially explain the current results.

However, while the Spy task seems quite successful in studying device- and task-oriented steps, it is not perfect. Improvements can be made with regards to the task interface, which must control for the visual appearance of the task in terms of colour, size and salience, as well as the distances between the interaction elements. In an ideal task, device-oriented steps and task-oriented steps are executed in exactly the same manner on exactly the same interaction elements, with only the relevance of the step to the task goal being manipulated. To address this, a new task is developed that accounts for all these issues. This task, the Frankenstein task, will be studied in detail in the next chapter.

Take-home message:

The experiments on the Spy task demonstrate that device-oriented steps are more problematic than task-oriented ones, in terms of error rates and step times. The most likely explanation for this is that device-oriented steps have lower activation levels. However, the task interface was not controlled for, and therefore further studies are needed.

Chapter 7

Direct Manipulation of Goal Relevance Using the Frankenstein Task: Experiments 5 and 6

In this chapter:

- A new experimental task is introduced, which manipulates the goal-relevance of steps and controls for the task interface.
- Two studies using this Frankenstein task are reported.
- The first study investigates error rates, step times and the proportion of omission errors.
- The second study assesses the effect of working memory load on device- and task-oriented steps.
- The findings are in line with the expectations, and support the main hypothesis.

7.1 Overview

The previous two chapters presented studies using the Doughnut and Spy tasks, which found that device-oriented steps are more error prone and take longer to execute than task-oriented steps. In addition, experiment 2, using the Doughnut task (see section 5.3), found some preliminary evidence for the hypothesis that working memory load affects device-oriented steps more than task-oriented steps.

However, as discussed previously, the tasks had several drawbacks, which made it difficult to generalise the results to other situations. For instance, on the Doughnut task, a large effect was seen on the post-completion step, which was further 'encouraged' by the presence of a competing signal and a false-completion signal. This makes it possible that the effects found were largely due to this one particular step. Moreover, while the Spy task made significant improvements over the Doughnut task and eliminated such confounds, it did not account for

differences in the task interface, such as button size and distance between interaction elements. As such, the observed effects could be due to these interface factors rather than to the relevance of the steps to the main goal.

The current chapter introduces a new experimental task that takes a novel approach to control for these confounding factors. It uses a single task interface, but provides two 'cover stories' for its function and operation. These cover stories contain a number of device- and task-oriented step pairs. In one cover story, a step makes a direct contribution towards the main goal, while in the other it is only relevant to the operation of the device. Crucially, the same interaction element is used for both steps, and they are executed at the same point in the task sequence. This means that factors related to the task interface are controlled for, allowing the direct investigation of the relevance of steps to the goal of the task.

The experiments presented in this chapter address all four behavioural predictions of the main hypothesis. The first of the two (experiment 5) has two aims. First, it establishes whether the Frankenstein task is successful in the study of the relevance of steps to the task goal. Second, it investigates error rates, step times and the proportion of omission errors on the different types of steps. The second study (experiment 6) is concerned with the influence of working memory load on these measures. It uses a similar set-up to experiment 2 (see section 5.3), providing working memory load using a secondary monitoring task. The chapter finishes with a discussion of the results and an analysis of the Frankenstein task as a task to study the goal-relevance of steps.

7.2 Experiment 5

7.2.1 Method

7.2.1.1 Participants

Thirty-two participants (twenty-one female) took part in the experiment. All were students at UCL, and they were paid £7 for their time and effort. They were aged between 18 and 30.

7.2.1.2 Materials

A new task was developed for this study, called the Frankenstein task (described in detail in section 4.2.3). This is a routine procedural task partially analogous to the Doughnut task, and it manipulates the goal-relevance of steps. To achieve this, it has two conditions: in the first, participants are asked to create monsters according to a set of specifications, and in the second, participants are asked to disassemble them instead. Crucially, the task interface and the order in which steps have to be executed are identical for both conditions[1]. The only difference between the two conditions is the cover story, and therefore the steps' relevance to the task goal. This allows the direct comparison of device- and task-oriented steps, while measuring error rates, step times and proportion of omission errors.

[1] With the exception of different button labels, and a number of animations were different to fit the relevant story.

Step	Create	Classification	Classification	Destroy
3	Activate slider	Device	Task	Use slider
4	Use slider	Task	Device	Deactivate slider
6	Activate body widget	Device	Task	Rip off head
9	Activate leg widget	Device	Task	Rip off arms
12	Activate arm widget	Device	Task	Rip off legs
15	Activate head widget	Device	Task	Rip off body
18	Paint monster	Task	Device	Activate programmer widget
21	Confirm programming choice	Device	Task	Wipe monster brain
22	Position laser	Task	Device	Activate laser
24	Release monster	Task	Device	Deactivate machine

Table 7.1: *Overview of the target steps on the Frankenstein task. The middle two columns indicate whether a step is device- or task-oriented for each condition. Note that steps 3 and 4 are repeated twice, once for each coordinate.*

A number of target steps are included that are relevant to the task goal in one condition but not in the other, and vice versa. For instance, in the 'Create' condition, a particular button serves to activate the leg widget. This is not relevant to the main goal, and as such it is considered a device-oriented step. In the 'Destroy' condition, on the other hand, the same button does not activate the leg widget, but instead rips off the legs from the monster. This step, is highly relevant to the task goal of destroying the monster, and is therefore considered task-oriented. Crucially, in both conditions, the same interface is used, and the order in which steps have to be carried out is the same. The Frankenstein task contains 10 of these target step pairs, and device- and task-oriented steps are alternated between conditions (see table 7.1 for an overview of the 10 target step pairs. It should be noted that steps 3 and 4 are repeated twice, once for each coordinate. As such, a total of 12 target steps are executed on each trial. For a complete overview of all steps on the Frankenstein task, see table 4.3 in chapter 4). The remaining steps were non-targets, and acted as 'fillers' to enhance the task experience and improve the story.

As done by Byrne and Bovair (1997), a monitoring task was added to ensure that error rates were at a measurable level. They used a variant of the well-known N-back task for this purpose (Kirchner, 1958), which was also employed in the current study. Digits were read out to participants by the computer, at a rate of one digit every three seconds. Participants were asked to remember the last three digits at any point in time. A beep sounded after random intervals of between 3-10 digits, indicating that participants had to repeat the last three digits out loud. Participants carried out the Frankenstein task and the monitoring task concurrently.

The Frankenstein was presented on two adjacent computer screens, attached to the same terminal. Both screens operated at a resolution of 1280x1024; the screen to the left showed the Locator, while the one on the right showed the main Frankenstein task interface. Participants used a mouse to interact with the task, and could move the cursor freely between the two screens.

7.2.1.3 Design

A mixed design was used. The first independent variable was the type of step, which was varied within participants. It had two levels, device-oriented and task-oriented, with each participant doing 4 and 6 of each type of target step (or vice versa). The second independent variable was the 'create/destroy' condition, which was varied between participants. It had two levels, 'create monster', and 'destroy monster', which are 'mirror images' of one another. That is, if step X is device-oriented in one condition, it is task-oriented in the other, and vice versa. Note that this is a dummy variable, as it is used purely for experimental design purposes and has no theoretical meaning. As such, the two levels will further be referred to as 'flip 1' and 'flip 2'. Table 7.1 indicates which steps are device- and task-oriented in which condition.

7.2.1.4 Measures

Data from only the target steps was measured, as the remaining steps were not manipulated in terms of their relevance to the task goal. A number of measures were included. First, the number of errors on the target steps was recorded. As in the previous experiments, only one error could be made on a given step. Second, the step time for each step was recorded. The step time is defined as the time from the successful execution of the previous step until the next meaningful mouseclick. Third, the type of error was also recorded. An omission error was identified as skipping a single step. Any errors in which more steps were skipped or previous steps were repeated were counted as 'other' errors.

In addition to these behavioural measures, eye movement data was also collected, to test whether the findings of the previous chapter are robust. The average fixation length, number of fixations, and total fixation duration were recorded.

7.2.1.5 Procedure

Each participant carried out the task individually. First, they read an information sheet explaining the monitoring task (see appendix A.4.3). Each participant then practised the task by itself, to ensure they had understood and were able to do it. Each participant completed three practice runs.

Second, participants viewed a short computer animation that explained the story behind the Frankenstein task and what their task was. They then read the detailed on-screen instructions on how to use the task interface to complete their task. The story and instructions for both conditions are detailed in appendix A.4. Participants were asked to complete the task as quickly and accurately as possible.

After reading the instructions, participants observed the experimenter complete one trial of the task. They were then allowed to explore the task interface without the monitoring task for one trial. After this, participants practised the main task with the monitoring task. If an error was made during training, it was pointed out immediately with a sound and a pop-up screen that explained what had gone wrong and what the participant should do instead. All errors had to be corrected before the task could be resumed. Participants had to complete 3 trials without

errors before they were allowed to carry on with the experimental phase.

After training was finished, the eye tracker was calibrated and started. Participants then completed 11 experimental trials. Errors were pointed out with a sound, but no pop-ups with hints were shown. Again, all errors had to be corrected before participants could resume the task. After completing the experiment, participants were debriefed about the purpose of the experiment.

7.2.2 Results

The main dependent variables were error rate, step times and type of error. It was expected that on device-oriented steps, error rates are higher, step times are longer, and proportionally more omission errors are made. Eye movement data is also reported. Data from 32 participants was recorded; only target steps are analysed. The average number of training trials was 6.1, with a range of between 3 and 10.

7.2.2.1 Error Rates

A total of 218 errors were made on target steps during the experiment. Given a total opportunity for error of 32 participants x 11 trials x 12 target steps = 4224, the overall error rate is 5.16%. The previous studies on the Doughnut and Spy task showed a small improvement in performance over time, so this was also investigated in the current study. Figure 7.1 shows the error rate on target steps per trial on the Frankenstein task. A Page's Trend test was used to investigate whether participants made fewer errors as they progressed through the experimental phase. This test showed no significant decrease in error rates as more trials were completed, $L = 14096, NS$. This indicates that participants were effectively trained before commencing the experimental phase. Moreover, a trend line was fitted to the graph, which showed that a linear regression line had the best fit. However, with an R^2 of 0.22 this fit is still considered to be poor.

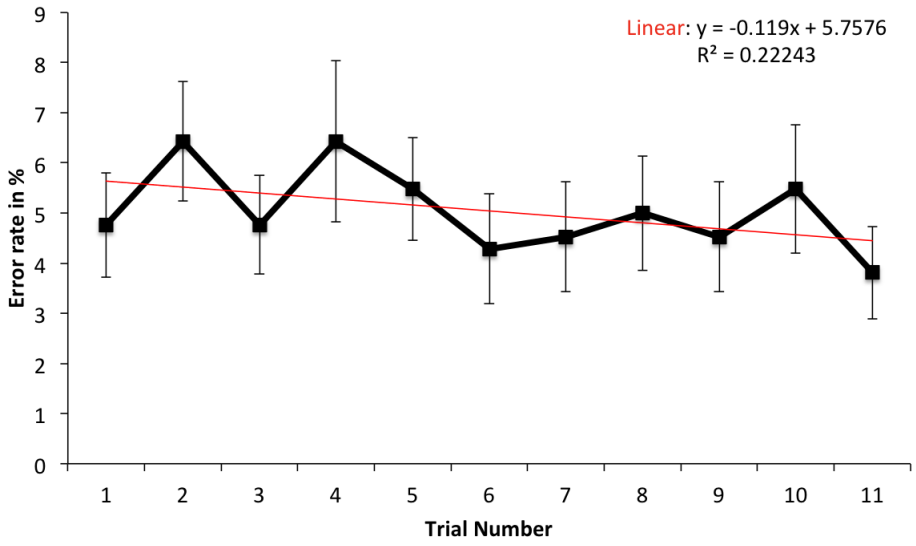


Figure 7.1: *Error rates on target steps for each trial on the Frankenstein task.*

	Error Count (Opportunity)	Mean Error Rate (SD), in %
Total	218 (4224)	5.16 (3.57)
Device-Oriented	145 (2112)	6.96 (5.97)
Task-Oriented	73 (2112)	3.55 (3.42)

Table 7.2: *Mean error rates on device- and task-oriented steps across all participants. The second column shows the total number of errors made for each step, and the opportunity for error. The third column shows the average error rate and standard deviation for each step.*

Table 7.2 shows the average overall error rates and standard deviations for task and device-oriented steps. A 2 x 2 mixed-design ANOVA was done (though note that the error data was not normally distributed). The type of step was the within-subjects variable, and condition (flip 1 or flip 2) was the between-subjects variable. This revealed a significant main effect of the type of step, $F(1, 30) = 10.31, p < 0.005, \eta_p^2 = 0.26$, indicating that error rates were higher at device-oriented than at task-oriented steps. There was no significant main effect of condition (flip 1 or flip 2), $F(1, 30) = 0.00, NS, \eta_p^2 = 0.00$. No interaction between the two dimensions was found, $F(1, 30) = 1.05, NS, \eta_p^2 = 0.03$.

However, given the violation of the assumption of normality, these results should be interpreted with some caution. As in experiment 2, it was determined whether transforming the data would improve the moderate skew present in the data. While a logarithmic transformation did indeed reduce the skew, it did not affect the results at all. Moreover, a non-parametric comparison of error rates on device- and task-oriented steps was done as well. A Wilcoxon signed-rank test for repeated measures was done, which did not take into account the flip condition. This revealed a significant effect of type of step, $Z = 3, 11, p < 0.005, r = 0.39$.

There was no significant correlation between error rates on device-oriented and on task-oriented steps, as demonstrated by Kendall's tau: $\tau = 0.24, NS$.

Figure 7.2 shows a further breakdown of the average error rates for each target step. As the figure illustrates, device-oriented error rates are significantly higher than task-oriented ones for three target steps, and show a trend in this direction for another four steps. On two steps, steps 6 (activate body widget (DS) or rip off monster's head (TS)) and 18 (paint monster (TS) or activate programmer widget (DS)), a trend in the opposite direction was observed.

7.2.2.2 Step Times

Step times were also investigated; it was expected that these were longer for device-oriented steps. As in the previous chapter, only correct cases were included. Due to a programming error, the step time on the device-oriented version of step 4 (deactivating the slider) was not accurately recorded; data from this step are excluded. The data was inspected for outliers, because extremely long step times may not reflect the time it takes for a goal to gain sufficient activation. Instead, it is likely that other processes are at work in these cases. The outliers were identified in the raw step time data (before averaging), to prevent having to exclude a participant for a single long step time. An outlier was identified as a data point that was more

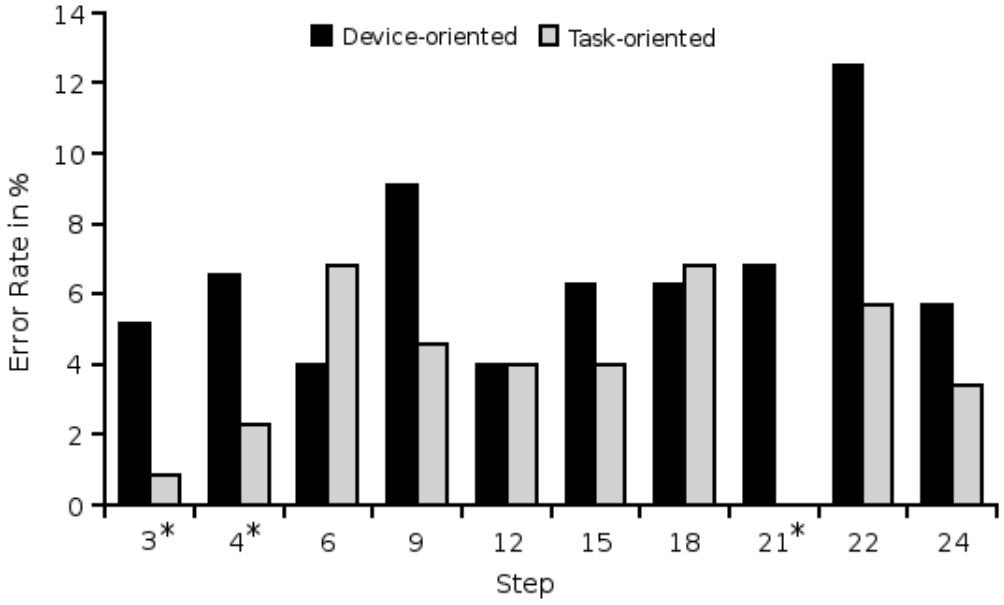


Figure 7.2: *Error rates for each target step. Black bars indicate device-oriented steps, grey ones indicate task-oriented steps. An asterisk (*) indicates that the difference between the device- and task-oriented versions of the step is significant, as indicated by Mann-Whitney U-tests.*

than 3 standard deviations removed from the mean. Four such outliers were found, each with a step time of more than 25 seconds; these were removed from the data set.

The mean step times were then computed for each participant. Table 7.3 shows an overview of the mean step times for device- and task-oriented steps. A 2 x 2 mixed-design ANOVA was done. The type of step was the within-subjects variable, and the condition (flip 1 or flip 2) was the between-subjects variable. This revealed a significant main effect of the type of step, $F(1, 30) = 21.7, p < 0.001, \eta_p^2 = 0.42$, indicating that step times on device-oriented steps were longer than on task-oriented steps. There was no significant main effect of condition, $F(1, 30) = 0.46, NS, \eta_p^2 = 0.015$, and also no significant interaction between the two variables, $F(1, 30) = 0.43, NS, \eta_p^2 = 0.014$.

There was a significant correlation between step times on device- and task-oriented steps within participants, as demonstrated by Kendall's tau: $\tau = 0.54, p < 0.001$.

In addition, figure 7.3 shows a further breakdown for each individual target step. For three out of nine step pairs, step times were longer on the device-oriented step than on the task-oriented one. Steps 6 (activate body widget (DS) or rip off monster's head (TS)), 15 (activate

	Mean Step Time (SD), in msec
Total	3976 (3054)
Device-Oriented	4456 (3384)
Task-Oriented	3520 (2625)

Table 7.3: *Mean step times on device- and task-oriented steps across all participants. Standard deviation is given in brackets.*

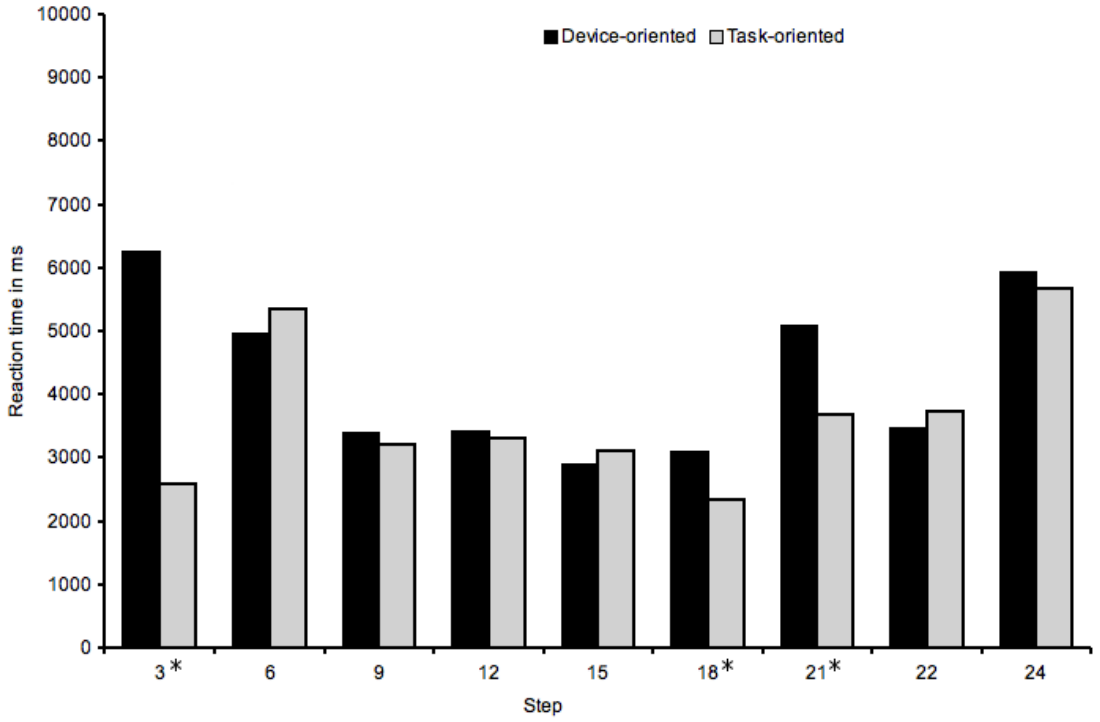


Figure 7.3: *Step times for each target step. Black bars indicate device-oriented steps, grey ones indicate task-oriented steps. An asterisk (*) indicates that the difference between the device- and task-oriented versions of the step is significant, as indicated by t-tests.*

head widget (DS) or rip off monster's body (TS)) and 22 (position laser (TS) or activate laser (DS)) had marginally longer step times on the task-oriented step.

7.2.2.3 Proportion of Omission Errors

It was hypothesised that errors on device-oriented steps were more likely to be omissions than those on task-oriented steps. For each participant, the proportion of omission errors was calculated for all device-oriented steps and for all task-oriented steps. For instance, if a participant made four device-oriented omission errors, and one other device-oriented error, then the proportion of device-oriented omission errors would be $4/(1 + 4) = 0.8$. The mean and median proportions for device- and task-oriented steps are shown in table 7.4. A breakdown of the proportions for each individual step is shown in figure 7.4. It demonstrates that, for four of the steps, the proportion of omission errors is significantly higher on device-oriented steps. However,

Proportion of Omissions	**Mean (SD)**	**Median**	**Number of Omissions (Total Error)**
Total	0.78 (0.41)	0.86	171 (218)
Device-Oriented	0.85 (0.43)	0.84	123 (145)
Task-Oriented	0.66 (0.71)	0.55	48 (73)

Table 7.4: *Proportion of omission errors on device- and task-oriented steps. The second column shows the mean (with standard deviation in brackets), the third column shows the median, and the last shows the total number of omissions and total errors (including omissions).*

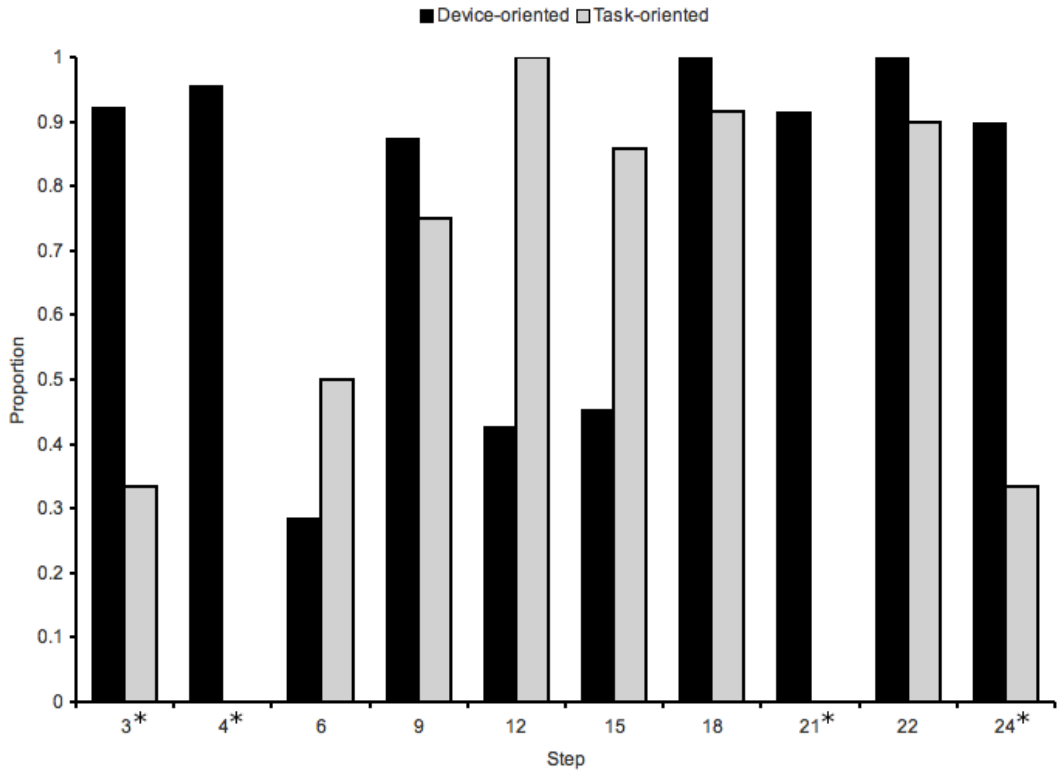


Figure 7.4: *Proportion of omission errors for each target step. Black bars indicate device-oriented steps, grey ones indicate task-oriented steps. An asterisk (*) indicates that the difference between the device- and task-oriented versions of the step is significant, as indicated by Mann-Whitney U-tests.*

the figure also reveals that there are a relatively large number of extremes (i.e. values of 0 or 1), which indicates that the proportions may not be normally distributed.

Unsurprisingly, these proportions were very significantly non-normal, to the point where a mixed-design ANOVA would be highly inappropriate. A Wilcoxon signed-rank test was done on the type of step variable, but this test did not take into account the flip condition. The test revealed that device-oriented steps have proportionally more omission errors than their task-oriented counterparts, $Z = -2.08, p < 0.05, r = -0.26$.

7.2.2.4 Eye Movements

Due to faulty hardware, eye movement data could be collected for 16 participants only. Only data from just before a target step was used; this was defined as the period between the completion of the previous step and the start of the current step. Moreover, only correct cases were included, for the same reason as described in the section on step times. It should be noted that steps 3 and 4 (using the sliders to input the coordinates) were not included in the analysis, because they were executed on a different screen and were as such out of the range of the eye tracker. For each participant, the average fixation length, number of fixations and total fixation duration were calculated, for the remaining device- and task-oriented steps. Table 7.5 shows a summary of these measures.

Eye Movements	AFL (SD), in ms	TFD (SD), in ms	NoF (SD)
Device-Oriented	376 (90)	3788 (727)	10.56 (2.9)
Task-Oriented	372 (71)	3760 (908)	10.56 (3.5)

Table 7.5: *Eye movements on device- and task-oriented steps. AFL indicates the average fixation length, TFD indicates the total fixation duration averaged per step, and NoF indicates the number of fixations per step. Standard deviations are given in brackets.*

Average Fixation Length A mixed-design ANOVA was done with step type as the within-subjects factor and flip as the between-subjects factor. This revealed no significant main effect of the type of step, $F(1,14) = 0.095, NS, \eta_p^2 = 0.007$. There was also no significant main effect of flip, $F(1,14) = 1.99, NS, \eta_p^2 = 0.12$. However, the interaction between these two dimensions was significant, $F(1,14) = 18.9, p < 0.005, \eta_p^2 = 0.57$.

Total Fixation Duration A similar ANOVA was applied to the total fixation duration data. This revealed no significant main effect of the type of step, $F(1,14) = 0.047, NS, \eta_p^2 = 0.003$. There was also no significant main effect of flip, $F(1,14) = 0.04, NS, \eta_p^2 = 0.002$, nor a significant interaction, $F(1,14) = 0.007, NS, \eta_p^2 = 0.000$.

Number of Fixations The same mixed-design ANOVA was applied, and revealed no significant main effect of step type, $F(1,14) = 0.000, NS, \eta_p^2 = 0.000$. There was also no significant main effect of flip, $F(1,14) = 0.90, NS, \eta_p^2 = 0.06$. However, there was a significant interaction between the two dimensions, $F(1,14) = 7.28, p < 0.05, \eta_p^2 = 0.34$.

7.2.3 Discussion

The current study investigated whether the relevance of a step to the task goal affects error rates, step times and type of error. An experiment was done that manipulated the goal relevance of steps, while controlling for other factors, such as the difficulty of steps and differences in the interface.

7.2.3.1 Behavioural Data

It was found that the error rates were significantly higher on device-oriented steps than on task-oriented steps. Moreover, the results showed that step times were longer on device-oriented steps, and there was a larger proportion of omission errors than on task-oriented steps. These findings confirm the three main predictions of the experiment.

Overall, the behavioural findings provide strong support for the hypothesis that device-oriented steps have lower activation levels than task-oriented ones. Lower activation levels make it more likely that a step inadvertently falls below the interference level, and hence is forgotten. Similarly, it takes longer to retrieve such steps, leading to longer step times. A novel prediction addressed in this study was that lower activation levels lead to a larger proportion of omission errors. This was indeed supported by the data. The finding that all three behavioural measurements show the expected effect provides strong support for the main hypothesis.

An unexpected finding was that on step 6 (activate the body widget (DS) or rip off the

head (TS)), error rates were somewhat higher when the step was task-oriented than when it was device-oriented. This is surprising for two reasons. First, step 6 is very similar to steps 9, 12 and 15, all of which had identical subgoal structures. Therefore, the same error pattern would be expected on each of these steps. Since the other three analogous steps *did* show the expected error pattern, it is unlikely that the experimental manipulation did not have its intended effect. Second, step 6 was designed to be similar to the device-initialisation step on the Doughnut task (activating the first widget, the Dough port), as it was the first activation step in the main task interface. On the Doughnut task, this step was relevant only to the device, and had a particularly high error rate (see chapter 5). Therefore, it was expected that the first activation step on the Frankenstein task would also exhibit a high error rate, but the results do not support this. This finding remains somewhat of a mystery.

One possible explanation for this finding could be that in the task-oriented cases, participants did not simply forget to rip off the head, but instead forgot *how* to do it. They may have temporarily forgotten that the head is ripped off by clicking the rip-off button, rather than by clicking on the corresponding head in the widget. The heads in the widget are indeed more visually salient than the button, and could act as a visual cue. In this scenario, the error would be in the incorrect usage of interface, rather than in omitting a device-oriented step altogether. While this explanation risks being biased towards the expected effect, the author's personal experience is consistent with this explanation.

Another step that showed the reverse of the expected error pattern is step 18, on which participants either painted the monster (task-oriented) or activated the programmer widget (device-oriented). It should be noted that the error rate on this step was relatively high on both versions of this step ($\sim$7%). As such, the error rate on the device-oriented version of this step is within the expected range. However, on the task-oriented version, the error rate is higher than expected. It is possible that the step is not as strongly task-oriented as intended. While painting the monster is technically task-relevant, it could be argued that a grey-scale monster is still a monster, since all its body parts are already in place. As such, this step may not make as large a contribution to the main goal as anticipated, but this explanation is not very satisfactory.

An alternative explanation that could account for both these findings is related to the task structure. As discussed in section 2.5.4, errors frequently occur at subtask boundaries. Indeed, both steps 6 and 18 occur immediately after such a boundary. Step 6 is the first step on the 'Galvanizer' subtask, immediately after completing the mapping task, while step 18 is the first step in programming the monster, after completing the subtask of adding all its body parts. Therefore, the higher error rate observed on these steps could, in part, be attributed to it lying on a task boundary.

The results supported the prediction that proportionally more omission errors are made on device-oriented steps than on task-oriented ones. However, the proportional measure used for this purpose is not very refined, as it does not take into account the absolute number of

omissions. For instance, it does not distinguish between cases where 1 out of 1 errors is an omission, or 100 out of 100, as in both cases the proportion of omission errors is 1. This also contributed to the non-normal distribution of the data, which prevented the use of parametric tests. As a result, the flip condition could not be evaluated, since no non-parametric alternative to the mixed-design ANOVA was available. These are issues that are inherent in the type of measure, and cannot be easily resolved. Nevertheless, because a significant effect was found, this result can be accepted with reasonable confidence.

7.2.3.2 Eye Movements

Studies 3 and 4 in chapter 6, using the Spy task, found a significant difference in eye movements between device- and task-oriented steps. This was interpreted as indicating that the cognitive mechanisms underlying these steps may be different. Therefore, it was expected that eye movements would be different on the Frankenstein task as well. However, this effect was not replicated. No differences were found between the two types of steps on any of the three eye movement measures.

The Spy task did not control for variation in the task interface, such as button size and distance between interaction elements. As such, it is possible that the effects found on experiments 3 and 4 were due to this, rather than to the steps' relevance to the task goal. Indeed, the lack of an effect on the Frankenstein task, which *did* control for the interface, undermines the previous explanation. While this does not necessarily mean that there is indeed no difference in the cognitive mechanisms underlying device- and task-oriented steps, it does demonstrate that the eye movements may not be a useful measure in the study of goal relevance. As such, eye movements will not be investigated further. Since they were intended to be exploratory rather than to test a hypothesis, this is not detrimental to the main argument of the work.

In short, the results of the current study provide support for the main hypothesis that device-oriented steps are more problematic than task-oriented steps, because they have lower activation levels. The study addressed three of the four predictions set out in the introduction. The fourth of these predictions, that working memory load has a disproportionate effect on device-oriented steps, will be addressed in the next experiment.

7.3 Experiment 6

This study provides an extension to experiment 5, by looking at the effect of working memory load on device- and task-oriented steps. A similar approach was taken in experiment 2 (see section 5.3 on page 95), which varied working memory load on the Doughnut task. It found that error rates on device-oriented steps were affected much more than those on task-oriented steps. However, this study was not able to investigate the other measures, such as step times. Moreover, the general discussion of the chapter argued that the Doughnut task is problematic for the direct comparison of device- and task-oriented steps.

Therefore, the current experiment uses the Frankenstein task. The previous study demon-

strated that it is highly successful in the study of the effect of steps' relevance to the task goal. The experiment will investigate the differential effect of working memory load on device- and task-oriented steps, by looking at error rates, step times and proportion of omission errors. It is expected that working memory load has a disproportionate effect on device-oriented steps, increasing the number of errors, step times and proportion of omissions.

7.3.1 Method

7.3.1.1 Participants

Fifty-two participants (thirty-three female) took part in the study. They were aged between 18 and 25 years old, with a mean age of 20.3. All participants were students at UCL, and were paid £7 for their time.

7.3.1.2 Materials

The same Frankenstein task as in experiment 5 was used; see section 7.2.1.2 for details. The same secondary task was also used, with an additional manipulation to allow for the variation of working memory load. In the high load condition, participants were asked to repeat the last three digits when the beep sounded, whereas in the low load condition, participants repeated only the last digit. These numbers were determined during a pilot study, which revealed that three digits was the maximum number that participants could reasonably remember without significant deterioration of performance.

7.3.1.3 Design

A mixed design was used. The first independent variable was working memory load. This was varied between participants, and had two levels: high and low. The second independent variable was the type of step, which was varied within participants, and also had two levels: device-oriented and task-oriented. In addition, the dummy variable of task condition was varied between participants, and had two levels: flip 1 (create monsters) and flip 2 (destroy monsters).

7.3.1.4 Measures

As in the previous study, the main measures were error rates, step times and proportion of omission errors.

7.3.1.5 Procedure

The procedure was identical to that in the previous experiment.

7.3.2 Results

The main dependent variables were error rate, step times and type of error. It was expected that working memory load had a greater influence on device-oriented steps than on task-oriented steps, on all three variables. Data from 52 participants was recorded. The average number of training trials was 7.

Two participants failed to execute the secondary task accurately; these were excluded from analysis because their working memory load could not be guaranteed.

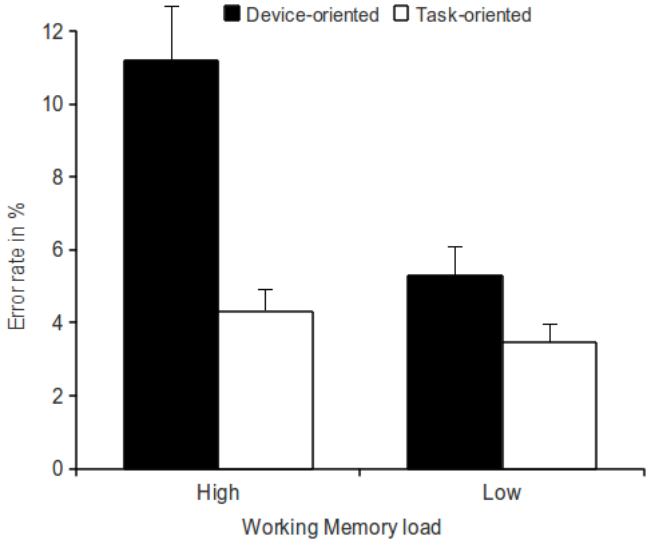


Figure 7.5: *Error rates under high and low load. Black bars indicate device-oriented steps, white ones indicate task-oriented steps. Error bars represent the standard error of the mean.*

7.3.2.1 Error Rates

A total of 385 errors were made during the experiment. Given a total opportunity for error of 50 participants x 11 trials x 12 target steps = 6600, the overall error rate is 5.83%. To simplify the analysis, the flip variable was not analysed separately. Since the previous study demonstrated that there was no effect of flip, this is not expected to influence the results.

Figure 7.5 shows the error rates on device- and task-oriented steps under low and high working memory load. A mixed-design ANOVA was done, with the type of step as the within-subjects variable, and working memory load as the between-subjects variable (though data was not normally distributed). This revealed a significant main effect of the type of step, with device-oriented steps giving rise to higher error rates than task-oriented steps, $F(1,48) = 27.7, p < 0.001, \eta_p^2 = 0.37$. There was also a significant main effect of working memory load, $F(1,48) = 13.2, p < 0.005, \eta_p^2 = 0.22$, with higher error rates under a high load. Moreover, there was a significant interaction between the two dimensions, $F(1,48) = 9.18, p < 0.005, \eta_p^2 = 0.16$. Simple effects analysis showed that in the low load condition, there was no difference between device- and task-oriented steps, $F(1,48) = 2.61, NS$. In the high load condition, on the other hand, there was a significant difference between the two steps, with device-oriented steps giving rise to higher error rates: $F(1,48) = 33.1, p < 0.001$. Conversely, on device-oriented steps, error rates were higher under high load than under low load, $F(1,48) = 13.4, p < 0.005$, while there was no difference between load conditions on task-oriented steps, $F(1,48) = 1.47, NS$. As in experiments 2 and 5, it was determined if transforming the data would improve the moderate skew present in the data. A logarithmic transformation was applied, but did not reduce the skew, nor did it affect the results. These results demonstrate that working memory load has a differential effect on device- and task-oriented steps.

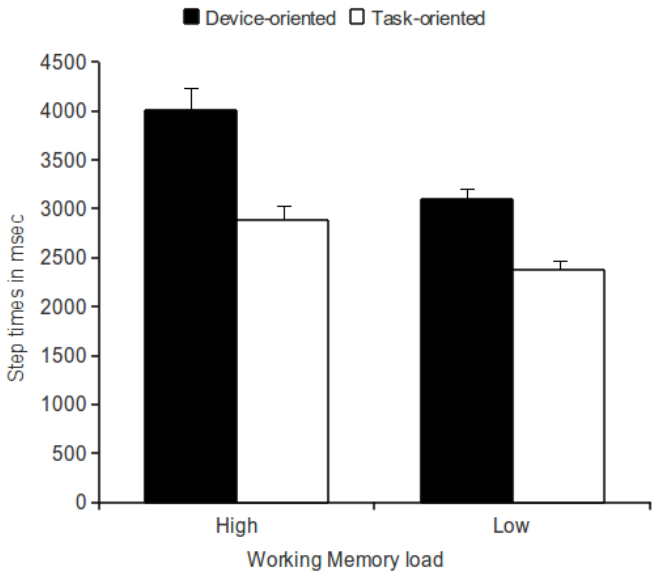


Figure 7.6: *Step times under high and low load. Black bars indicate device-oriented steps, white ones indicate task-oriented steps. Error bars represent the standard error of the mean.*

7.3.2.2 Step Times

Similar effects were found on step times; figure 7.6 shows an overview. A mixed-design ANOVA revealed a significant effect of type of step, $F(1,48) = 97.9, p < 0.001, \eta_p^2 = 0.67$. There was also a significant main effect of working memory load, with longer step times under a high load, $F(1,48) = 17.0, p < 0.001, \eta_p^2 = 0.26$. There was also a significant interaction between the two dimensions, $F(1,48) = 4.74, p < 0.05, \eta_p^2 = 0.09$. Simple effects analysis revealed that step times were significantly longer on device-oriented steps than on task-oriented steps, both in the low load condition ($F(1,48) = 31.0, p < 0.001$) and in the high load condition ($F(1,48) = 70.1, p < 0.001$). Indeed, on device-oriented steps, step times were longer under high load than under low load, $F(1,48) = 15.7, p < 0.001$. The same effect was found on task-oriented steps, $F(1,48) = 10.8, p < 0.005$.

7.3.2.3 Proportion of Omission Errors

Figure 7.7 shows an overview of the proportion of omission errors on device- and task-oriented steps, under high and low working memory load. The data were significantly non-normal, and therefore a mixed-design ANOVA was deemed inappropriate. Instead, a Wilcoxon signed-rank test was done on the type of step, and a Mann-Whitney U test was done on working memory load. These revealed a significant effect of the type of step, with proportionally more omission errors on device-oriented steps, $Z = -3.01, p < 0.005$. On the other hand, no significant effect of working memory load was found, $Z = -0.49, NS$. Unfortunately, these tests were not able to test for interactions between the two variables.

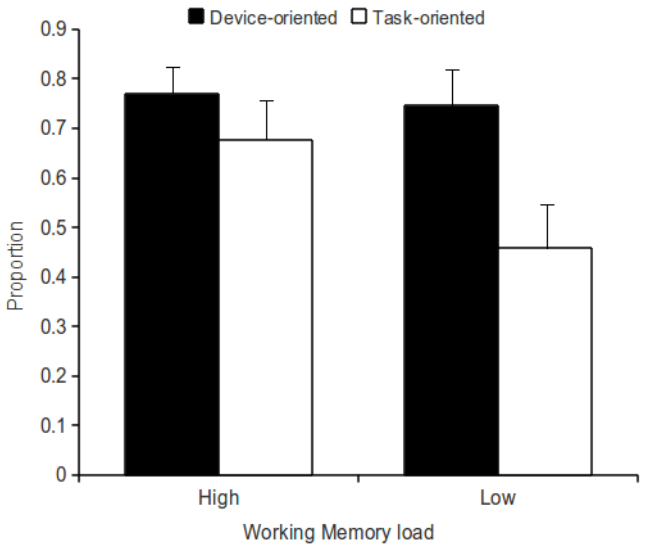


Figure 7.7: *Proportion of omission errors under high and low load. Black bars indicate device-oriented steps, white ones indicate task-oriented steps. Error bars represent the standard error of the mean.*

7.3.3 Discussion

The current study investigated the effect of working memory load on device- and task-oriented steps. It was hypothesised that a high working memory load has a disproportionate effect on device-oriented steps, leading to a larger increase in error rates, step times and proportion of omission errors than on task-oriented steps.

The results supported these predictions. Error rates were higher on device-oriented steps, and an interaction between the type of step and working memory load was also found. This indicates that the load has a disproportionate effect on device-oriented steps. A highly similar pattern was found for the step times, with device-oriented steps taking longer than task-oriented ones, and an interaction between the two dimensions. Again, this indicates that load has an increased effect on device-oriented steps.

The analysis of the proportion of omission errors was problematic, in that the data violated the assumptions of ANOVA to such an extent that the validity of the results would have to be questioned. While ANOVAs are generally reasonably robust to violations of their assumptions (Schmider et al., 2010), there is of course a limit to this. As such, non-parametric tests were used to establish the effects of the type of step and of working memory load. These showed that the proportion of omission errors was larger on device-oriented steps than on task-oriented steps. However, there was no overall effect of working memory load. Since an interaction could not be established, from these results it is not possible to conclude whether working memory load has a disproportionate effect on the omission errors on device- and task-oriented steps.

The monitoring task used to vary working memory load was highly similar to the one used by Byrne and Bovair (1997). They found that a high load (accomplished by asking participants to repeat the last three digits read out to them) significantly increased post-completion error rates, though only in individuals with a relatively low working memory capacity. This is in line

with the findings of the current study, which shows that these results also extend to slip errors beyond the PCE. Byrne and Bovair (1997) explain their findings in terms of the competition between goals in memory. A high working memory load means that there are more goals in memory that compete with the target goal. This makes it more difficult for the target goal to reach the required activation level to be executed. Post-completion errors occur because the PC subgoal no longer receives activation from the main goal (since it has been completed), and as such loses its activation more quickly than other goals. This occasionally leads to an error. In the case of a device-oriented step, the hypothesis of the current work is that it does not receive as much activation as task-oriented steps, and as such is more at risk of being omitted. A higher load on working memory then leads to more competition, making it even more difficult for device-oriented steps to compete with others. Thus, the account of PCEs proposed by Byrne and Bovair (1997) can also account for the current findings, with the exception of the mechanisms that lead to the lower activation levels.

A difference between Byrne and Bovair's (1997) work and the current work is in the low load condition. In the current study, participants had to attend to a single digit. However, participants in Byrne and Bovair's low load condition did not have to attend to (or even listen to) any digits at all. This is a potential shortcoming in their study, as it may have altered participants' approach to the main task. The current study aimed to remedy this by asking participants to attend to the digits, while keeping memory load low by requiring them to simply repeat the last digit. It was found that this approach was highly effective in manipulating participants' working memory load and producing a statistically significant effect.

7.4 Overall Discussion

The current chapter presented two studies that investigated the effect of goal-relevance on a variety of measures. Experiment 5 studied its effects on error rates, step times, and the proportion of omission errors. Experiment 6 assessed the effect of working memory load on these measures.

The main hypothesis of the current work is that steps that have low relevance to the task goal are more problematic than those that make a significant contribution to the goal. The main mechanism proposed for this is that these device-oriented steps have lower activation levels than their task-oriented counterparts. This was predicted to lead to higher error rates, longer step times, proportionally more omission errors and a higher susceptibility to working memory load. The two experiments provided support for all these predictions. As such, they strongly support the main hypothesis.

Nevertheless, the alternative mechanisms for the problems encountered on device-oriented steps must also be considered. These include high activation levels on erroneous steps, and degraded context representations. The experimental results on the Frankenstein task do not support the idea that such problems can be caused by the intrusion of erroneous steps that have gained too much activation. The Frankenstein task was designed to manipulate the goal-

relevance of the target steps, while keeping all other factors constant, including the steps following the target steps. If the *next* step was the primary cause of the higher error rates on device-oriented steps, it would have been expected that the steps within each target step pair on the Frankenstein task would have similar error rates and step times. After all, the steps following the target steps were kept constant. Indeed, this is not what was found: the differences between device- and task-oriented steps were robust. As such, the current experimental findings do not support the alternative hypothesis that the problems on device-oriented steps are caused by high activation levels on an intruding step.

Another hypothesis that needs to be addressed is that the problems are caused by degraded (context) representations. In this hypothesis, the idea is that the mental representations of device-oriented steps are more degraded than those of task-oriented steps. As a result, it is predicted to be more likely that a similar representation captures behaviour, which could lead to an error. While this hypothesis could explain the higher error rates observed on device-oriented steps and why working memory load affects these steps, it is less able to account for why a larger proportion of these errors are omissions. After all, the majority of errors that are made through this mechanism are capture errors (e.g. Cooper and Shallice, 2000). Moreover, this approach cannot account for the differences found in step times.

In short, the results found in the two experiments using the Frankenstein are highly supportive of the hypothesis that device-oriented steps are more problematic because they have lower activation levels than task-oriented steps. Unlike the Doughnut and Spy tasks, the design of the Frankenstein task allowed the other hypotheses to be addressed; these are not supported.

7.4.1 Evaluation of the Frankenstein Task

The results presented in this chapter demonstrate that the Frankenstein task is highly successful in studying the effects of goal relevance of steps. It generates measurable error rates, which many researchers have argued is not easy to achieve in experimental settings (e.g. Byrne and Bovair, 1997). Like the Phaser task as used by Byrne and Bovair (1997) and Chung and Byrne (2008), the Frankenstein task was designed to be highly engaging to reduce the effects of self-monitoring. It was thought that a task such as building monsters, with accompanying visual effects, would be effective in distracting participants from the experimental setting, and thereby reducing self-monitoring. Indeed, many participants spontaneously reported that the Frankenstein experiment was the most entertaining experiment they had ever participated in.

Many other experimental tasks that study error, such as the Phaser task (Byrne and Bovair, 1997) and the Doughnut task (Byrne and Bovair, 1997), generate relatively low error rates when used by themselves, making it more difficult to statistically analyse the results. Instead, methods such as interruptions or a secondary monitoring task are used to increase error rates. Indeed, the Frankenstein task also uses a secondary monitoring task to ensure error rates are at a measurable level. However, it is not entirely clear what the effect of this is on the validity of the

results. In some cases, it can present a difficulty in the interpretation of experimental results, such as the use of interruptions in Ratwani et al.'s (2008) prediction paradigm. They found that post-completion errors can be detected *before* they are made, by looking at participants' eye movements. However, because most PCEs occurred after an interruption, it is possible that their findings represent the recovery from the interruption, rather than the imminent error. Moreover, there is some evidence that different types of secondary tasks, perhaps even different types of working memory load, may affect performance differently. As such, an ideal task generates high enough error rates by itself, without the need for additional manipulations. The results of experiment 6 demonstrate that error rates in the low load condition are, in fact, substantial enough to allow statistical analysis. As such, it would be beneficial to determine whether the Frankenstein task can also be used by itself, without the secondary load task.

A drawback of the Frankenstein task is its relatively complex experimental design. In its most basic version, it has two experimental variables: the type of step (device- or task-oriented) and the task condition (make or destroy monsters). Therefore, any further manipulations will result in a highly complex design. This makes it potentially more difficult to analyse the data. Moreover, the Frankenstein task is designed to measure error rates, and these are almost never normally distributed. As such, it is preferable to use non-parametric tests to analyse the data. However, there appear to be no non-parametric alternatives for tests such as a 2-way mixed design ANOVA, which is required to analyse the most basic results on the Frankenstein task to the full extent.

A related issue is that the target steps on the Frankenstein task are paired, so each device-oriented step is accompanied by a task-oriented step that uses the same interaction element. It would be preferable to analyse the data while accounting for the pairing of steps, as this increases the sensitivity to within-pair changes. However, there appears to be no easy way to do this.

One option would be to adopt a within-participants design, asking participants to do both the create and destroy conditions. However, this approach is deeply problematic, because it is not clear how this affects participants' mental models of the task. When learning one condition of the task, participants will develop an understanding of which steps in the task sequence contribute to their main goal, and which do not. When the goal of the task subsequently changes to the other condition, while the task interface remains essentially the same, this old knowledge of the steps' relevance to the task goal may interfere with the new task goal. As such, it is likely that the goal relevance of steps in this second condition will be compromised, invalidating the experimental manipulation.

The approach the current chapter has adopted is to average over the different device- and task-oriented steps, so that each participant contributes one device-oriented measure and one task-oriented measure. While this may not be ideal, the data showed that this was effective in generating significant results. Moreover, the results showed that the flip condition has no

effect on the results, and as such it may be beneficial to exclude it from analysis to reduce the complexity of the design. This would also make it possible to use non-parametric tests, which in turn increase the validity of the results.

7.5 Conclusion

The newly developed Frankenstein task was introduced in this chapter. Unlike the Doughnut and Spy tasks, it accounts for interface factors while directly manipulating the goal relevance of steps. Two experiments that use the Frankenstein task were presented.

The results are in line with expectations. The first experiment demonstrated that error rates are higher, step times are longer and a larger proportion of errors are omissions on device-oriented steps. The second experiment showed that working memory load affected device-oriented steps more severely than their task-oriented counterparts. These results support the hypothesis that steps with low relevance to the task goal have lower activation levels than steps that do make a direct contribution to the user's goal. Moreover, the design of the task allowed the examination of the alternative hypotheses of higher activation levels on incorrect steps, and degraded context representations; these were not supported.

The results show that the Frankenstein task is highly successful in studying the role of goal-relevance in the occurrence of slip errors. While it presents some analytical difficulties, the findings show that the task can generate significant results. Moreover, the design allows the direct investigation of the role of goal relevance in the occurrence of slip errors in routine procedural tasks.

Take-home message:

The Frankenstein task manipulates the relevance of steps to the task goal, while controlling for the task interface and step difficulties. Two experiments using this task demonstrated that error rates are higher, step times are longer and proportionally more omission errors are made on device-oriented steps. Moreover, working memory load has a disproportionate effect on device-oriented steps, supporting the hypothesis that these steps have lower activation levels than their task-oriented counterparts.

Chapter 8

General Discussion and Conclusion

In this chapter:

- The experimental findings are discussed in detail, and the implications for existing research are analysed.
- The limitations of the studies are considered, and suggestions for future research are proposed.
- The contribution of the current work is discussed, and conclusions are drawn.

8.1 Overview

Human task performance is inherently fallible. Errors occur in all aspects of life, and are committed by experts and novices alike. While most may simply be annoying, occasionally they can have disastrous consequences. As such, recent research on human error has focussed on understanding errors and developing strategies to reduce their occurrence or deleterious effects.

There are many factors that contribute to the occurrence of errors. Many of the earlier studies have focussed on situational factors, such as interruptions (Li et al., 2006) or working memory load (Byrne and Bovair, 1997). However, these environmental influences are not always able to explain why certain steps in a task are more prone to errors than others. Tsai and Byrne (2007) argued that sequential tasks often have a certain error pattern, which shows how error prone each of the steps is. Indeed, such a robust error pattern was observed on the Doughnut task (e.g Li et al., 2008; Chan, 2008; Hiltz et al., 2010). A number of recent studies have focussed on factors inherent to the task that could explain why certain steps more frequently result in errors than others, for instance a post-completion task structure (Byrne and Davis, 2006). Such factors typically affect one (or more) specific action(s) in the task, leading to a particular error pattern for that task. This is also the approach taken by the current work: the overall aim is to identify such task-inherent factors that systematically increase error rates on some steps, so that these can be designed out of devices.

The main hypothesis tested in the current work is that the relevance of an action to the

main goal plays an important role in how error prone it is. The hypothesised mechanism for this is that steps that are not relevant to the task goal have lower activation levels than those that are. This hypothesis was tested through a series of experimental studies, that focussed on four predictions. If activation levels on device-oriented steps are indeed lower, then it was expected that:

- Error rates are higher
- Step times are longer
- A larger proportion of errors are omissions
- Working memory load has a greater influence

Throughout the experimental studies, all four predictions were confirmed. The next sections discuss in detail the findings and their implications for the existing literature.

8.2 Experimental Findings

8.2.1 Error Rates

The first major behavioural finding was that error rates were higher on device-oriented steps than on task-oriented ones. This finding was robust across all experiments and all tasks. The Doughnut and Spy tasks demonstrate that, in a large variety of steps, the ones that were marked in the classification study as device-oriented had higher error rates than those that were marked as task-oriented. Moreover, the Frankenstein task demonstrated that when steps' relevance to the main goal was directly manipulated, those with low relevance had higher error rates than those that were directly relevant to the task goal. These findings provide strong support for the main claim of this thesis that device-oriented steps are more prone to errors than their task-oriented counterparts.

An alternative explanation for the higher error rates observed on device-oriented steps is that, rather than being an effect of being less relevant to the task goal, these steps are less well-learned. It seems reasonable to argue that steps that are not as well learned will more frequently lead to errors. In principle, it is quite likely that this applies to many device-oriented steps, since they generally occur on a single device only. However, this explanation is not satisfactory, for two reasons. First, it is highly unlikely that the participants in the current studies had made doughnuts or monsters before, or had any experience flying a plane. As such, participants were equally experienced in each step. Second, the Frankenstein task consists of two equally well-learned procedures, while the goal relevance of the steps was manipulated. Yet the difference in error rates persisted. As such, this demonstrates that goal relevance is a key factor in determining error rates, rather than how well-learned each step is. This does not rule out that this factor plays a role in real-world tasks, but it is not the main cause of the differences found between device- and task-oriented steps in the experiments.

While an approach from goal relevance is able to account for a fair amount of variation in the error rates observed on the three experimental tasks, the error diagrams show that there is also variation between different device-oriented steps and between different task-oriented steps. Unsurprisingly, this means that, besides goal relevance, other factors also play a role in determining performance. Identifying these factors provides a promising direction for future research. In particular, future work can investigate the particularly high error rate on the first selector step on the Doughnut task. It is a long-standing question in the 'Doughnut task community' why this step is so very error prone, while the other selector steps result in more moderate error rates.

8.2.2 Step Times

Step times were found to be longer on device-oriented than on task-oriented steps. Again, this was consistent throughout the four experiments that tested this, on two different tasks.

A limitation of the Spy task is that it does not control for differences in the task interface, such as button size and distance between interaction elements. As such, it is possible that the differences in step times found on this task are a result of these interface factors, rather than the goal relevance of the steps. The analysis of the step time data attempted to take these factors into account by checking for correlations between step times and distances, which were not found. However, this is not a very powerful approach, and as such the effect of distance cannot be completely eliminated.

The Frankenstein task, on the other hand, did account for these interface factors. Distances and button sizes were kept constant for each device- and task-oriented step pair. This allowed the direct comparison of step times. Indeed, it was found that these are longer for device-oriented steps than for task-oriented steps, on both experiments. This demonstrates that the difference in step times is indeed robust.

8.2.3 Type of Error

A novel prediction addressed in this thesis is that device and task-oriented steps differ in the type of errors they give rise to. It was hypothesised that device-oriented errors are more frequently omission errors, as opposed to sequence errors or perseveration errors, for instance. Indeed, the data were supportive of this. The two studies on the Frankenstein task found that the proportion of omission errors was higher on steps with low task relevance.

The type of errors that occur at a particular step in a task sequence is a topic that has not frequently been addressed in the experimental literature. Interestingly, most accounts of post-completion errors suggest that the majority of these slips should be omissions. However, few of the experimental studies on the PCE have addressed this prediction. Instead, any type of error that occurs at the PC step is generally considered to be a PCE.

One study that did address the different types of errors on a particular step comes from Li et al. (2008). They discussed two different types of error that occurred on certain device-oriented steps: an omission error, involving the simple omission of the step, and a sequence

error, in which another step was executed, such as one belonging to a different subgoal. They found that omission errors were by far the most frequent errors at this step. Unfortunately, Li et al. (2008) presented no data on the type of errors made on task-oriented steps, so it is not possible to compare these. Nevertheless, the current findings are in line with the large proportion of omission errors found by Li et al. (2008).

A number of theoretical models of routine procedural action do make predictions about the type of error that should occur on particular steps. Trafton et al. (2011), for instance, argue that, in a routine procedural task, perseveration errors (in which a previous step is repeated) will be more frequent than omission or anticipation errors after an interruption. They argue that, because activation levels decay gradually, the previous step will have the next-highest activation level. Because of noise, occasionally this step may be retrieved instead of the current, correct step, resulting in a perseveration error. Omission errors, on the other hand, occur when after an interruption, the previous subgoal is retrieved, but the system erroneously assumes that it has already been executed. Interestingly, the current work did not use interruptions, yet the omission error rate was much higher than the perseveration error rate. These findings are not in line with Trafton et al.'s (2011) predictions.

8.2.4 Working Memory Load

Working memory load was found to affect device-oriented steps more severely than task-oriented ones. This was apparent both in the increased error rates and longer step times on device-oriented steps under high load. Such an effect of load on step times was much smaller on task-oriented steps, while it was absent for error rates. As such, the current work demonstrates that, when working memory load is increased, those steps that do not contribute to the task goal suffer most.

This finding is in line with previous work by Byrne and Bovair (1997), who found that working memory load increases the occurrence of the PCE. However, the current work goes beyond this in two ways. First, it not only looks at the effect on error rates, but also on step times. Second, the current work goes beyond the post-completion step and also looks at how different steps are differentially affected. It demonstrates that the effect found by Byrne and Bovair (1997) is not specific to PCEs, but also applies to device-oriented steps in general.

Byrne and Bovair (1997) divided their participants into two groups according to working memory capacity. They found that the effect held only in low-capacity individuals, while no effect was found for high-capacity individuals. They argued that in the latter, their working memory capacity was not reached by the task, and as such less competition between memory items was taking place. In the current thesis, however, no such grouping was made based on capacity. Nevertheless, significant effects of working memory load were found, both in experiment 2 (using the Doughnut task, see section 5.3) and in experiment 6 (using the Frankenstein task, see section 7.3). As such, the current work showed that for device- and task-oriented steps, it is not always necessary to divide participants into low and high capacity groups in order to

produce significant effects.

These findings have implications for the design of devices that are used in high workload environments, such as busy hospital wards or airplane cockpits. In these situations, it is particularly important that device-oriented steps are designed out of devices as much as possible, as the experimental results have shown that under high load the problems observed on these steps increase.

8.2.5 Correlation Between Device- and Task-Oriented Steps

The experimental results showed an interesting pattern in terms of the correlation of performance on device- and task-oriented steps. Error rates on the two types of steps were not correlated within participants. This means that if a participant makes many errors on device-oriented steps, this does not necessarily mean that the same will be observed on task-oriented steps (and vice versa). Interestingly, the step times *were* correlated between different types of steps: if a participant took a relatively long time to execute device-oriented steps, it was likely that they would also take a long time to execute task-oriented steps (and vice versa).

The lack of a correlation between device- and task-oriented error rates could indicate that they have different underlying mechanisms. As discussed above, errors on device-oriented steps may be qualitatively different from those on task-oriented steps. The experimental studies found that the proportion of omission errors is higher on device-oriented steps, and argued that they arise from different mechanisms. In this light, it is not surprising that no correlation between the error rates on the different types of steps was found.

Step times, on the other hand, *were* found to correlate between device- and task-oriented steps. It is thought that the time taken to execute the next step is closely related to the activation level on the step: the lower the activation level, the longer it takes for a step to be retrieved (e.g. Altmann and Trafton, 2002). While these activation levels are thought to differ for device- and task-oriented steps, the mechanism through which they affect step times is thought to be the same in both types of steps. This means that they are expected to correlate within participants, which is indeed supported by the current findings.

The finding that error rates on device- and task-oriented steps are not correlated within participants also has a potential practical implication. It indicates that a person who is highly skilled in a particular task, and makes very few errors while executing it, may not necessarily perform as well when using a different device to execute the task. This further highlights the argument that device-oriented steps should be designed out of devices as much as possible.

8.2.6 Eye Movements

The studies using the Spy task (see chapter 6) found a difference in the eye movements on device-oriented and task-oriented steps. The results showed that the fixation lengths were shorter and the number of fixations was larger on device-oriented steps. This is an extremely interesting finding, because different eye movement patterns are often associated with different cognitive processes (e.g. Epelboim et al., 1995; Henderson et al., 1999; Hayhoe and Ballard, 2005). It

was hypothesised that, since device-oriented steps are more difficult to remember, participants may have spent more time searching the interface on these steps. This is consistent with earlier findings that searching behaviour leads to shorter fixation lengths than for instance viewing behaviour (Henderson, 2003).

However, this effect was not replicated on the Frankenstein task: no differences were found in any of the measures. As such, the results of the Spy and Frankenstein experiments are inconsistent. It can be argued that the Frankenstein task better controls for factors in the interface, by keeping it the same while manipulating the goal relevance of steps. As several lines of work have shown (e.g. Henderson et al., 1999; Hayhoe and Ballard, 2005), even small changes in the visual task interface can make a difference to eye movements. It is possible that the findings on the Spy task can be explained by such interface factors, since they were not controlled for (even though the interface was designed to keep these effects to a minimum). Since the Frankenstein task *does* account for these, the results on this task are likely to be more reliable.

Nevertheless, the lack of an effect on the number of fixations on the Frankenstein task is somewhat surprising. Device-oriented steps are thought to be more difficult to remember, and indeed the results showed that they take longer to execute. As such, this was expected to also lead to more fixations, but this is not what was found.

In short, the eye movement results on the Spy and Frankenstein tasks are inconsistent. However, since eye movements are sensitive to the visual structure of the task interface, and the Frankenstein task controlled for this, it is thought to be more likely that eye movements are, in fact, not different as a result of the goal relevance of steps. However, since this was an exploratory measure, these inconclusive findings do not negatively affect the validity of the other results.

8.3 Activation Levels

A major hypothesis addressed in this thesis is that the problems associated with device-oriented steps are caused by low activation levels. While activation is a theoretical construct used in various cognitive models (e.g. Anderson and Lebiere, 1998; Cooper and Shallice, 2000), it can in fact be argued to also be relevant to the applied domain. Jonides et al. (2008) argued that activation is 'real' in the sense that memory representations are 'active' in the brain when they are in working memory. Similarly, Hiltz et al. (2010) argued that the notion of cognitive salience directly represents how active a memory representation is: steps with less active memory representations (i.e. lower activation) will have lower cognitive salience and thus be more difficult to remember.

As discussed in the previous section, the main findings of the current work are that, on device-oriented steps:

- Error rates are higher

- Step times are longer
- A larger proportion of errors are omissions
- Working memory load has a greater influence

These are all in line with the experimental predictions. This section discusses the implications of this in relation to the activation level hypothesis and the proposed alternative mechanisms for error. It then discusses how the various cognitive models can account for these findings, and if and how they are able to represent device- and task-oriented steps.

8.3.1 Error Mechanisms

Three major mechanisms for the occurrence of errors were identified in the discussion of the cognitive models in chapter 2 (see section 2.6). Table 8.1 gives an overview of the major predictions and how these relate to the experimental findings.

8.3.1.1 Low Activation Levels

Lower activation levels are thought to be the mechanism that best explains the findings. First, the error rates were consistently found to be higher on device-oriented steps. This can be explained by the lower activation levels: if these are not high enough to compete with the next most active goal or schema in memory, then an error is made. The occasional occurrence of slips can be explained by the noise that is applied to the system. Under normal circumstances, the activation level on device-oriented steps will be just high enough to compete with other goals. However, the noise can occasionally cause it to dip below the level of the next most active goal, leading to an error. Note that the noise must be much greater on task-oriented steps in order for these steps' activation to dip below the interference level. As a result, their error rates are much lower.

Similarly, this mechanism predicts that the proportion of omission errors is higher on device- than on task-oriented steps. If a target step's activation level occasionally dips below that of

Error Mechanism	**Prediction**			
	Error Rates	**Step Times**	**Proportion of Omissions**	**Working Memory Load**
Lower Activation Levels	Device > Task	Device > Task	Device > Task	Device > Task
Higher Activation Levels on Incorrect Steps	Device > Task	Device > Task	Unclear	Device > Task
Degraded Context Representation	Device > Task	Unclear	Mostly Capture Errors	Device > Task
Finding	*Device > Task*	*Device > Task*	*Device > Task*	*Device > Task*

Table 8.1: *Overview of the predictions of the three different mechanisms for errors and how they relate to the experimental findings (last row). Note that 'Device > Task' indicates that, for the particular measure, the effect is larger on device- than on task-oriented steps. For instance, in the 'Error Rates' column it means that error rates are higher on device- than on task-oriented steps, etc.*

a competitor, the competitor may 'win'. In this case, the target step is omitted. In contrast to device-oriented steps, task-oriented steps lead to relatively few omission errors. Their higher activation levels mean that they are more robust to errors, and as such it often takes more for these steps to go wrong. Random noise may not be sufficient to push them below the activation level of a competitor. Instead, a different step in the task sequence (not specifically the next one) may gain too much activation, leading to a capture error or a perseveration error, for instance.

The step times were also found to be consistently longer on steps that do not contribute to the task goal. This is in line with the idea that steps with low relevance to the task goal are more difficult to remember: it takes longer to retrieve such steps from memory. Lower activation levels on these steps are able to explain these findings. In the M-f-G model, if a goal or chunk has a low activation level, it takes longer to be retrieved (Altmann and Trafton, 2002). Moreover, activation accumulates over time, and the rate at which this happens may be slower for device-oriented steps than for task-oriented ones.

The last prediction that this work addresses is that working memory load has a greater influence on device- than on task-oriented steps. In the IAN model (Cooper and Shallice, 2000, 2006), load can be seen as the number of competing schemas in memory: the more of these, the higher the load, and the more difficult it is for each to compete. If a schema is already less active, because it is less relevant to the task goal, then the high load will make it even more difficult for it to compete. Task-oriented steps, on the other hand, may still be relatively capable of competing with other schemas. The M-f-G model (Altmann and Trafton, 2002, 2007) makes similar predictions. While the ACT-R architecture on which it is based does not have an explicit working memory, a higher load can be represented by a larger number of chunks competing with each other for a limited number of retrieval cycles, resulting in similar effects. The current experimental work found that this effect was present in both the error rates and the step times.

In short, the experimental findings of this work are highly supportive of the low-activation level mechanism. Moreover, this mechanism is supported by a wide variety of previous work. For instance, Rieman et al. (1994) argued that steps that do not do any 'real work' have lower activation levels. Hiltz et al. (2010) make a similar argument, claiming that steps that have low cognitive salience because they do not contribute to the main goal can be represented as having low activation levels. Moreover, Byrne and Bovair (1997) argued that the post-completion error is caused by low activation levels on the PC step. Therefore, low activation levels can be seen as a highly plausible mechanism for the problems observed on device-oriented steps.

8.3.1.2 High Activation Levels on Incorrect Steps

Back et al. (2007) argued that a frequently observed slip error in their experimental task is caused by another step 'springing to mind' much more strongly than the correct step. Indeed, an alternative mechanism that could explain frequent errors such as those observed on the experimental tasks is that incorrect steps gain too much activation. This mechanism works in much the same way as the previous one, leading to higher error rates, longer step times,

and an increased effect of working memory load on steps with more moderate activation levels. However, in this case, the fault lies with the intruding step, rather than the currently correct step.

This approach is in line with some of the experimental findings. For instance, most of the error-prone steps on the Doughnut task are followed by a highly task-relevant step. It is possible that these task-oriented steps capture attention away from the device-oriented steps, resulting in the experimental findings. However, this explanation is problematic for other tasks. After all, there is no reason to believe that such intruding steps occur preferentially on device-oriented steps, especially if the activation levels on device-oriented steps are unaltered. Moreover, this is not in line with the argument put forward by Kirschenbaum et al. (1996), who claimed that several tool-oriented steps in a row would be more problematic than when they are evenly distributed across the task. In addition, the experiments using the Frankenstein task (see chapter 7) also demonstrated that this explanation is not satisfactory. Most of the target steps on this task were followed by a non-target step which was held constant. As such, if the next step was the cause of the errors, it would be expected that manipulating the goal relevance of the target step would have no effect. The experimental data showed that the differences between device- and task-oriented steps remained. Thus, it can be concluded that a high activation level on incorrect steps is not the main mechanism for the problems observed on device-oriented steps.

It was also found that the proportion of omission errors was higher on device- than on task-oriented steps. An explanation from higher activation levels on incorrect steps is problematic for this finding. Since the error is caused by the highly active step, it is in essence a capture error. If the next step in the task sequence is the capturing one, it will essentially result in the omission of the device-oriented step. Alternatively, if the capturing step is another step altogether, the manifestation of the error becomes unpredictable.

It should be noted that, in real world tasks, both of the error mechanisms described above can operate at the same time, increasing the effects of both. Indeed, this can happen when a device-oriented step is followed directly by a highly salient task-oriented step. In this scenario, the device-oriented step will have a relatively low activation level, while the task-oriented step will have a relatively high activation level. Thus, the difference in the activation levels is larger than normally expected, and this combination of steps may be even more prone to errors.

8.3.1.3 Degraded Context Representations

The third approach discussed in this thesis is that errors occur because the patterns representing actions are degraded. This approach is qualitatively different from the others, as it is based on a modelling approach that is not regulated by the *amount* of activation per se, but rather the *pattern* of activation. Indeed, this approach can explain why error rates are higher on device-oriented steps, and why working memory load affects them more than their task-oriented counterparts.

Errors occur in this model when these representations degrade, or become less clear. When this happens, they may become more similar to related representations, and lapse into these inappropriate patterns. Thus, the more degraded a pattern, the more error-prone it is. This is in line with the current findings, assuming that device-oriented steps are represented by patterns that are more degraded than those representing task-oriented steps (note that the reason why they could be more degraded is discussed in the next section on how the cognitive models account for the findings). Since working memory load can be represented by further degrading the patterns, this can explain why a high load loads to a further increase of error rates on device-oriented steps.

Using this error mechanism, the majority of errors are capture errors, in which a similar, but incorrect context takes over from the current context. In principle, this could also lead to omission errors, if the capturing context is relevant to the *next* step. However, it is not clear why these would be any more likely than 'other' capture errors. Moreover, it is important to keep in mind that the SRN model (Botvinick and Plaut, 2004, 2009) that underlies this error mechanism does not generally produce erroneous sequences that were not previously learned.

Another issue that affects this error mechanism is that it does not make explicit predictions about the time it takes to execute steps. While this may merely be a reflection of the relative novelty of the SRN model, it does limit the explanatory power the model has.

In short, it can be concluded that the lower activation level mechanism is the most successful in explaining the experimental findings, as well as the theoretical underpinnings of the role of goal relevance in performance. A further point to note is that the continuous nature of activation levels nicely represents the non-binary nature of device- and task-oriented steps. While the experimental part of this thesis is mostly concerned with steps that are at either end of the spectrum, there are also steps that are not as strongly device- or task-oriented, but instead make only a modest contribution to the task goal, for instance. An approach from activation levels allows the full spectrum to be represented. The next section is concerned with how the cognitive models could represent the relevance of steps to the task goal and how this could lead to low activation levels.

8.3.2 Implications for Cognitive Models

The findings of the current thesis strongly suggest that the problems associated with device-oriented steps are caused by lower activation levels. A question that naturally follows from that is *why* the activation levels are lower. This was discussed extensively in chapter 3, which argued that steps that are and are not relevant to the task goal are represented in a different manner. However, it was not discussed how this may be represented in the cognitive models considered in this thesis. This section investigates how the M-f-G model, the IAN model and the SRN model could account for the lower activation levels on device-oriented steps.

8.3.2.1 Mental Models

Chapter 3 discussed a number of previous studies that strongly build on users' mental models (e.g. Young, 1981; Payne et al., 1990). A user's task model can be thought of as a mental representation of the task, and what is necessary to achieve the main task goal. The user's device model, on the other hand, can be thought of as a mental representation of how the device works, and what is necessary to achieve the task using that device. In many cases, there is considerable overlap between the two, which results in an easily usable system and good performance, as Moran (1983) and Payne et al. (1990) have argued. However, when there is a discrepancy between the two models, for instance when there are certain steps that only occur in the device model, performance may be affected. These steps solely contribute to the operation of the device rather than to the achievement of the main goal: they are device-oriented steps. The discrepancy between the two user models forms the basis of this theoretical account.

In terms of the cognitive models, this could be simulated by providing them with a 'mental model' of how to use the device and one of how to do the task. An important underlying idea is that the mental models can provide activation for the steps in the procedure. The task model provides activation for all the steps that provide a direct contribution towards the goal. The device model, on the other hand, provides activation for all steps necessary to operate the device. This means that those steps that are represented only in the device model (the device-oriented steps) will have lower activation levels than those that are represented in both models, since they receive activation from only a single source.

The next question is then how this can be instantiated in the various models of routine procedural tasks. The M-f-G model is based on the ACT-R cognitive architecture, and as such can easily be provided with a device and a task model. The mechanism of spreading activation can then provide activation for each of the goals: if a goal is represented in both device- and task models, it will receive more activation through this mechanism than if it were represented only in the device model. As a result, task-oriented steps are expected to be retrieved more accurately and also faster. This mechanism seems capable of explaining the experimental results, and, importantly, contains a notion of whether or not each step is relevant to the task goal. Nevertheless, it appears that to date, such a model has not yet been developed, so a direction for future research can be to develop a model of one of the experimental tasks and provide it with a device model and a task model.

The IAN model does not seem to allow the same flexibility. Schemas and goals in this model are represented in a hierarchy, with each schema receiving activation from its parent schema. However, there appears to be no possibility to provide the model with different types of knowledge, or to represent the different schemas in a different manner. Similarly, there seems to be little flexibility in the hierarchy in terms of steps' contribution to the task goal. As such, this approach may not be compatible with the IAN model.

8.3.2.2 Cognitive Salience and Goal Relevance

The concept of cognitive salience is argued to be closely related to the more theoretical construct of activation levels: steps with low cognitive salience will have lower activation levels than steps with high cognitive salience, which will have higher activation levels (e.g. Hiltz et al., 2010). While cognitive salience is an abstract and not a well-defined concept, there are several ways in which it could be implemented in the models.

In the ACT-R architecture underlying the M-f-G model, a reward can be associated with the firing of production rules, such as the successful retrieval of the next goal in the task sequence. In behavioural terms, the reward can represent the goal's contribution towards the main goal. As such, goals that make a larger contribution can be associated with a larger reward than those that do not move the model closer towards the goal. Indeed, if a larger reward is associated with a production rule, it is more likely to be selected for firing, and a smaller reward makes this relatively less likely. This, in turn, affects the frequency of errors on the step that the production executes. While this appears to be a simple and elegant mechanism within the architecture that can simulate goal relevance, it should be noted that this approach largely bypasses activation levels, as it operates mainly on production rules.

The IAN model does not have such a reward mechanism, and must represent cognitive salience in a different manner. Behaviour is governed through a number of different sources of activation, and the balance between these is regulated by a number of parameters. Cooper and Shallice (2000) argue that an imbalance in these parameters can lead to a variety of errors. For instance, if the influence of the external world is relatively large, then this can lead to capture errors. Similarly, too little self-activation can lead to omission errors. This could be a useful approach in accounting for device- and task-oriented steps. Schemas representing device-oriented steps could have a lower self-influence than schemas representing task-oriented steps. This would lead to lower activation levels on the device-oriented schemas, making it more difficult for them to compete. However, this approach may be problematic, because it would require these parameters to be different for different schemas.

Another approach in representing goal relevance in the IAN model is through crux and enabling/tidying actions (see chapter 3), which could be seen as being related to task- and device-oriented steps, respectively. However, these do not seem to be represented differentially in the task hierarchy. Instead, they are mediated through pre- and post-conditions, allowing enabling and tidying actions to be omitted where needed, to increase efficiency. Nevertheless, in light of the current findings, this may provide an interesting avenue for future research in the IAN model.

In short, the previous two sections discussed a number of different approaches to modelling the concepts of device- and task-oriented steps and goal-relevance. There are a number of different ways in which this can be instantiated, and this provides a valuable and interesting avenue for future research. The discussion above did not consider the SRN model, because it

represents the task in a fundamentally different manner. As such, the next section discusses how the SRN model can account for the experimental results and explain the effect of goal relevance.

8.3.2.3 SRN Model

The main mechanism for errors in the SRN model (Botvinick and Plaut, 2004) is through degraded context representations. In relation to device- and task-oriented steps, these can be instantiated by the representations of device-oriented steps being more degraded than those of task-oriented ones. However, there appears to be no obvious reason why the representations would be more degraded for one step than for another. The steps are not represented hierarchically, or in an otherwise meaningful way: there appears to be nothing that can set apart device- and task-oriented steps.

Another issue is that there is no explicit representation of goals in the SRN model. While Botvinick and Plaut (2004) do not deny the importance of goals in cognition, the model is not driven by goals to the same extent as the M-f-G or IAN models. Indeed, as Cooper and Shallice (2006) have argued, goals "play no role in the functioning of the SRN model". They also argue that goals are necessary for flexible behaviour, and as such the SRN model breaks down when it encounters altered or novel situations. Crucially, in the current work, this means that the notion of goal-relevance is incompatible with the SRN model. Moreover, it means that there is no main goal to provide context or activation for its constituent steps.

Interestingly, the model does provide a mechanism for why steps that are specific to a particular device could lead to more frequent errors. The model is greatly dependent on training, or previous experience. As Cooper and Shallice (2006) showed in their reimplementation of the SRN model, it makes omission errors of the kind A → (B) → C (with (B) being the omitted step) only if the model was also trained on a sequence A → C. In relation to device-specific errors, the model may first be trained on a device that uses sequence A → C. When it then uses another device to execute the task, and this device requires step B between steps A and C, step B may sometimes be forgotten. In this situation, the pattern for step A on the first device is highly similar to the pattern for step A on the second device. As such, when noise is introduced to the system, the pattern on step A may drift and occasionally lapse into the sequence of the other device. However, as discussed in chapter 3, this view of the origin of device- and task-oriented steps is not very well supported.

In short, there appear to be a number of ways in which the different cognitive models could account for the occurrence of device- and task-oriented steps in routine procedures. However, to date, there has been no work that has successfully modelled these different types of steps, or has been able to account for similar findings. It should be noted that these are not likely to be fundamental limitations of the models. Rather, it may simply be the case that this is not yet implemented. Indeed, the tasks typically used to illustrate the workings of the models (such

as making tea for the IAN and SRN models) do not contain any device-oriented steps as such, and therefore there is no associated error pattern to explain. Nevertheless, it will be extremely interesting to deploy these models to execute for instance the Frankenstein task, and determine whether they are able to generate similar error patterns and related behavioural observations.

8.4 Limitations

8.4.1 Limited Duration and Routine Procedural Performance

The current work is concerned with the execution of routine procedural tasks. This implies that the person doing such a task is well-trained in its execution, and has a high level of competence. Execution is usually fairly automatic. Anderson (1995) distinguished between three stages of skill acquisition. In the first stage, people acquire the declarative knowledge needed to execute the task. In the second stage, rules are formed from the declarative knowledge. In the third stage, these rules can be applied more automatically, and fewer cognitive resources are needed. The current work is concerned mostly with execution at (the beginning of) the third stage.

The experimental studies presented in the current work aimed to train participants up to a similar level. However, given time and budgetary constraints, the amount of training that could be provided was relatively limited. As such, it is possible that participants did not fully reach the desired level of competence. This could mean that participants' performance may not be fully representative of routine, automatic, unconscious performance. This is an issue that affects many studies of errors in routine procedural action. For instance, Back et al. (2007) aimed to investigate expert performance, but their experimental participants only executed a small number of training trials. Similarly, Gray's (2000) participants only executed five training trials (plus two familiarization trials) before starting the experimental phase, even though he claimed to test performance at the beginning of Anderson's (1995) third stage. Indeed, this practice seems to be widespread in experimental research on human error. However, there is no simple solution to this problem.

The current work addressed this by investigating participants' performance over time. It was hypothesised that when participants became more skilled at the task, their error rates would reduce until they reach a stable point. A similar approach was taken by Byrne and Bovair (1997), who compared error rates over consecutive trials to assess whether participants had acquired sufficient skill. On the first two studies of this thesis (chapter 5), it was found that the duration of the training was not long enough to allow performance to fully stabilise. This shortcoming was addressed in the later studies, which required participants to train to criterion before starting the main experiment. This led to a substantial increase in the number of training trials to approximately 10 on the Spy task and 7 on the Frankenstein task. This approach was found to be reasonably effective, as the number of errors remained stable throughout the experimental phases. However, it cannot be ruled out that participants were still perfecting their skills while executing the experiment.

A related issue is that it is not easy to determine how much training is enough, and when a participant's skill level has sufficiently approached routine procedural performance. The error rate is a fairly 'crude' measure, and may not capture the finer improvements in skill. Other measures that could be used for future experiments are the time to complete a trial, or participants' ability to multitask while executing the experimental task. This is an issue that should be explored further, because it affects many empirical studies of slip errors in routine action.

Such future work could also include a discussion of the level of skill that is being studied. Many studies of human error seem to adopt as a target domain the operation of complex, safety-critical equipment. Specifically, medical professionals, airline pilots and nuclear power plant operators are often named (e.g. Chung and Byrne, 2004; Byrne and Davis, 2006; Back et al., 2010). As such, it may be beneficial to study in detail what level of performance such operators reach, so that experiments can be conducted at the right level. It could be argued that such experts do not always reach the autonomous level of highly skilled, automatic action, but still rely on conscious processing to a certain extent. For instance, even a highly trained surgeon will need some conscious cognitive processing to successfully perform the surgery.

Moreover, it has been argued that it is not necessary or even desirable to train participants very extensively. The tasks used in experimental studies are arguably less complex than for instance the operation of a nuclear power plant, and therefore less training is needed for participants to reach an expertise level similar to that of nuclear operators. This is the approach taken by Monk (1986) in his study on mode errors. He used a simple task on which participants could quickly attain expert performance, and argued that their level of skill was representative of the level at which operators of more complex systems work.

8.4.2 Generalisation to Real-World Tasks

The current work tested whether an approach from goal relevance could account for (part of) the observed error patterns on the experimental tasks. Of course, two out of the three experimental tasks were designed with device- and task-oriented steps in mind (the Frankenstein and Spy tasks), and the third had a number of very strong device- and task-oriented steps (the Doughnut task) as indicated by the classification study (chapter 4). Indeed, the steps on the Frankenstein task were chosen to be as 'extreme' as possible.

However, this is not necessarily the case on other tasks. In everyday life, the relevance of steps to the task goal is often not as clear or pronounced. In poorly designed devices, different users may even have different mental models of different steps. As such, the extent to which an action is relevant to the main goal may account for a smaller proportion of errors on real-world tasks.

8.4.3 Role of Training

A question that remains is what the role of training is in the representation of steps in the user's mind. Previous work demonstrated that the training is extremely important in providing people with their mental representations of the task and the device (e.g. Kieras and Bovair, 1984;

Cox and Young, 2000). Indeed, the same was found in the current work. However, relatively little attention was paid to the development of the training materials. It was ensured that they were effective for the studies, but the work did not go beyond that.

The main method of training in the current work was to present participants with a cover story for the task, which detailed the function of each step and the order in which they have to be executed. This appeared to be effective, given that the two different cover stories for the Frankenstein task resulted in opposite error patterns. Other approaches have also been taken, such as providing device diagrams to participants (e.g. Kieras and Bovair, 1984; Hiltz et al., 2010). It would be highly beneficial to further investigate the role of training on the mental representations of users, and what the most effective training method is.

A related difficulty is determining whether the chosen method is successful in providing participants with the desired representation of steps. There are currently few reliable methods to address peoples' mental representations of tasks directly, other than testing their performance, such as error rates and step times, as done in the current work. Another approach was taken by Hiltz et al. (2010), who questioned their participants on the operation of the device after presenting them with the instructions. However, this cannot rule out that participants developed a different mental model that happened to make the correct prediction. Indeed, it is likely that people develop their own interpretation of the mental models the experimenter is trying to 'impose' on them. It is difficult, and arguably not even desirable, to prevent this. Therefore, the training materials and experimental set-up should take this into account.

8.4.4 Task Structure

As discussed in the literature review (see chapter 2), task structure has an important impact on the error patterns and step times on hierarchical tasks (e.g. Ruh et al., 2010). Indeed, it may have impacted performance on the experimental tasks as well. However, because the effect of task structure is much better understood than the effect of goal relevance, it was not a main focus of this thesis.

Nevertheless, the task structure of the experimental tasks may have had an impact on the error and step time data. For instance, on the Doughnut task, the first step on each subtask showed a particularly high error rate. Because each of these first steps were device-oriented, this was interpreted as evidence that device-oriented steps have higher error rates than task-oriented steps. However, an alternative explanation is that, since these errors occur at subtask boundaries, they may be a result of the hierarchical task structure rather than their lack of contribution towards the task goal. Indeed, the effect is even stronger on the first subtask of the Doughnut task, which exhibits an unusually high error rate. As such, it is possible that the task structure of the Doughnut task can account for part of the observed error pattern.

Similarly, on the Frankenstein task, there were two steps that showed an unexpected error pattern, beyond what could be explained by the goal relevance hypothesis. Indeed, both of these occurred just after a subtask, and therefore their increased error rates could be explained

by their task structure.

The lack of consideration of task structure is a shortcoming of this work. While task structure as a topic was not the main focus of the thesis, it does appear to have affected the error patterns on the experimental tasks, and therefore its effects should be taken into account. However, it is not immediately clear how to resolve this issue. It is not a feasible option to design the task structure out of an experimental task, because this would result in highly unnatural, flat tasks. Moreover, without subtasks and subgoals, it will be much more difficult to manipulate how relevant each step is to the task goal. A more suitable solution would be to keep the task structure constant across different experimental conditions. Indeed, this was the approach taken by the Frankenstein task, but evidently this is not necessarily sufficient to account for the effects of task structure. Moreover, as Ruh et al. (2010) have shown, different points in the task structure are affected differently by factors such as load, which further complicates the issue. As such, an interesting avenue for future research would be to investigate how task structure and goal relevance interact to affect performance.

8.4.5 Definition of Omission Errors

A novel claim addressed in this work was that device-oriented errors are more likely to be omissions than task-oriented errors. Omission errors were defined as forgetting a single step in the task sequence, and continuing with the next step instead. However, this definition inappropriately included a related error, while being too restrictive about another. For instance, omissions higher up in the task hierarchy, such as the omission of a complete subtask, were not counted as omission errors but instead as 'other' errors.

Omission errors have been inconsistently defined in previous work. For instance, Li et al. (2008) included only single-step skips as omission errors, while subtask skips were considered to be sequence errors instead. Other work has used broader definitions of omission errors. For instance, Cooper and Shallice (2000) define omissions as "an action sequence in which one step or subtask is not performed", thus explicitly including the skipping of subtasks.

In the current work, only single-step skips were defined as omissions, because the hypothesis about device- and task-oriented steps was concerned with the step level, rather than the subtask level. It was predicted that single device-oriented steps may be omitted more frequently, as opposed to subtasks, which in turn may include both device- and task-oriented steps. However, for future work on device- and task-oriented steps, a more appropriate terminology for the type of error targeted may be a "single-step omission".

Another issue related to omission errors in the current work is that they are difficult to distinguish from errors in which two steps are simply reversed. In the data analysis, every skipped single step was counted as an omission. However, this inevitably included errors in which two steps were accidentally reversed, since in this scenario the sequence of steps was indistinguishable from the sequence of steps generated by an omission error. Norman (1981) argued that such reversal errors are distinct from omission errors: the reversal of two steps is

the result of faulty triggering of action schemas, while the skipping of a step results from a loss of activation instead (Norman, 1981).

This issue was caused by the requirement that all errors had to be corrected. A side-effect of this was that it was not possible to distinguish between these two errors, since they resulted in identical sequences of steps. An obvious solution would be not to force participants to correct their errors. However, this is undesirable, because it introduces a large number of other confounds. For instance, some participants may chose to correct errors anyway, making comparison with other participants more difficult. Another example is that in a task with many two-state switches such as the Spy task, this could lead to implausible device states that can hamper future progress.

Future work could investigate whether there are alternative ways to distinguish between omission and reversal errors. An option could be to use eye movements to gain insight into the focus of attention across the task interface.

8.5 Contribution

The current work makes contributions to the theoretical, practical and methodological domains, through a theoretical analysis of previous work and a series of empirical studies.

8.5.1 Theoretical Contribution

The current work makes a theoretical contribution by adding to the body of knowledge on slip errors and routine procedural tasks. Most existing experimental studies on slip errors have been restricted to the post-completion error, with a few investigating other slips such as mode errors. Moreover, the majority of existing work on concepts related to device- and task-oriented steps has been theoretical, and has focussed on learning processes rather than skilled action. The current work takes a novel approach by experimentally investigating device- and task-oriented steps. Three major findings emerged from this work.

First, the experimental studies demonstrated that device-oriented steps are more problematic than task-oriented ones. This is apparent in a number of different measures, as the current work goes beyond the frequently studied error rates, and also explores step times, type of error and eye movements. First, error rates are higher on device-oriented steps than on task-oriented ones. Second, step times are longer on device-oriented steps. Third, the proportion of omission errors is greater, and fourth, the influence of working memory load is greater on device-oriented steps. These findings are consistent throughout the different studies and tasks (though not all experiments addressed all measures). As such, the current work has demonstrated that the differences in performance between device- and task-oriented steps are robust.

Second, the work showed that the concept of goal relevance is at the core of the differences found between device- and task-oriented steps. This hypothesis was first developed through a theoretical analysis of the existing literature on related topics, and further support was provided by a small qualitative study with experts in the field of HCI. It was then tested empirically using

the Frankenstein task, which directly manipulated the goal relevance of steps, while keeping all other factors the same. This study demonstrated that the goal relevance of steps plays an important role in determining the frequency of errors.

Third, the current work also provides a theoretical contribution to the cognitive models of routine procedural action. It was found that the most likely explanation for the experimental results is that steps that have low relevance to the task goal have lower activation levels than steps that are directly relevant to the goal. For the cognitive models, this means that they must allow different steps in a task sequence to have different activation levels, depending on the step's contribution to the main goal.

8.5.2 Practical Contribution

The current work also makes a practical contribution. The knowledge gained from this thesis can be used to mitigate errors in routine procedural tasks. Specifically, the findings suggest that steps that are less relevant to the task goal are inherently problematic, and should be avoided as much as possible in the design of devices. Moreover, the results demonstrated that this is especially the case in high load environments. As such, devices that are operated in busy environments such as hospital wards should ideally be free of device-oriented steps.

8.5.3 Methodological Contribution

The current work also makes a methodological contribution, in the form of an experimental task to study the role of goal relevance in the occurrence of slips in routine procedures: the Frankenstein task. The empirical study of errors is inherently difficult. It is not easy to generate high enough error rates in laboratory settings, while controlling for confounding factors that could affect them. As such, the addition of the Frankenstein task as a successful task to study the effect of goal relevance on slip errors is a useful contribution of this work. Unlike earlier experimental tasks such as the Doughnut task, the Frankenstein task allows the direct investigation of goal relevance, while allowing a range of different measures to be recorded.

It should be noted that the Spy task, while also designed to study device- and task-oriented steps, is not seen as a major methodological contribution, because it does not control for interface factors.

8.6 Directions for Future Research

8.6.1 Cognitive Modelling

An important hypothesis addressed in the current work was that activation levels are lower on those steps that are not directly relevant to the main goal. This was approached by investigating the behavioural 'markers' of activation levels, such as error rates and step times. However, another way to study activation levels in a more direct manner is through the use of cognitive models. After all, the activation levels in cognitive models can be determined directly. Future work can focus on developing a cognitive model that takes into account the goal relevance of steps.

This approach can also be used to assess how the low activation levels on device-oriented steps come about. As noted above, the IAN, M-f-G and SRN models do not currently account for different activation levels on different steps depending on their goal relevance. However, there seem to be several promising mechanisms that could be used to model this. One approach would be to give the model device knowledge and task knowledge, and investigate whether this is sufficient to bring about the different activation levels and associated behavioural effects. Another approach builds on the ACT-R cognitive architecture, on which the M-f-G model is based. It has a built-in 'reward' system, which allows different productions to be associated with different rewards. Using this mechanism, productions representing device-oriented steps could be given a lower reward than productions representing task-oriented steps. This deviates from the activation level hypothesis, but it seems to be a simple and intrinsic way to represent goal relevance while also explaining the experimental results.

8.6.2 Identification of Device- and Task-Oriented Steps

The classification study presented in chapter 4 demonstrated that people are reasonably capable of classifying steps as device- or task-oriented, even when they were previously unfamiliar with the concepts. This is important not only for research on device- and task-oriented steps, but also for the practical applications of the current findings. The results suggest that device-oriented steps are more problematic, and therefore it would be beneficial to design them out of devices. However, in order to do so, device designers must be able to easily identify which steps in a task are device-oriented and which ones are task-oriented.

However, the method employed in the classification study, in which participants read definitions and subsequently classified steps as one or the other, may not be the most appropriate one, even though it provided reasonably successful results for the study. One reason is that it strongly relies on the raters having a correct understanding of the concepts. In the study presented in this thesis, this was achieved by using a number of test cases, and evaluating participants' motivations for classifying these. However, this may not always be possible outside the laboratory, and as such it is possible that a misunderstanding of the concepts causes steps to be wrongly classified. Moreover, this method incorporates a degree of subjectiveness, as demonstrated by the occasional disagreement between participants. This makes it necessary for several people to classify steps, before an 'agreed' classification can be determined.

Thus, future work should develop a framework for easily identifying device- and task-oriented steps. A useful starting point for this is May et al.'s (1993) work on Enabling States Analysis (see also May et al., 1992a,b; Whitefield et al., 1993). As discussed in section 3.2.2.3, they distinguish between work tasks and enabling tasks, with work tasks being directly relevant to the task goal and doing actual work, while enabling tasks are concerned with getting the system into a state in which progress can be made, while they do not do any 'real' work.

May et al. (1993) present a framework for analysing tasks as serving enabling or work goals, which provides a systematic and controlled method for classifying steps. The Enabling States

Analysis as proposed by May et al. (1992a), May et al. (1992b) and May et al. (1993) has a strong design focus, and as such a detailed discussion of this work is beyond the scope of this thesis. Nevertheless, the main four stages of the method are discussed here.

In the first stage, the aims of the system and the context in which it will be used are described. The second stage is concerned with developing a detailed analysis of the goals the user wants to achieve. At this stage, the analysis is still independent of the particular technology used and focusses solely on the goals to be achieved. In the third stage, the enabling states are analysed, thus lending the method its name. This stage builds on the hierarchical task analysis developed in the previous stage. The fourth and final stage takes the enabling states and defines the enabling *tasks* that will be required to reach the enabling states. Whitefield et al. (1993) argue that the enabling tasks should be done by the system as much as possible, in line with the argument made by the current thesis that device-oriented steps should be minimised as much as possible.

The Enabling States method is primarily intended to be used in the design of new systems (Whitefield et al., 1993). However, the same technique is also useful as a starting point for the systematic identification of device-oriented (or enabling) tasks in existing systems. For instance, the technology-independent task analysis generated in the second stage can be compared with an analysis of the existing device. The discrepancies between the two could reveal where there may be enabling steps present. This method is likely to be more systematic and less subjective than the method used in the classification study (chapter 4), and therefore makes for an interesting and relevant topic for future work.

From a classification perspective, the Enabling States method is closely related to Young's (1981) work on task-action mapping. As discussed in section 3.2.1.2, this work analyses the actions necessary to complete the task, and maps them onto the actions necessary to operate the device (see for instance table 3.1). The dissociations in the mapping can reveal which steps are device- and which are task-oriented. Again, this allows for a more objective classification of steps, and is therefore a useful contribution to the future development of a classification framework.

The qualitative study presented in chapter 3 also revealed a number of strategies used by the expert participants to classify steps. For instance, the procedure for doing the task using a particular device can be compared with the procedure for doing the task by hand. Any steps that do not have to be done by hand can be considered device-oriented. Similarly, for each step in the task sequence on a particular device it can be determined whether this step makes a difference in the real world. Steps that do not make a difference are likely to also not contribute to the task goal, and therefore are device-oriented. Another strategy mentioned was to consider for each step whether the designer of the device had a choice in including the step. These strategies can be investigated further in future work, to assess which give the most reliable results.

8.6.3 Error Mitigation

An important finding of this thesis is that device-oriented steps are more problematic than task-oriented ones. This provides strong support for the argument that device-oriented steps should be designed out of devices as much as possible. A direction for future research can be to investigate this in more detail. There are several issues that need to be addressed. First, is designing out steps actually effective in reducing errors? This appears to be a trivial question, but it cannot simply be assumed to be the case. However, there is some experimental evidence that this approach is extremely effective for the post-completion error: Byrne and Davis (2006) found a complete elimination of PCEs when the post-completion task-structure was resolved. Indeed, Zimmerman and Bridger (2000) found that redesigning ATM machines to return the bank card *before* dispensing the cash led to the complete elimination of forgotten bank cards.

Second, to what extent is it feasible to design out all device-oriented steps? Many different constraints affect the design of devices. For instance, if the device interface has a small surface area, it may not be possible to avoid using different modes, thereby introducing mode-switching steps. It is expected that in some devices, it is simply not possible to design out all device-oriented steps, or it may not be feasible for financial or technical reasons. In this case, it is important to also investigate how errors on these steps can be mitigated. Several approaches have been used to reduce slips such as the post-completion error. For instance, Chung and Byrne (2008) have studied the effect of visual cues on the occurrence of the PCE, and found that a cue must appear just-in-time, be salient, and be specific in order to effectively reduce errors. Future work can investigate whether this approach, and others, are also effective in reducing errors on device-oriented steps.

A related approach that can be investigated in future research is whether the perceived goal relevance of steps can be manipulated. A prediction the current work makes is that if a step is manipulated to be directly relevant to the task goal, it will be less likely to be forgotten. Hiltz et al. (2010) did exactly this: they provided participants with an altered goal for their experimental task, in which some of the error-prone steps were made to be directly relevant to it. However, they found that this manipulation was not effective in reducing error rates. Nevertheless, since there were several confounds in their study, it will be worthwhile to study this in more detail in future work.

8.7 Conclusion

Errors in routine procedures do not occur at random, they are more frequent on some steps in a task than on others. The current work investigated one factor that can affect the error pattern on sequential tasks: the relevance of each step to the task goal. An analysis of the previous literature on this topic showed that some steps do not directly contribute to the goal, but are instead required for the operation of the device. These are called device-oriented steps, and contrast with task-oriented steps. These, in turn, are directly relevant to the task and bring the

user closer to their goal.

The main hypothesis addressed in this work is that device-oriented steps are more problematic than task-oriented steps. A number of experimental studies were conducted to address this. The first set of experiments demonstrated that an approach from the relevance of a step to the task goal can explain a substantial part of the error pattern observed on the Doughnut task. The second set of studies showed that, on a task with a wide variety of steps, those that are device-oriented have higher error rates and take longer to execute than those that are task-oriented. The third set of studies directly manipulated goal relevance, and demonstrated that it is indeed an important factor in explaining error rates. Moreover, these studies found longer step times, proportionally more omission errors and a greater susceptibility to working memory load on device-oriented steps. Since the findings are consistent throughout the different studies and tasks, the current work has demonstrated that the differences in performance on device- and task-oriented steps are robust.

The thesis argues that the findings are best explained by lower activation levels on device-oriented steps. This makes it more difficult for these steps to compete with others, thus leading to more errors. It is thought that the lower activation levels are a result of the different ways in which device- and task-oriented steps are represented in people's mental models: device-oriented steps are represented only in people's device model, while task-oriented steps are represented in both their device and task models. Moreover, steps that do not contribute to the task goal have lower cognitive salience than those that do, again leading to lower activation levels.

The results of the current work provide support for the argument that device-oriented steps should be designed out of devices as much as possible. They are more problematic, especially in situations with a high working memory load. Therefore, devices that are operated in busy environments such as hospital wards should ideally be free of device-oriented steps. In conclusion, devices should allow users to focus on *managing the mission, not the machine.*

Take-home message:

Device-oriented steps do not make a direct contribution towards the task goal but are required for the operation of the device, while task-oriented steps bring the user directly closer to their goal. Device-oriented steps are more problematic than their task-oriented counterparts, because the activation levels on these steps are lower. Therefore, these steps should be designed out of devices as much as possible.

Bibliography

Altmann, E. M. and Trafton, J. G. (2002), Memory for Goals: An Activation-Based Model, *Cognitive Science* **26**, 39–83.

Altmann, E. M. and Trafton, J. G. (2007), Timecourse of Recovery From Task Interruption: Data and a Model, *Psychonomic Bulleting & Review* **14**(6), 1079–1084.

Ament, M. G. A., Blandford, A. and Cox, A. L. (2009), Different Cognitive Mechanisms Account for Different Types of Procedural Steps, *in* N. A. Taatgen and H. van Rijn (eds.), *Proceedings of the 31th Annual Conference of the Cognitive Science Society*, Amsterdam, pp.2170–2175.

Ament, M. G. A., Cox, A. L., Blandford, A. and Brumby, D. (2010), Working Memory Load Affects Device-Specific but Not Task-Specific Error Rates, *in* S. Ohlsson and R. Catrambone (eds.), *Proceedings of the 32nd Annual Conference of the Cognitive Science Society*, Portland, OR, pp.91–96.

Ament, M. G. A., Lai, A. Y. T. and Cox, A. L. (2011), The Effect of Repeated Cue Exposure on Post-Completion Errors, *in* L. Carlson, C. Hölscher and T. Shipley (eds.), *Proceedings of the 33rd Annual Conference of the Cognitive Science Society*, Boston, MA.

Anderson, J. R. (1995), *Cognitive Psychology and its Implications*, 4th edition, W H Freeman, New York.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C. and Qin, Y. (2004), An Integrated Theory of the Mind, *Psychological Review* **111**(4), 1036–1060.

Anderson, J. R. and Lebiere, C. (1998), *The Atomic Components of Thought*, Erlbaum, Hillsdale, NJ.

Back, J., Blandford, A. and Curzon, P. (2007), Slip Errors and Cue Salience, *in* W. Brinkman, D. Ham and B. L. W. Wong (eds.), *Proceedings of the 14th European Conference on Cognitive Ergonomics: Invent! Explore!*, London, United Kingdom, pp.221–224.

Back, J., Brumby, D. P. and Cox, A. L. (2010), Locked-out: Investigating the Effectiveness of System Lockouts to Reduce Errors in Routine Tasks, *in Proceedings of the 28th International Conference on Human Factors in Computing Systems*, Atlanta, Georgia, USA, pp.3775–3780.

Back, J., Cheng, W. L., Dann, R., Curzon, P. and Blandford, A. (2006), Does Being Motivated to Avoid Procedural Errors Influence Their Systematicity?, *in Proceedings of HCI 2006*, Vol. 1, pp.2–9.

Baker, S. P., Qiang, Y., Rebok, G. W. and Li, G. (2008), Pilot Error in Air Carrier Mishaps: Longitudinal Trends Among 558 Reports: 1983-2002, *Aviation, Space, and Environmental Medicine* **79**(1), 2–6.

Blandford, A. and Young, R. M. (1998), The Role of Communication Goals in Interaction, *in Adjunct Proceedings of HCI 98*, pp.14–15.

Botvinick, M. and Plaut, D. C. (2002), Representing Task Context: Proposals Based on a Connectionist Model of Action, *Psychological Research* **66**, 298–311.

Botvinick, M. and Plaut, D. C. (2004), Doing Without Schema Hierarchies: A Recurrent Connectionist Approach to Normal and Impaired Routine Sequential Action, *Psychological Review* **111**(2), 395–429.

Botvinick, M. M. and Bylsma, L. M. (2005), Distraction and Action Slips in an Everyday Task: Evidence for a Dynamic Representation of Task Context, *Psychonomic Bulletin & Review* **12**(6), 1011–1017.

Botvinick, M. M. and Plaut, D. C. (2009), Empirical and Computational Support for Context-Dependent Representations of Serial Order: Reply to Bowers, Damian, and Davis (2009), *Psychological Review* **116**(4), 998–1002.

Byrne, M. D. (2008), Preventing Postcompletion Errors: How Much Cue Is Enough?, *in Proceedings of the Thirtieth Annual Conference of the Cognitive Science Society*, Cognitive Science Society, Austin, TX.

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

Byrne, M. D. and Davis, E. M. (2006), Task Structure and Postcompletion Error in the Execution of a Routine Procedure, *Human Factors* **48**(4), 627–638.

Casey, S. M. (1998), *Set Phasers On Stun: And Other True Tales of Design, Technology, and Human Error*, Aegean Publishing Company, Santa Barbara, CA.

Casey, S. M. (2006), *The Atomic Chef: And Other True Tales of Design, Technology, and Human Error*, Aegean Publishing Company, Santa Barbara, CA.

Chan, L. (2008), An Investigation into the Effect of Incentives on Post Completion Errors, Master's thesis, University College London.

Chung, P. H. and Byrne, M. D. (2004), Visual Cues to Reduce Errors in a Routine Procedural Task, *in Proceedings of the Twenty-Sixth Annual Conference of the Cognitive Science Society.*

Chung, P. H. and Byrne, M. D. (2008), Cue Effectiveness in Mitigating Postcompletion Errors in a Routine Procedural Task, *International Journal of Human-Computer Studies* **66**, 217–232.

Churchill, E. F. and Young, R. M. (1991), Modelling Representations of Device Knowledge in Soar, *in Proceedings of AISB-91: Conference of the Society for the Study of Artificial Intelligence and Simulation of Behaviour.*

Cooper, R. P. and Shallice, T. (2000), Contention Scheduling and the Control of Routine Activities, *Cognitive Neuropsychology* **17**(4), 297–338.

Cooper, R. P. and Shallice, T. (2006), Hierarchical Schemas and Goals in the Control of Sequential Behavior, *Psychological Review* **113**(4), 887–916.

Cox, A. L. and Young, R. M. (2000), Device-Oriented and Task-Oriented Exploratory Learning of Interactive Devices, *in* N. Taatgen and J. Aasman (eds.), *Proceedings of the Third International Conference on Cognitive Modelling*, Universal Press, Veenendaal,The Netherlands, pp.70–77.

Epelboim, J., Steinman, R. M., Kowler, E., Edwards, M., Pizlo, Z., Erkelens, C. J. and Collewijn, H. (1995), The Function of Visual Search and Memory in Sequential Looking Tasks, *Vision Research* **35**(23/24), 3401–3422.

Fitts, P. M. (1954), The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement, *Journal of Experimental Psychology* **47**(6), 381–391.

Fitts, P. M. and Posner, M. I. (1967), *Human Performance*, Brooks Cole, Belmont, CA.

Gazzaniga, M. S. (2010), *The Cognitive Neurosciences*, 4th edition, MIT Press, Cambridge, MA.

Gray, W. D. (2000), The Nature and Processing of Errors in Interactive Behaviour, *Cognitive Science* **24**(2), 205–248.

Gray, W. D. (2004), Errors in Interactive Behaviour, *in* W. S. Bainbridge (ed.), *Encyclopedia of Human-Computer Interaction*, Berkshire Publishing Group.

Gray, W. D. and Fu, W.-T. (2004), Soft Constraints in Interactive Behavior: the Case of Ignoring Perfect Knowledge In-The-World for Imperfect Knowledge In-The-Head, *Cognitive Science* **28**, 359–382.

Gray, W. D., Sims, C. R., Fu, W.-T. and Schoelles, M. J. (2006), The Soft Constraints Hypothesis: A Rational Analysis Approach to Resource Allocation for Interactive Behavior, *Psychological Review* **113**(3), 461–482.

Hayhoe, M. and Ballard, D. (2005), Eye Movements in Natural Behavior, *TRENDS in Cognitive Sciences* **9**(4), 188–194.

Henderson, J. M. (2003), Human Gaze Control During Real-World Scene Perception, *TRENDS in Cognitive Sciences* **7**(11), 498–504.

Henderson, J. M., Jr, P. A. W. and Hollingworth, A. (1999), The Effect of Semantic Consistency on Eye Movements During Complex Scene Viewing, *Journal of Experimental Psychology: Human Perception and Performance* **25**(1), 210–228.

Hiltz, K., Back, J. and Blandford, A. (2010), The Roles of Conceptual Device Models and User Goals in Avoiding Device Initialization Errors, *Interacting with Computers* **22**(5), 363–374.

Hodgetts, H. M. and Jones, D. M. (2006), Interruption of the Tower of London Task: Support for a Goal-Activation Approach, *Journal of Experimental Psychology: General* **135**(1), 103–115.

Howes, A. and Young, R. M. (1996), Learning Consistent, Interactive and Meaningful Task-Action Mappings: A Computational Model, *Cognitive Science* **20**, 301–356.

Huang, H., Ruksenas, R., Ament, M. G. A., Curzon, P., Cox, A. L., Blandford, A. and Brumby, D. P. (2011), Capturing the Distinction Between Task and Device Errors in a Formal Model of User Behaviour, *in Proceedings of the 4th International Workshop on Formal Methods for Interactive Systems*.

Jonides, J., Lewis, R. L., Nee, D. E., Lustig, C. A., Berman, M. G. and Moore, K. S. (2008), The Mind and Brain of Short-Term Memory, *Annual Reviews of Psychology* **59**, 193–224.

Kelley, K. D. (2007), An Exploration of Internal Cues to Reduce Omission Errors in a Procedural Task, Master's thesis, University College London, Faculty of Life Sciences.

Kieras, D. E. and Bovair, S. (1984), The Role of a Mental Model in Learning to Operate a Device, *Cognitive Science* **8**, 255–273.

Kirchner, W. K. (1958), Age Differences in Short-Term Retention of Rapidly Changing Information, *Journal of Experimental Psychology* **55**(4), 352–358.

Kirschenbaum, S. S., Gray, W. D., Ehret, B. D. and Miller, S. L. (1996), When Using The Tool Interferes With Doing The Task, *in Proceedings of CHI '96*, Vancouver, Canada, pp.203–204.

Kollarits, J. (1937), Beobachtungen uber Dyspraxien (Fehlhandlungen), *Arkiv fur die Gesamte Psycholgie* **99**, 305–399.

Landis, J. R. and Koch, G. G. (1977), The Measurement of Observer Agreement for Categorical Data, *Biometrics* **33**, 159–174.

Li, S. Y.-W. (2006), An Empirical Investigation of Post-Completion Error: A Cognitive Perspective, PhD thesis, University College London, Department of Psychology.

Li, S. Y.-W., Blandford, A., Cairns, P. and Young, R. M. (2008), The Effect of Interruptions on Postcompletion and Other Procedural Errors: An Account Based on the Activation-Based Goal Memory Model, *Journal of Experimental Psychology: Applied* **14**(4), 314–328.

Li, S. Y. W., Cox, A. L., Blandford, A., Cairns, P. and Abeles, A. (2006), Further Investigations into Post-Completion Error: the Effects of Interruption Position and Duration, *in Proceedings of CogSci2006, the Twenty-Eighth Annual Meeting of the Cognitive Science Society*, Vancouver, Canada.

May, J., Byerley, P. F., Denley, I., Hill, B., Adamson, S., Patterson, P. and Hedman, L. (1993), The enabling states method, *in* P. F. Byerley, P. J. Barnard and J. May (eds.), *Computers, Communication and Usability: Design Issues, Research and Methods for Integrated Services*, Elsevier Science Publishers, Amsterdam, Chapter 3.1, pp.247–290.

May, J., Denley, I., Voigt, U.-B., Hermann, S. and Byerley, P. F. (1992a), Designing IBC services which enable users to reach their goals, *in* G. C. van der Veer, M. Tauber, S. Bagnara and S. Antalovits (eds.), *Proceedings fo the 6th European Conference on Cognitive Ergonomics*, Rome, pp.117–133.

May, J., Whitefield, A., Denley, I., Voigt, U.-B., Hermann, S. and Esgate, A. (1992b), Integration of services for human end-users (2): A Case Study of a Cooperative Document Production System, *in* P. F. Byerley and S. Connell (eds.), *Integrated Broadband Communications: Views from RACE-Usage Aspects*, Elsevier Science Publishers, Amsterdam, Chapter 4.3.

Monk, A. (1986), Mode Errors: A User-Centred Analysis and Some Preventative Measures Using Keying-Contingent Sound, *International Journal of Man-Machine Studies* **24**, 313–327.

Moran, T. P. (1983), Getting Into a System: External-Internal Task Mapping Analysis, *in Proceedings of CHI '83*, pp.45–49.

Newell, A. (1990), *Unified Theories of Cognition*, Harvard University Press, Cambridge, MA.

Newell, A. and Simon, H. A. (1972), *Human Problem Solving*, Prentice Hall, Englewood Cliffs, NJ.

Norman, D. A. (1981), Categorization of Action Slips, *Psychological Review* **88**, 1–15.

Norman, D. A. and Shallice, T. (1986), Attention to Action - Willed and Automatic Control of Behavior, *in* R. J. Davidson, G. E. Schwartz and D. Shapiro (eds.), *Consciousness and Self-Regulation: Advances in Resarch and Theory*, Plenum Press, Chapter 1, pp.1–18.

Payne, S. J. (1994), Acquisition of Display-Based Skill, *in CHI '94 Conference Companion*, Boston, MA.

Payne, S. J., Squibb, H. R. and Howes, A. (1990), The Nature of Device Models: The Yoked State Space Hypothesis and Some Experiments With Text Editors, *Human-Computer Interaction* **5**, 415–444.

Rasmussen, J. (1982), Human Errors. A Taxonomy for Describing Human Malfunction in Industrial Installations, *Journal of Occupational Accidents* **4**, 311–333.

Rasmussen, J. (1987), Cognitive Control and Human Error Mechanisms, *in* J. Rasmussen, K. Duncan and J. Leplat (eds.), *New Technology and Human Error*, Wiley, London, Chapter 11.

Ratwani, R. M. (2008), A Spatial Memory Mechanism for Guiding Primary Task Resumption, PhD thesis, George Mason University.

Ratwani, R. M. and Trafton, J. G. (2009), Developing a Predictive Model of Postcompletion Errors, *in* N. Taatgen and H. van Rijn (eds.), *Proceedings of the 31th Annual Conference of the Cognitive Science Society*, Amsterdam.

Ratwani, R. M., McCurry, J. M. and Trafton, J. G. (2008), Predicting Postcompletion Errors using Eye Movements, *in Proceedings of CHI '08*, pp.539–542.

Reason, J. (1979), Actions Not As Planned: The Price of Automatization, Aspects of Consciousness, Volume 1: Psychological Issues, Wiley, London, United Kingdom.

Reason, J. (1990), *Human Error*, Cambridge University Press.

Reason, J. (2002), Combating Omission Errors Through Task Analysis and Good Reminders, *Quality and Safety in Health Care* **11**, 40–44.

Rieman, J., Byrne, M. D. and Polson, P. G. (1994), Goal Formation and the Unselected Window Problem, *in CHI' 94 Research Symposium*, Boston, MA.

Ruh, N., Cooper, R. P. and Mareschal, D. (2010), Action Selection in Complex Routinized Sequential Behaviors, *Journal of Experimental Psychology: Human Perception and Performance* **36**(4), 955—-975.

Schmider, E., Ziegler, M., Danay, E., Beyer, L. and Buhner, M. (2010), Is It Really Robust?: Reinvestigating the Robustness of ANOVA Against Violations of the Normal Distribution Assumption, *Methodology* **6**(4), 147–151.

Sellen, A., Kurtenbach, G. and Buxton, W. (1992), The Prevention of Mode Errors Through Sensory Feedback, *Human-Computer Interaction* **7**, 141–164.

Senders, J. W. and Moray, N. P. (1991), *Human Error: Cause, Prediction and Reduction*, Erlbaum, Hillsdale, NJ.

Spearman, C. (1928), The Origin of Error, *Journal of General Psychology* **1**, 29–53.

Tamborello, F. P. (2009), A Computational Model of Routine Procedural Memory, PhD thesis, Rice University, Houston, TX.

Tesler, L. (1981), The Smalltalk Environment, *Byte* pp.90–147.

Trafton, J. G., Altmann, E. M. and Ratwani, R. M. (2011), A Memory For Goals Model of Sequence Errors, *Cognitive Systems Research* **12**, 134–143.

Tsai, J. and Byrne, M. D. (2007), Evaluating Systematic Error Predictions in a Routine Procedural Task, *in Proceedings of the Human Factors and Ergonomics Society 51st Annual Meeting*.

Whitefield, A., Esgate, A., Denley, I. and Byerley, P. F. (1993), On distinguishing work tasks and enabling tasks, *Interacting with Computers* **5**(3), 333–347.

Young, R. M. (1981), The Machine Inside the Machine: Users' Models of Pocket Calculators, *International Journal of Man-Machine Studies* **15**(1), 51–85.

Young, R. M. (1983), Surrogates and Mappings: Two Kinds of Conceptual Models for Interactive Devices, *in* D. Gentner and A. L. Stevens (eds.), *Mental Models*, Lawrence Erlbaum Associates, pp.37–52.

Zimmerman, C. M. and Bridger, R. S. (2000), Effects of Dialogue Design on Automatic Teller Machine (ATM) Usability: Transaction Times and Card Loss, *Behaviour & Information Technology* **19**(6), 441–449.

Appendix A

Instructions given to participants

A.1 Classification Study

These instructions were presented to participants on paper.

A.1.1 Overview

Thank you for taking part in this study of Human Computer Interaction. The purpose of this study is to evaluate a concept that is useful in explaining why people make errors.

The session will be conducted as follows. First, you will be introduced to the concept being evaluated in this study. You will then be taught how to execute a number of different tasks. After this, you will be asked to apply your understanding of the concepts to the tasks.

A.1.2 Concepts

The first part of this study is to familiarise yourself with the concepts being studied. Please read the following section carefully. If anything is unclear, or if you have any questions, please ask the experimenter.

Consider a task in which you have to use a certain device to achieve a goal. We will use the example of the SuperCoffee 3000, a state-of-the-art espresso maker. You have used this machine for over a year now, so you are very familiar with the way it works, and you have developed your own routine for operating the machine.

As with any coffee maker, there are certain actions in your routine that are very important for achieving your goal of making a cup of coffee in the morning. For instance, you have to put water into its reservoir, add coffee beans, and place the coffee cup in the right place. Without these actions, you will not be able to make your cup of coffee. *These actions are referred to as task-oriented.*

Like any machine, the SuperCoffee 3000 has its own instructions and way it must be operated. For instance, before you can specify how many cups of coffee you want, you have to hold down a button before you can use the 'increase number of cups' button. Another example: before you can grind the coffee beans, you need to make sure the SuperCoffee 3000 is in grinding mode by pressing the grinding button. These actions are not so much concerned with your main goal of making coffee, but more with the operation of the SuperCoffee 3000 machine. *These*

actions are referred to as device-oriented.

A.1.2.1 Definitions

> *"Device-oriented steps are those that are more concerned with the operation of the device than with the achievement of the main task goal."*
>
> *"Task-oriented steps are those that are more concerned with the achievement of the main task goal, than with the operation of the device."*

A.1.2.2 Characteristics

In this study, you will be asked to classify a number of actions as device-oriented or task-oriented. However, it may not always be obvious whether an action is one or the other, or perhaps something different altogether. The following characteristics can be useful in determining the classification of an action. Note that some actions may possess characteristics from both columns.

Device-Oriented	**Task-Oriented**
Concerned with operating the device	Concerned with achieving the main goal
Specific to a particular device	Required regardless of the type of device
Can be omitted if the device allows	Cannot be omitted

A.1.3 Tasks

You will now be taught how to do the tasks. Please ask the experimenter to start your training.

A.1.4 Classification

Now that you have been trained on the tasks, you will be asked to classify a number of the actions from the task sequence. The experimenter will give you a number of cards that each show an interaction element that corresponds to a step. You will then be asked to sort the cards into three piles:

- Device-oriented
- Task-oriented
- Other/don't know/ambiguous

You can refer to the definitions above at any time during the process. The experimenter may also ask you to motivate your choices and will record your answers. Please note that there is no right or wrong classification.

Please ask the experimenter to start the procedure.

A.2 The Doughnut Task

These instructions were presented to participants on paper.

Thank you for taking part in this study of Human-Computer Interaction. In this session you will be playing a computer-based game called "The Wicket Doughnut Machine".

In this game you are the baker and you have two tasks:

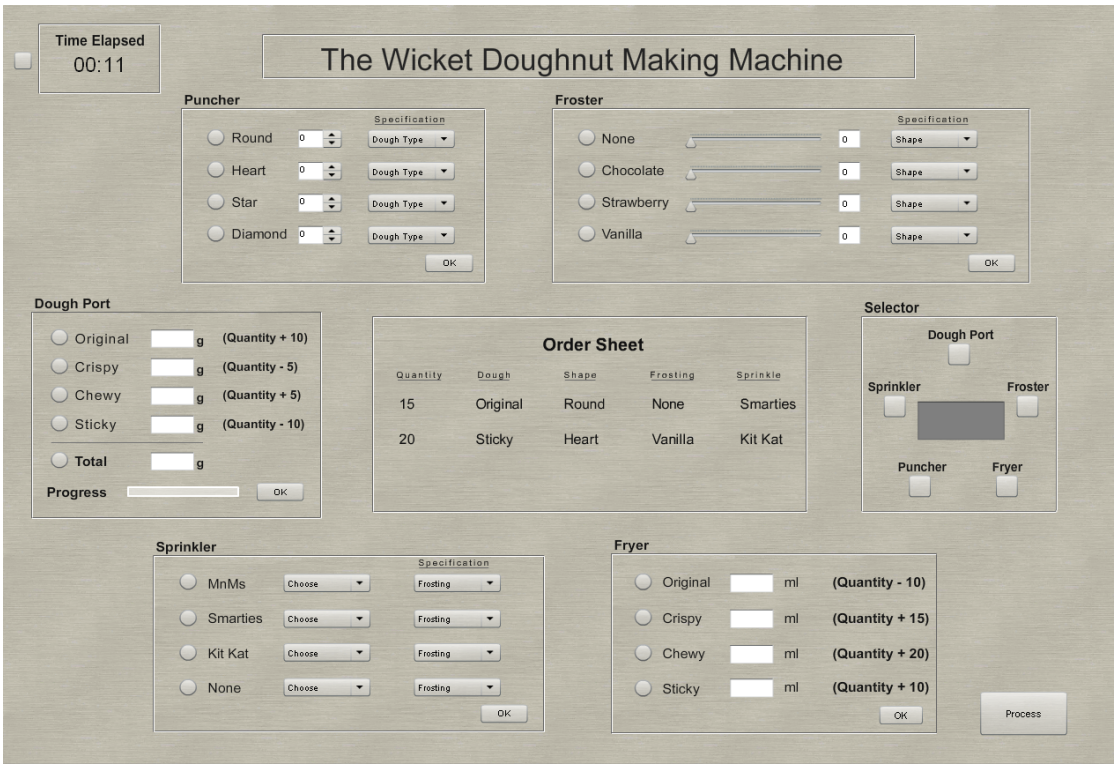
1) Make doughnuts using the Wicket Doughnut Machine. However, the doughnut machine is slightly faulty, and you, as the baker, have to try your best to make the right amount of doughnuts ordered! It's no good to your bakery to produce more than the order and it's definitely not a good way to keep your customers to make less than the order!

2) Monitor the number of doughnuts sold using the Doughnut Live Feed. The shops you are baking the doughnuts for have a promotion each week for a particular type of doughnut. While baking the doughnuts with the machine, you also have to count how many doughnuts of this type were sold. Each time this number reaches 20, you need to notify the shop.

There is a specific way to operate the machine and the call centre. You will be walked through the instructions step-by-step.

The Wicket Doughnut Machine

Below is the Wicket Doughnut Machine. It has five main compartments: Dough Port, Froster, Fryer, Puncher and Sprinkler.



The five compartments need to be operated in the following sequence:

Dough Port → Puncher → Froster → Sprinkler → Fryer

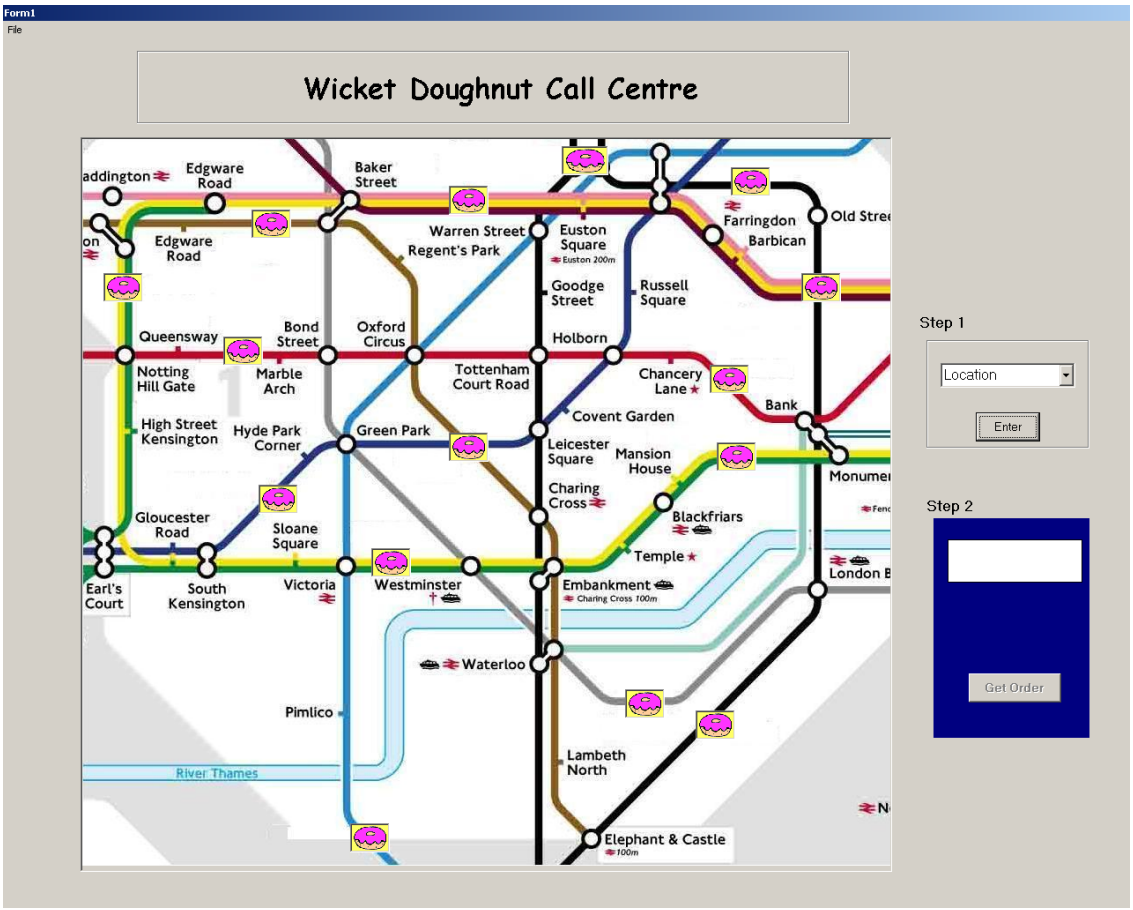
Otherwise, the machine will not work!

Each compartment needs to be activated with the Selector before you can use it. A simple click on the appropriate Selector button will do the trick. After completing all the compartments in the right order, the doughnuts are produced by clicking the Process/Clean button. A small screen will pop up telling you if you were successful, and how many doughnuts were made. The doughnut machine then needs to be cleaned by clicking the Process/Clean button.

A new call will then come in from one of the shops in London, which requires a switch to the call centre.

The Call Centre

Below is the Call Centre. It is operated in two steps. First, select the location indicated by the Wicket Doughnut Machine and click enter. Second, find the location on the map, and drag it into the white box in step 2. Click 'Get Order' to finish the task and return to the Wicket Doughnut Machine. The next order will then already be waiting for you!



Doughnut Live Feed

While making doughnuts using the machine, you need to monitor the number of doughnuts sold of a specific kind. Once the number reaches 20, the shops need to be notified of this. You can do this by clicking the button on the Doughnut Live Feed.

After notifying the shops, you should start counting again from zero. If the end of a trial is reached and the count has not reached 20, carry the number over to the next (do not start from 0 when starting a new trial).

You do not need to monitor the doughnut sales while you are using the call centre.

You will need to do a total of 11 trials on the Wicket Doughnut Machine

This is not an assessment or a test of IQ. You are only required to follow the instructions just presented and carry out the trials. However, you are encouraged to perform as quickly and accurately as you can in each task and in each trial.

If you have any questions, please ask the experimenter now.

A.3 The Spy Task

These instructions were presented to participants on-screen.

Your future mission will consist of two important tasks. First, you must fly the plane to a secret destination. To do this, you must first prepare the plane for take-off.

Turn on the cockpit by clicking the Power button.

You must make sure you are navigating to the correct destination. Enter the destination that you have received from your contact, and confirm.

Before being allowed to take off, you must have permission from Air Traffic Control (ATC), to avoid collision with other planes.

This is done by radio transmission.

To be able to talk to ATC, you must first turn on the radio signal by clicking the radio transmission button.

Because ATC gets many different requests, you must specify the type of your request: whether you require permission for take-off or for landing.

Select the appropriate button.

Submit your request to ATC. If they answer positively, you are free to take off. If not, you must repeat the request.

Before proceeding, turn radio transmission off again to prevent your enemies from being able to track you.

You are now ready for take-off!

Before your plane can go anywhere, you must release the brakes.

To take off, increase the thrust generated by the engines. The plane will slowly start moving.

In order to create lift, you must extend the wing flaps.

Now you must wait until the plane gains enough speed and lift so that it takes off.

When the plane is in the air, take in the landing gear to reduce the amount of friction.

The plane's autopilot will now fly you to the destination you specified earlier.

When you are nearing your destination, reduce speed by decreasing the thrust generated by the engines to 0.
This will start the descent and you will glide towards the landing strip.

Before being allowed to land, you must speak to ATC at the destination airport. Turn on radio transmission.

Specify the type of request, and submit your landing request.

Once you receive your landing permission, switch radio transmission off again.

You are now approaching the landing strip, so retract your wing flaps...

... and lower your landing gear.

Immediately after the plane has touched down, switch the engines to reverse thrust to slow the plane down. You cannot apply your brakes yet
because they will cause the plane to skid off the landing strip.

Once the plane has slowed down to a halt at the secret spy terminal, apply the hand-brake.
This will automatically turn off the engines as well, because the hand-brake is not strong enough to cope with them while they are on.

You are about to send your secret message, so to avoid any eavesdroppers you must turn off the cockpit.

The engines are still in reverse-thrust. They have now sufficiently slowed and cooled down, so you must now switch them back to forward thrust.

You have now completed the first part of the task.

Now that you have arrived at the spy terminal of the destination airport, you must deliver the secret message.
The panel to do this is hidden to avoid detection in case your plane is captured.
So you must first bring up this hidden panel by clicking in the area to the left of the switch panel.

To make sure no terrorists or hostile governments are eavesdropping on your secret message, you must establish a secure, encrypted connection.
Simply flip the switch to turn on the encrypted connection.
Once you have established this, you can safely transmit the message.

Type your secret message into the field, and submit it. The spy at the receiving end will then give you the next message to deliver, and your next destination.

You must end the encrypted transmission.
Then hide the secret compartment again so you can safely fly to the next destination.

You have now completed your first mission.

A.4 The Frankenstein Task

The instructions appear on-screen. Page breaks are indicated with a horizontal line.

A.4.1 Create Condition

Welcome! You are in your last year of study at the Frankenstein Institute of Synthetic Biology.

Before you graduate, you must do one final test: you must demonstrate that you have mastered the art of Monster Making.

Your assignment is to use the Galvanizer 3000, a state-of-the-art Monster Making Machine, to create 10 new monsters.

You will now be taught how to make monsters using the Galvanizer 3000.

Click the Start button to begin.

Thank you for taking part in this study of Human-Computer Interaction. In this session, you will be playing a computer-based game called "The Frankenstein Game". In this game, you are a student at the Frankenstein Institute of Synthetic Biology. For your final year project, you must demonstrate your monster-making skills using a special monster-making machine: the Galvanizer 3000.

To make a monster, you have to do the following:
1) Get the instructions for the new monster. These will tell you how to make the monster, but they are hidden in a secret location. You will be given a set of coordinates that indicate where

you can find the instructions. Find them on the map, and the instructions will appear.

2) Build the new monster. You have to specify the type of legs, arms, body, and head. It is important to make the monster exactly as in the instructions, otherwise you might fail your assignment! It is no good to put the wrong head or arms on the monster, or leave it unpainted. When you are done building the monster, you have to bring it to life!

There is a specific way to operate the machine and get the instructions. You will be walked through the process step-by-step.

Press the 'Next' button to continue.

How to Get the Instructions

Below is the instruction screen, which allows you to find the instructions for the monster.

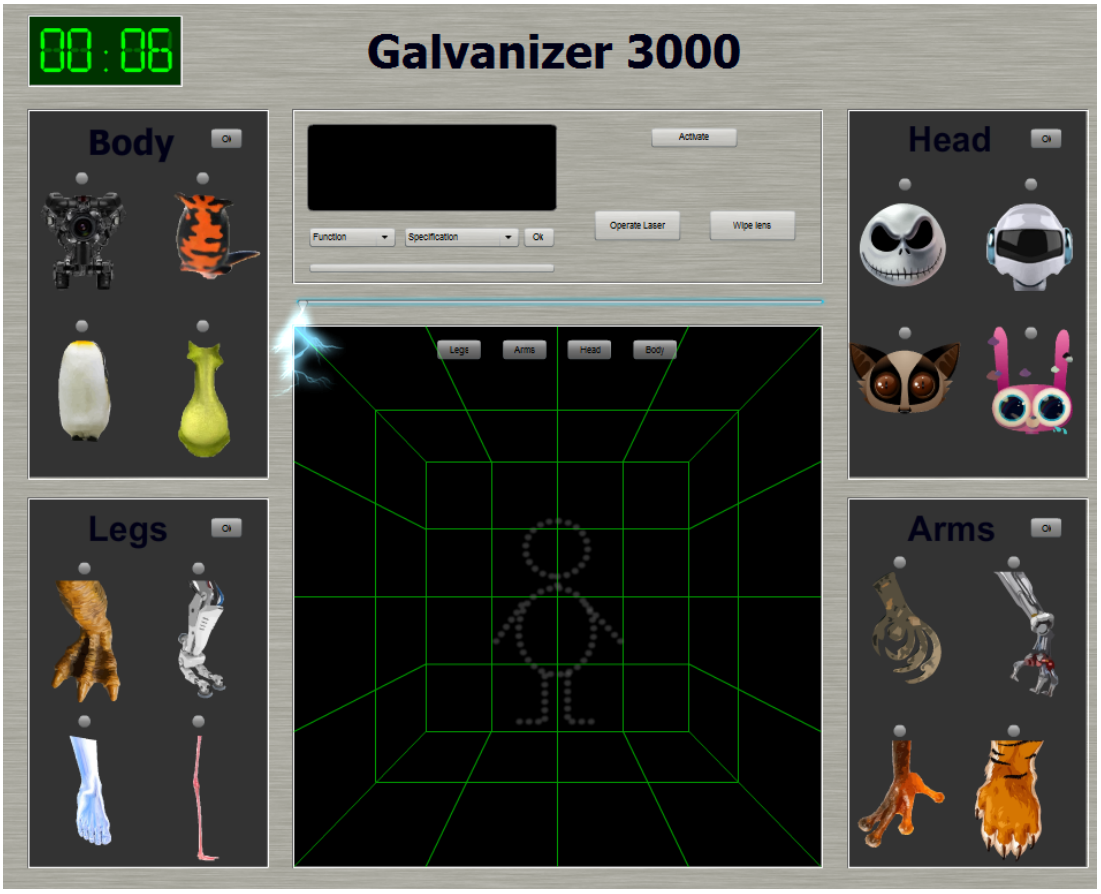
To find out where the instructions are hidden, you must first contact Dr Frankenstein (headmaster of the Frankenstein Institute of Synthetic Biology), by clicking the telephone. He will provide you with the coordinates where you can find the instructions. Click the 'Get Coordinates' button, and they will be displayed on the small screen on the right.

You must then enter these coordinates on the map using the sliders. To do this, activate the relevant slider by clicking its display window. You can now drag the slider to the right coordinate; the display window will show the number. Repeat this procedure on the other slider.

Once you have entered the coordinates correctly, press the 'Get Monster' button. The monster specifications will then be shown.

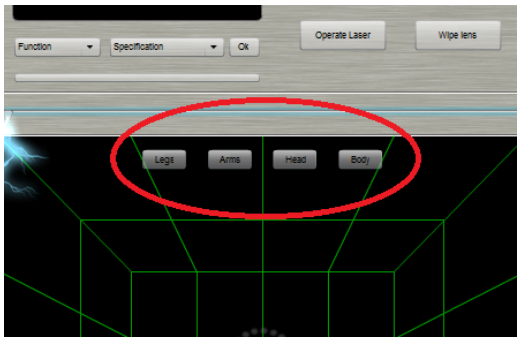
The Galvanizer 3000

Below is the Galvanizer 3000, the machine you will use to make the monsters. It has 6 main widgets: the Arms widget, the Legs widget, the Head widget, the Body widget, the Programmer, and the Monster Viewer.



The first part of making the monster requires you to specify the body parts. You have to specify these in the following order:
Body → Legs → Arms → Head
otherwise the machine will not work.

Before you can specify a body part, you need to activate the appropriate widget. This is done by clicking the corresponding button in the Monster Viewer.

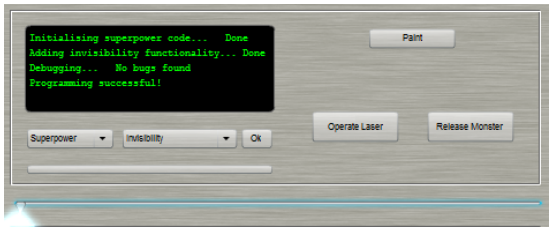


To add a body part, simply select the required item, and click the 'OK' button to add it to the monster.

After you have selected all the body parts and added them to the monster, you have to paint the monster so that it will have the correct colour. You can do this by pressing the 'Paint' button.

Once you have specified all body parts in the right order and painted it, the monster is ready to be brought to life! To do this, you have to program it, and blast it with a powerful laser.

The instruction drawings will show you how the monster should be programmed. Select the correct function from the drop-down menu, and subsequently the correct specification. Press 'OK' to confirm your selection and program the monster. The screen will show the progress of the programming sequence. You must then deactivate the screen by clicking long button under the drop-down menus.



When you have successfully programmed the monster, it is time to bring it to life! The instructions will tell you in what position the laser has to be in order to work properly. Slide the laser head to this position, and press the 'Operate Laser' button to Galvanize!

It's ALIVE!!!

The last step is to release the monster into the wild; simply click the 'Release Monster' button.

You will now see a demonstration of the monster-making process. After the demonstration, you will be able to practise a number of times. Press Next to continue.

A.4.2 Destroy Condition

It is the year 2546. Humanity has found the formula for creating Life, and is re-creating the many species lost in the Global Warming Catastrophe of 2010.

However, the evil lord Frankendoom has gotten hold of the formula, and is abusing it to create monsters.

These monsters are destroying the world and threatening the existence of the human race!

Your job is to save the human race from extinction by destroying the monsters! You will use a special machine to do this.

Are you ready to take on this grand challenge?

Click the Start button to begin your training!

Thank you for taking part in this study of Human-Computer Interaction. In this session, you will be playing a computer-based game called "The Frankenstein Game".

In this game, you must save the world from destruction. Your mission is to stop the monsters created by the evil Dr. Frankendoom from taking over the world. To do this, you use a special machine that disassembles the monsters and prepares them for further research in UCL's laboratories.

To destroy a monster, you have to do the following:
1) Capture the monster. You will receive a call from the Monster Squad, who will give you a set of coordinates that indicate where you can find the monster. Find them on the map, and the monster will be captured.
2) Disassemble the monster. You have to remove the legs, arms, body and head separately. After this, you must classify each of the body parts so that the scientists in UCL's laboratories can study them further. This way, they can find the monsters' weaknesses and find out how to stop them from destroying the world.

There is a specific way to operate the disassembling machine and to capture the monster. You will be walked through the process step-by-step.

Press the 'Next' button to continue.

How to Capture the Monster

Below is the Capture screen, which allows you to answer telephone calls and find the monster and capture it.

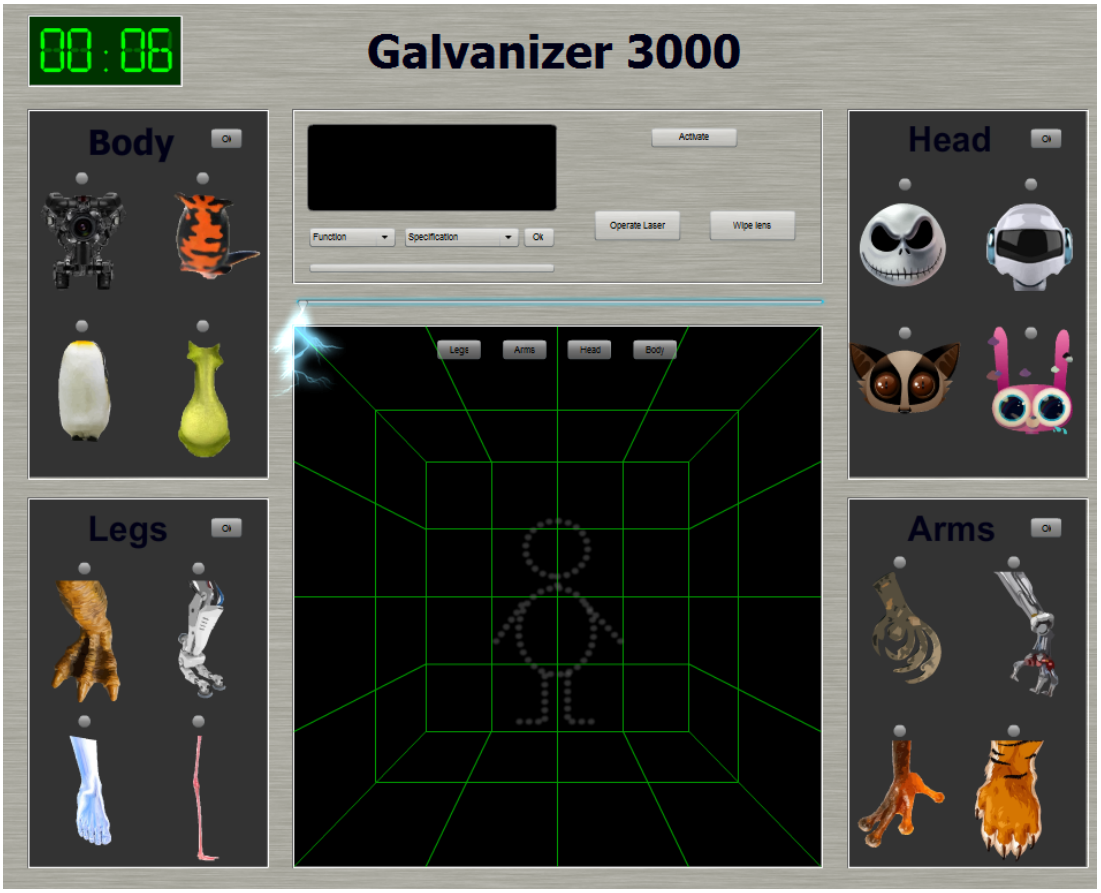
To answer an incoming call, simply click the telephone. The person on the other end will provide you with the coordinates where you can find the monster. Click the 'Get Coordinates' button, and the coordinates will be displayed on the small screen on the right.

You must then enter these coordinates on the map using the sliders. To do this, drag the relevant slider to the correct coordinate using its green button; the display window will show the number. Deactivate the slider by clicking the relevant display window. Repeat this procedure with the other slider.

Once you have entered the coordinates correctly, press the 'Capture Monster' button. The monster will then be revealed at its location and will be captured.

The Disassembler

Below is the Disassembler, the machine you will use to disassemble the monsters. It has 6 main widgets: the Arms widget, the Legs widget, the Head widget, the Body widget, the Wiper, and the Monster Viewer.

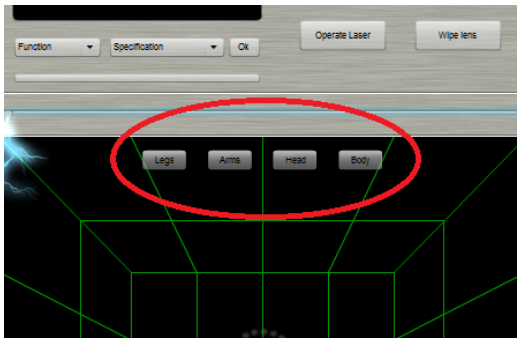


The first part of disassembling the monster requires you to remove the body parts and classify them. You have to remove them in the following order:
Head → Arms → Legs → Body
otherwise the machine will not work.

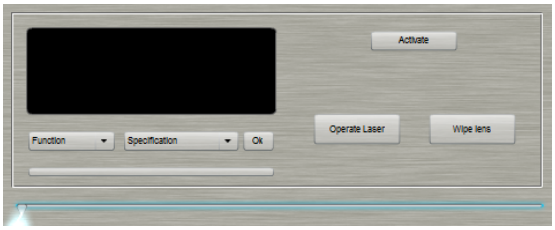
To remove a body part, click the corresponding button in the Monster Viewer.

You can then classify the body part, and clicking the 'OK' button will confirm this. You can then move on to removing the next body part.

Once you have removed all body parts in the right order, the disassembly process is almost complete. You must now wipe the monster's brain, and blast it with a powerful laser so it can never do evil again.



To do this, activate the Wiper widget, and select the function to wipe from the drop down menus, then press 'OK'. Before the action will be carried out, the small screen will display a warning that you are about to wipe the monster's brain. To execute the wiping action, press the long button underneath the drop down menus.



After you have successfully wiped the monster, you must blast it with a powerful laser to make sure it can never do evil again. Activate the laser by clicking its head, and press the 'Operate Laser' button.

The monster has now been destroyed! You have successfully complete one mission. The UCL scientists will now take over and investigate the monster further.

The last step is to deactivate the laser, simply click the 'Deactivate Laser' button.

You will now see a demonstration of the monster-disassembly process. After the demonstration, you will be able to practise a number of times. Press Next to continue.

A.4.3 N-Back Task

A digit will be presented on the screen roughly every 3 seconds. Your job is to compare this digit to a previous one, and decide if it is the same.

On the first block of trials, you will be comparing the current digit with the previous one (1-back). In the second block, you will be comparing the current digit with one you saw two digits ago (2-back). On the third run, you will compare the digit to the one you saw three digits ago (3-back), and so on. The maximum number of digits-back is 5.

For instance, imagine you are comparing the current digit to the 2-back. You see these digits:

3 → 2 → 3 → 6 → 5 → 6 → 3 → 4 → 3 → etc

Your responses should be:

S → D → D → S → D → D → S → etc

Responding is easy. If you decide that the digit is the same as n-back, simply click on the digit. If you decide it is not the same, do nothing. A green V will appear if your response (or lack thereof) is correct, and a red X will appear if it is not.

Please note that you may find the task very difficult when the number of digits-back goes up. You do not need to worry if you get many wrong; this is completely normal. The task is simply designed to be very difficult.

Appendix B

Classifications

Doughnut Task				
Step	**Expert 1**	**Expert 2**	**Novice 1**	**Novice 2**
0. Respond to Call Centre		*Not rated*		
1. New order	T	T	T	T
2. Activate dough port	D	D	D	D
3. Enter dough port data	T	T	T	T
4. Confirm dough port	NS	T	D	T
5. Wait for progress bar	D	D	D	D
6. Activate puncher	D	D	D	D
7. Enter puncher data	T	T	T	T
8. Confirm puncher	NS	T	D	T
9. Activate froster	D	D	D	D
10. Enter froster data	T	T	T	T
11. Confirm froster	NS	T	NS	T
12. Activate sprinkler	D	D	D	D
13. Enter sprinkler data	T	T	T	T
14. Confirm sprinkler	NS	T	NS	T
15. Activate fryer	D	D	D	D
16. Enter fryer data	T	T	T	T
17. Confirm fryer	NS	T	NS	T
18. Process order	T	T	D	T
19. Dismiss pop-up		*Not rated*		
20. Clean doughnut machine	D	D	D	T

Table B.1: *Classification of all the steps of the Doughnut Task. 'D' denotes device-oriented, 'T' denotes task-oriented, and 'NS' indicates that the participant was not sure.*

Spy Task				
Step	**Expert 1**	**Expert 2**	**Novice 1**	**Novice 2**
Flying part				
1. Activate the plane/main display	D	D	T	D
2. Select destination	T	T	T	T
3. Confirm destination	T	D	NS	T
4. Switch on radio	D	T	D	D
5. Select request type	T	T	NS	T
6. Submit request to ATC	T	T	T	T
7. Switch radio off	D	D	D	D
8. Release brakes	D	T	D	D
9. Increase engine thrust	T	T	T	T
10. Extend wing flaps	T	T	T	T
11. Take in landing gear	T	T	T	T
12. Decreasing engine thrust	T	T	T	T
13. Switch on radio	D	T	D	D
14. Select request type	T	T	T	T
15. Submit request to ATC	T	T	T	T
16. Switch off radio	D	D	D	D
17. Retract wing-flaps	T	T	T	T
18. Extend landing gear	T	T	T	T
19. Switch engines to reverse thrust	T	T	T	D
20. Apply brakes	D	T	T	T
21. Switch plane/main display off	D	D	D	D
22. Switch engines back into forward thrust	D	D	T	D
Delivery part				
23. Activate secret transmission compartment	D	D	T	D
24. Establish encrypted transmission channel	D	T	T	D
25. Type in message	T	T	T	T
26. Send message and receive new destination	T	T	T	T
27. Turn off encrypted transmission	D	T	NS	D
28. Deactivate secret transmission compartment	D	D	T	D

Table B.2: *Classification of all the steps of the Spy Task. 'D' denotes device-oriented, 'T' denotes task-oriented, and 'NS' indicates that the participant was not sure.*

Frankenstein Task - Create Condition				
Step	**Expert 1**	**Expert 2**	**Novice 1**	**Novice 2**
1. Pick up telephone		*Not rated*		
2. Request coordinates		*Not rated*		
3. Activate slider	D	D	D	D
4. Move slider	T	T	T	D
5. Get monster		*Not rated*		
6. Activate body widget	D	D	D	D
7. Select body		*Not rated*		
8. Confirm body		*Not rated*		
9. Activate leg widget	D	D	D	D
10. Select legs		*Not rated*		
11. Confirm legs		*Not rated*		
12. Activate arm widget	D	D	D	D
13. Select arms		*Not rated*		
14. Confirm arms		*Not rated*		
15. Activate head widget	D	D	D	D
16. Select head		*Not rated*		
17. Confirm head		*Not rated*		
18. Paint monster	T	T	T	T
19. Select functions to program monster		*Not rated*		
20. Debug and program		*Not rated*		
21. Deactivate widget	D	D	D	D
22. Position laser	T	D	NS	T
23. Blast laser		*Not rated*		
24. Release monster	T	T	T	T

Table B.3: *Classification of all the steps of the Frankenstein Task (Create condition). 'D' denotes device-oriented, 'T' denotes task-oriented, and 'NS' indicates that the participant was not sure.*

Frankenstein Task - Destroy Condition				
Step	**Expert 1**	**Expert 2**	**Novice 1**	**Novice 2**
1. Pick up telephone		*Not rated*		
2. Request coordinates		*Not rated*		
3. Move slider	T	T	D	T
4. Deactivate slider	D	D	D	D
5. Get monster		*Not rated*		
6. Remove monster head	T	T	T	T
7. Classify head		*Not rated*		
8. Confirm head		*Not rated*		
9. Remove arms	T	T	T	T
10. Classify arms		*Not rated*		
11. Confirm arms		*Not rated*		
12. Remove legs	T	T	T	T
13. Classify legs		*Not rated*		
14. Confirm legs		*Not rated*		
15. Remove body	T	T	T	T
16. Classify body		*Not rated*		
17. Confirm body		*Not rated*		
18. Activate wiper widget	D	D	D	D
19. Select function to wipe		*Not rated*		
20. Confirm		*Not rated*		
21. Wipe monster brain	D	T	T	T
22. Activate laser	D	D	D	T
23. Blast laser		*Not rated*		
24. Deactivate machine	D	D	D	T

Table B.4: *Classification of all the steps of the Frankenstein Task (Destroy condition). 'D' denotes device-oriented, 'T' denotes task-oriented, and 'NS' indicates that the participant was not sure.*

Appendix C

Examples of Errors Occurring in Real Life

This appendix shows a number of light-hearted examples of slip errors that have occurred in real life. It is not meant to be a serious contribution to this thesis, but it merely shows that, occasionally, errors can be amusing too.

Figure C.1: *An extreme version of a post-completion error. After completing the main goal of getting petrol, the driver appears to have omitted the entire clean-up subgoal of removing the fuel hose, and replacing the gas cap. From Failblog.org, image uploaded by Anonymous.*

Figure C.2: *This error is apparently not a one-off fail. From Failblog.org, image uploaded by Anonymous.*

Figure C.3: *In fact, it seems to happen rather frequently. From Failblog.org, image uploaded by Anonymous.*

Figure C.4: *I was making a bread in the breadmaker, but failed to notice that I had forgotten to place the kneading paddle in the bread pan before starting the baking process. The result: loose ingredients baked to perfection.*

Figure C.5: *It appears the car's driver forgot to close the sun roof after parking the car. From Failblog.org, image uploaded by Gilly.*

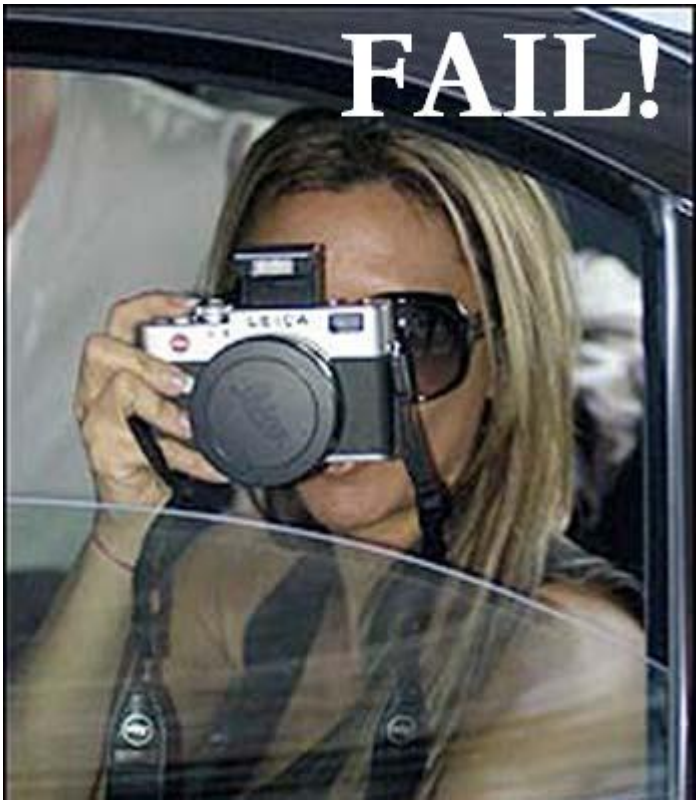


Figure C.6: *It appears the lady in the picture has forgotten to remove the lens cap before attempting to take a photograph. From Failblog.org, image uploaded by Daniela.*