Towards a mood sensitive integrated development environment to enhance the performance of programmers

A thesis submitted for the degree of Doctor of Philosophy

Iftikhar Ahmed Khan

School of Computing, Information Systems and Mathematics

Brunel University

December 2008

Abstract

The aim of the research was to analyze the possibility of developing an Integrated Development Environment (IDE) that could improve a programmer's performance by considering their current mood. Various experiments were conducted to study this idea. However, the impact of moods on programmer performance was initially examined in the literature. Based on this, a Cognitive Programming Task Model (CPTM) was developed showing that various cognitive functions and programming activities are interrelated. A second model derived from the literature, the Cognitive Mood Model (CMM), suggested that moods are also interrelated with various cognitive functions. Combining these two models indirectly suggests a relation between moods and programming tasks, which was presented as the Mood Programming Model (MPM).

As direct empirical support was lacking for this relation, two experiments were conducted to study the effect mood could have on performance in a debug task. Validated mood-inducing movie clips were used to induce specific moods along two-mood dimensions: valence and arousal. The first study was conducted online. The results showed that arousal is a significant factor when considering programmer performance whereas valence was found to have no significant effect. The second study was a continuation study to validate the findings from the first study within lab conditions. The results were not able to confirm the findings of the first experiment. The reasons for these findings were explained accordingly.

As mood was found to have an effect on a programmer's coding and debugging performance, this factor might be considered when developing a support system. The next step in the research was therefore to consider mood measuring in a non-interruptive way. The next two experiments were based around the hypothesis that "moods can be measured from the keyboard and mouse interaction of the computer user". In the first experiment an application was installed on participants' computers to record their key presses and mouse clicks in a log file. Their self reported moods in intervals of 20 minutes were also stored in the same file over an average period of eight days. Correlations between participants' self reported moods and their keyboard and mouse use revealed that it might be possible to measure moods of the some of the participants. The second experiment took place in the lab, where participants were asked to perform programming like tasks while listening to

mood inducing background music. Their moods were measured with a Galvanic Skin Response (GSR) meter whereas key presses and mouse clicks again were recorded in log files. The correlations between GSR measurements and keyboard and mouse interaction validated the findings of the experiment in the field that it might be possible to measure the mood of some users from their computer use. Analyzing participants' personality traits showed dutifulness and self discipline as indicators that a person's mood correlates with his/her interaction behaviour.

Considering that mood has an effect on programmer performance and that it might be possible to measure mood in a non-intrusive manner, the last question to focus on was whether a computer-generated intervention could change a programmer's mood and consequently improve their performance. In the final experiment programmers had to dry run algorithms for 16 minutes with the expectation that a level of boredom would set in. After this the video clip instructed them to participate in some physical exercises. Participants continued tracing algorithms for 8 minutes after the intervention. Results showed that the mood change after the intervention coincided with a programmers improved ability to provide the correct output of the algorithms.

Together these findings lay the foundation for developing an IDE that can measure the programmer mood in a non-intrusive way and make effective interventions to improve programmer performance.

Declaration

The following publications have been produced, directly and indirectly, from the research undertaken in this investigation.

Khan, I.A., Brinkman, W-P. and Hierons, R. M. (2006). Programmer's mood and their performance, 13th European Conference on Cognitive Ergonomics, September 20-22, 2006, ETH Zurich, Switzerland, pp. 123-124.

Khan, I.A., Brinkman, W-P. and Hierons, R.M. (2007). Moods and Programmers Performance, 22nd Psychology of programming Conference, July 02-06, 2007, Joensuu, Finland, pp. 3-16.

Khan, I.A., Brinkman, W-P. and Hierons, R.M. (2007). Mood Independent Programming, 14th European Conference on Cognitive Ergonomics, August 02-06, 2007, London, UK, pp. 269-272.

Khan, I.A., Brinkman, W-P., and Hierons, R.M. (2008). Toward a Mood Measuring Instrument from Interaction Behaviour, 23rd Conference on Psychology of Programming, 10-12 September 2008. Lancaster University, UK; pp. 155-169.

Khan, I.A., Brinkman, W-P., and Hierons, R.M. (2008). Measuring Personality from the Use of Keyboard and Mouse, 15th European Conference on Cognitive Ergonomics, 16-19 September 2008, Maderia, Portugal. Article No. 38.

Acknowledgements

First of all I am thankful to the God, who provided me the patience and passion to get this work done. I will also like to thank my parents and their continuous support all over the years.

Henry Adams said, “A teacher affects eternity; he can never tell where his\her influence stops”. Very same is true about Dr. Willem-Paul Brinkman. He was always available to correct me, guide me and to encourage me. He always taught me the tips and tricks to do research and helped toward my way to seek the scientific knowledge. He therefore will always influence me and his teachings will always guide me in my professional career.

If a reader is able to understand whatever is written in this thesis, than credit goes to Professor Robert Mark Hierons. Whenever he read a draft of the thesis he mentioned hundreds of mistakes and thus kept this thesis improving.

I will also like to thank my friends and other participants who helped me a lot by participating and motivating me in the course of my research.

Last and most important of all, I will like to thank the Higher Education Commission of Pakistan and the University of Engineering and Technology Peshawar for providing me a chance and funding to keep this research going.

Table of Contents

List of Figures

List of Tables

Chapter 1: Introduction

1.1 Overview

This chapter provides an introduction into the work presented in this thesis, describes the motivations behind the research and introduces the research questions. The chapter starts by discussing the motivation behind the research and then discusses research questions. The subsequent section explains the approach taken to investigate these research questions by presenting an overview of the thesis.

1.2 The Software Industry

The software industry is growing and globalizing very rapidly. More than ten years ago it already had an annual growth rate of 10-15% (Barr and Tessler, 1996). The software industry's key inputs are highly motivated, creative and enthusiastic individuals starting from high-level managers to analysts, software engineers, programmers, testers etc. About 236,000 people were working in the US software industry in 2005 and millions of others are working in this industry all over the world. The software industry's output depends on the skills and performance of Information Technology (IT) workers. Software development related skills might also be vital into many other industries (Barr and Tessler, 1996).

As software development involves various phases and each phase requires different cognitive skills, various people are involved in these phases because of specific skills. The task of converting analysis and design specifications into implement-able software is mainly the work of programmers. Programmers like everyone else make errors and may introduce errors into software applications. About 27% of system failures in various different companies are due to software defects according to a survey conducted by Gartner Dataquest (Rocco and Igou, 2001). There are various factors that contribute to software defects and researchers have various solutions, like better project management and more user involvement but psychological, social and organizational issues are often not addressed by these solutions (Smith and Keil, 2003). Errors seem inherent to human work, and they occur even in the most favourable conditions (Ko and Myers, 2003). For example, psychological issues like attention deficits can cause a decrease in work performance (Coetzer and Rixhmond, 2007), as lapses in attention are a known cause of accidents,

loss of time, efficiency deterrence, and personal productivity (Cheyne et al., 2006). If someone is not able to focus or is absent minded then this might lead to various errors (Reason, 1984). Similarly if programmers are less focused or are absent minded then this might lead to various errors in software while developing it. In other words to produce good error free software, being attentive seems to be important.

1.3 Programming, Mood and Emotion

Software development involves different cognitive and human aspects. Human performance and the quality of the resultant work are affected by different conditions and circumstances (Shneiderman, 1978). One of these factors is the mood and emotion of humans. For example, Norman (2004, p. 7) states that "Someone who is relaxed, happy, in a pleasant mood, is more creative …". In other words, people in the software industry when in a less pleasant mood might be unable to consider software problems from different angles and therefore be less able to find a viable solution. Therefore moods seem to be related to the quality of software and also to the performance of humans. Various psychological studies such as Tice et el., (2001), Chang & Wilson (2004), Lowther & Lane (2006) and Lane et al. (2005) suggest that moods and emotions have an effect on human performance. Moods are known to have an impact on reasoning which is an important cognitive element of most programming tasks (Jones, 2003). As the literature suggests moods as a factor that affects cognitive processes, it seems likely that programmers' performance is also affected by their mood, and potentially explaining lapses in their attention.

There are various psychological reasons for attention diversion. Larson et al. (1997) used the Cognitive Failure Questionnaire (CFQ) to study the relationship between mishaps and driving accidents and found a link between high CFQ scores and accidents, which were the result of distractibility, poor selective attention and mental error. This and several other studies (Cheyne et al., 2006; Danaher, 1980) show that cognitive aspects affect attention and the human focus. Research into cognition traditionally deals with concepts such as reasoning, perception, intelligence, learning, and various other properties that describe numerous capabilities of the human mind. Moods, emotions and feelings are also considered to be cognitive by psychologists like Schmitt (1969) and Izard et al. (1984). Relaxation, happiness and moods such as pleasantness can increase someone's creativity (Norman, 2004). In other words, this suggests that programmers in a less pleasant mood might be less able

to view software problems from various viewpoints and therefore less able to come up with alternatives.

Throughout our daily life we experience a series of different moods. We feel saddened by the death of a friend, whilst feel joyful and happy on the birth of a child or at a wedding. Similarly, in almost every task from flying a plane to performing an operation or programming a real time system, one would expect to feel a wide range of positive and negative feelings (Martin, 2001). These feelings and moods may disrupt our daily tasks. These disruptions along with interference in energy, sleep, and thinking are common, painful, and too often fatal (Jamison, 2003). It has been found that moods affect various different activities of people like creativity (Russ and Kaugars, 2001; Kaufman, 2003), memory tasks (Lewis and Critchley, 2003), reasoning (Chang and Wilson, 2004), behaviour (Kirchsteiger, Rigotti and Rustichini, 2006), cognitive processing (Rusting, 1998), information processing (Armitage, Conner and Norman, 1999), learning (Weiss, 2000; Ingleton, 1999), decision making (Gardner and Hill, 1990; Kirchsteiger, Rigotti and Rustichini, 2006), and performance (Chang and Wilson, 2004; Lowther and Lane, 2006; Lane et al, 2005).

1.4 Research Objectives

Computers are not mere tools to solve complex problems nowadays. Research is being conducted on the affective aspects of computers, so that they can communicate in a more human-like way with humans. Expectations are that this could help computer to be more helpful as well as more trustworthy than computers without this capability. Furthermore, a machine that could understand the mood of the user might be able to behave accordingly to relieve and help users to increase their performance. A substantial amount of work by Picard (1995) and her affective computing group and various other researchers are already striving for this goal. The motivation of the research presented in this thesis should be seen in this tradition. The work presented in this thesis, however, focuses specifically on a specific user group, i.e. software programmers.

Equipping the software development environment with the ability to measure the programmer's mood would allow this environment to detect conditions that might deteriorate the performance of the programmer. One step further would be that the supporting environment acts on this and provides programmers with suitable suggestions to alter their mood in order to improve their performance. With this vision

in mind, the work presented here lays the foundation for the design and development of such a support system. It focuses on three questions: (1) whether moods affect the programmer performance, (2) whether mood can be measured in a non-interruptive and un-obstructive manner, and (3) whether computer-generated intervention can help programmers to change their mood and improve their performance. The overall research question can therefore be formulated as: *Could a computer support system measure the programmer's moods in a non-interruptive and un-obstructive manner and generate effective interventions to alter the programmer's mood in order to improve their performance?*

1.5 Research Approach and Overview of Thesis

The research approach is a mixture of analytical reasoning, literature review, empirical experimentation and applying software engineering principles to develop software that supports the experiments or analyse the obtained data. The organisation of this thesis follows the research conducted. The research started with examining the literature and led to three models that could explain a possible relation between the programming task and programmer mood. Chapter 2 describes these analytical models which help to formulate the 3 hypotheses that were examined in the remaining part of thesis. These hypotheses were:

$H_{1:}$ Moods have an impact on programmers' performance.

$H_{2:}$ Mood can be measured from the computer user's interaction with keyboard and mouse.

$H_{3:}$ A development environment with an intervention or suggestion system could help to improve programmers' performance.

These three hypotheses support the overall hypothesis of this thesis, which is:

$H_{4:}$ An integrated development environment could measure the programmer's moods in a non-interruptive and un-obstructive manner and generate effective interventions to alter the programmer's mood in order to improve his\her performance.

The first empirical research work of the thesis is presented in Chapter 3. Two experiments are discussed that were conducted to test the first hypothesis. Once it was established that mood could affect programmer performance, work focussed on measuring mood in a non-interruptive and un-obstructive manner. After examining existing mood measures, two explorative empirical experiments were conducted to

examine whether keyboard and mouse interaction data correlated with data obtained from existing mood measures. Chapter 4 presents a field experiment in which interaction data was collected from a group of computer users over a period of eight days. Besides recording their behaviour, these users were asked to report their mood every 20 minutes. Chapter 5 presents an experiment conducted in the laboratory, where variation in mood was increased by using mood inducing background music while participants work on a series of programming tasks. Besides recording their keyboard and mouse interaction, the participant's mood was recorded by measuring their galvanic skin resistance, which is a psycho-physiological mood measure. The results of both experiments supported the second hypothesis.

The third hypothesis was also examined in a laboratory experiment as is discussed in Chapter 6. The results of this experiment suggest that a computer intervention in the form of a physical exercise film can help to change the mood of programmers as well as their performance when dry running a series of algorithms.

Chapter 7 reflects on the outcomes of the research and conclusions that can be drawn from it. The chapter presents the blueprint for an envisioned mood sensitive integrated development environment that can enhance the performance of programmers. Limitations of research and ideas of possible future research are also discussed in this chapter.

Chapter 2: Hypotheses of the study

2.1 Introduction

Chapter 1 explained the motivation of the project and set out the general research questions, this chapter establishes the theoretical foundation of the research. It puts forward three hypotheses that are empirically examined in the remaining chapters of this thesis. Together these hypotheses suggest that programmers can be supported in their work by using a computer-based development environment that can detect their mood and generate effective interventions to alter the programmers' mood and improve their performance.

The first hypothesis forms the basis of the research by suggesting that *programmers' mood can affect their programming performance*. It legitimizes potential research on a development environment that considers the programmers' mood. Limited direct research exists to support this hypothesis. Therefore support will be established by looking at previous work that has explored the relation between human task performance and mood in general and especially with programmers' activities. An indirect relation will be formed between moods and programmers' programming task performances to support the hypothesis. This indirect relation will be in the form of models, which are based on literature that suggests a relation between programming tasks and cognition as well as moods and cognition. The first section of this chapter therefore is devoted to examining the relation between moods and programming performance in the context of the tasks performed by programmers.

Although this study mainly focuses on moods and their impact on programming performance, the concept of emotion is often interchangeably used for the concept of moods. Therefore the chapter will briefly discuss moods and emotions and how they differ and relate to each other. Literature suggests moods are either one dimensional or multidimensional. As this study used a two-dimensional model of mood the next section is devoted to two-dimensional mood models. The position is taken that the two-dimensional mood model is a suitable model to be used in this study. Following on from the discussion on moods, emotions and their dimensions, the rest of the chapter is organized into three parts. The first part reviews the effect of moods on people's performance and one of the positions taken in the study that moods have an effect on programmers' performance. Mood might affect people in their

performance of different activities and here the relationship between their activities, performance and moods in general and the debugging activities of the programmers' in specific will be examined.

Measuring the mood of a programmer is a key element in this research. To be practically acceptable for programmers, a measurement instrument should be non-intrusive, and non-interruptive of their work. Traditional measuring instruments such as questionnaire, psychological or psycho-physiological measures (special gloves, special caps and helmets, special jackets, attached wires and instruments) seem inappropriate therefore. On the other hand various studies suggest a relationship between people's behaviours and their moods. A potential alternative therefore, is to study the programmers' interaction with computer applications and Integrated Development Environment (IDE) to determine their moods. This leads to the second hypothesis that states: *Moods can be measured from the computer user's interaction with keyboard and mouse*. The rationale behind this hypothesis is based on mood measurement using interaction behaviour and will be presented in the third section of this chapter.

As indicated before measuring mood is an aim of this research but this measurement should be non-intrusive and non-interruptive. In the second part of the chapter different methods used to measure moods will be reviewed. This review will be made in terms of interaction of a programmer with computer applications in general and IDE in specific. Advantages and disadvantages of these methods will be analyzed when used in the practical environment. There will be a discussion on the possibility of measuring computer users' and programmers' mood from their keyboard key patterns and mouse movements.

An Integrated Development Environment is computer software that traditionally aids computer programmers in software development. The last part of this chapter will look into different ways a programming environment can serve programmers. Programming environments (IDE) are aiding programmers to avoid mistakes in syntax as well as various logical errors. In other words, these IDE's are aiding programmers to improve their work. These programming environments might also help programmers to improve their performance by using knowledge about programmers' mood and then devising different suggestions such as 'take some rest' or 'coffee' or do some exercise etc. This forms the third hypothesis: *A development*

environment with an intervention or suggestion system could help to improve programmers' performance.

In the following sub-sections there will be a discussion on moods and emotions followed by a discussion on one-dimensional versus multidimensional mood models used for measuring moods and emotions.

2.2 Moods and Emotions

Kleinginna and Kleinginna (1981) in a review on definitions of emotions explained difficulties in defining emotions but agreed on most of the Plutchik's definitions. Pluctchik (1980) explained emotions in 10 postulates of which some are (p.80): "1. Emotions are aroused by external stimuli. 2. Emotional expression is typically directed toward the particular stimulus in the environment by which it has been aroused. 3. Emotions may be, but are not necessarily or usually, activated by a physiological state. 4. There are no 'natural' objects in the environment toward which emotional expression is directed. 5. An emotional state is induced after an object is seen or evaluated, and not before".

Various resent studies still (Gendolla, 2000; Cowie et al., 2001; Izard, 2007) used Plutchicks' definition of emotions. Parkinson et al., (1996) considered moods and emotions as affective states. He defined moods as (p.4) "Affects which refer to mental states involving evaluative feelings, in other words psychological conditions when the person feels good or bad, and either likes or dislikes, what is happening".

Moods are affective states that last a little longer and are weaker states of uncertain origin, whereas emotions last for a short time are more intense, and have a clear object or cause (Fridja, 1993). Moods may last from a few minutes to days whereas emotions come and go in minutes and sometimes even in seconds (Ekman, 2003). Most psychologists consider mood and emotion as the same entity. For example, Dow (1992) considers mood as a type of emotion. Most studies also use the terms mood and emotions interchangeably without distinguishing between them (for example Kirchsteiger, Rigotti and Rustichini, 2006; Kaufmann and Vosburg, 1997). According to Ekman (2003) mood is a continuous emotional state, whereby moods differ from emotions based on the time they are retained and on the variations in the intensity.

Further, given an emotion, it is often possible to identify the events that caused it, however, the event responsible for causing a mood is rarely known. Moods can just

happen, for instance; one can wake up in the morning with some particular mood. Zimmermann et al. (2003) stated that moods might result in preconceived emotions as people may not be aware of their moods until attention is drawn toward them. Shorter events as with emotions cannot be easily distinguished from other short events. For example, while watching TV a user could get distracted by the telephone and at the same time is amazed at what he sees on the screen. Both events can cause emotions and so it is difficult to know the exact cause of emotion (Zimmermann et al, 2002).

In conclusion moods and emotions are considered as the same with few differences in their time and intensity. People are aware of events that caused emotions but usually they are not certain of the events that caused their moods. Both moods and emotions have impacts on several aspects of the people's lives (see section 2.2). Moods last longer so they might affect aspects related to programming continually but with less intensity. This research focuses mainly on the impact of moods rather than emotions because of the several reasons i.e: Most of the time people are not aware of the events that caused their mood, and they might have less control to overcome these situations e.g. blue feelings (Fiedler, 2001). On the other hand, an event that caused emotion is known to them and so they might have more control in overcoming bad emotions like anger. The basic aim of this research is to design affect aware Integrated Development Environment (IDE) that can aid programmers to improve their performance. Therefore an IDE considering emotion could be too interruptive, as emotions live shortly with more intensity and therefore could become annoying. As moods last longer, their measurement and response from the environment over the mentioned period could prove to be more useful.

2.3 One-Dimensional Versus Multi-Dimensional Mood Model

The degree of happiness or sadness is often labeled with the term *valence* whereas a subjective state of feeling activated or deactivated is often refer to with *arousal* (Sanchez et al., 2005). The representations of valence and arousal on orthogonal X- and Y-axes form a two-dimensional model of mood as proposed by Lang (1995). Figure 2.1 explains some affective states and their correspondence to some valence-arousal combinations. For example, the "Rejoice" could be a pleasant and high arousal state while "gloomy" could be an unpleasant and low arousal state. Similarly, "terrified" could be an unpleasant but high arousal state whereas "soothing" could be a pleasant but low arousal state (Sanchez et al., 2005).

Some researchers believe moods to be one-dimensional and trace it back to Charles Darwin's theories (Darwin, 1965). There are various other researchers who consider moods as multidimensional such as Dienstbier (1984). Russell (1980) suggested a spatial model to represent affective states and indicated that affective states fall in a circle with the following angling order (p. 1161): "pleasure (0°), excitement (45°), arousal (90°), distress (135°), displeasure (180°), depression (225°), sleepiness (270°), and relaxation (315°)". A detailed discussion about one-dimensional versus multidimensional moods is provided by Remington and Fabrigar (1995). The use of two-dimensional models to examine a mood is gaining greater acceptance (Thayer, Newman and McClain, 1994). Various researchers consider mood as, at least, a two-dimensional construct, with an evaluation dimension (valence or pleasure) and an arousal dimension. For example, Almagor and Ben-Porath (1989) divided mood into two dimensions: an arousal or activation dimension and a hedonic tone or pleasantness dimension. Others like Morris (1995) prefer adding a third dimension to express the dominance (controlled, uncontrolled). This study used only a two-dimensional model, with the two dimensions valence and arousal as these two dimensions are primary and are used in most emotional judgments (Bradley and Lang, 1994).

Researchers with a nominal view of moods and emotions, whereby emotions are described in basic states, believe that general arousal may have some minor affect on intensity or duration of emotion. On the other hand, researchers who consider moods and emotions as multidimensional believe that emotional experience is formed by the combination of arousal underlying this experience and certain cognitive assessment of the meaning of the arousal (Dienstbier, 1984). Moods and emotions in multidimensional view are often measured using self-reports. According to Lang (1980) self-reports are more reliable applying a multidimensional view of measuring moods than a single-dimensional view. This study will mainly use self-reported measures with multidimensional approach (valence, arousal) because of the following reasons: 1) Most of the research used multidimensional emotions rather than the basic emotions or one dimension of emotions (Picard, 1997), 2) Self-reports are more reliable with multidimensions than with the one dimension categories such as anger, fear etc (Lang, 1980).

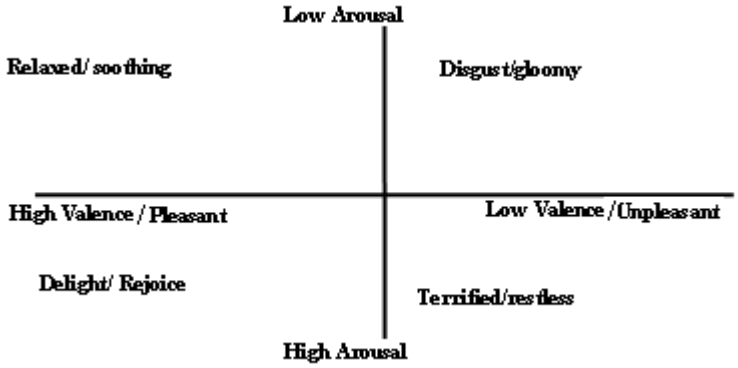


Figure 2.1: *Valence-Arousal model with some examples of some specific moods or emotions*

For self reported measurement of valence and arousal on the two dimensional model SAM (Self Assessment Manikin) devised by Lang (1980) is used in this study. Figure 2.1 shows use of this scale along X-axis and Y-axis. The Horizontal X - axis scale starts from 1 (High valence) to 9 (Low valence). Similarly the Vertical Y-axis scale starts from 1 (High Arousal) to 9 (Low Arousal).

2.4 Moods and Programmers' Performance

The basic objective of this section is to establish indirect support for the first hypothesis of the study that states that "Moods have an impact on programmers' performance". This support will be provided with the help of relevant literature as well as by developing models indicating indirect support for the hypothesis. The proposed models will be developed using relevant research in cognitive sciences, on programming tasks and moods. The purpose of these models is to explain the relation between cognition and programming tasks, the relation between moods and cognition, and thus indirectly the relation between moods and programming tasks.

The section will start with emphasising the importance of moods in daily life. This will be followed by a discussion on moods and their impact on programming, cognitive tasks as well as on programmers' performance. As moods are considered to affect cognition and as programming involves various cognitive processes, the next part will discuss the psychology of programming and the impact of different cognitive elements on programming tasks. The impact of cognitive elements will be extended to debugging and program comprehension in the next part that will be followed by the formulation of a programming, cognition mood model.

Moods and emotions are important to understand the world and their absence can cause distress and disaster. For example, Damasio (1995) found that people who lose emotional ability often take disastrous decisions as they do not feel happy or sad about events. Moods also aid to understand circumstances as well as they can alert us to dangers. For instance, an emotion like 'fear' enables someone to realize the critical situation, which in turn conveys one's emotional and physical state to the others. Moods can also help us to overcome problems and to gain advantage of useful situations. Disorders in mood are common, painful and could prove to be fatal (Jamison, 2004). Moods are also known to affect thinking, narrow down alternatives and make it difficult to control things and therefore might affect performance indirectly while someone is at work (Ekman, 2003). Tice et al., (2001) argued that people under emotional stress may give priority to feel better and in the process may ignore long range goals and thus in turn cause decrease in performance.

Research shows that moods affect performance but a question of importance becomes what could be the reason for mood? In daily life there are various causes that could explain someone's mood. For example, an individual is likely to feel sad on someone's death and is likely to feel happy on marriage or birthday. Similarly, flying a plane, performing an operation, playing finals and presenting are some of the tasks in which people could expect a wide range of positive and negative feelings (Martin, 2001). Similarly, overwork or boring work may lead to under arousal and an interesting and exciting task can result in over arousal. People in sad mood try to repair their bad moods. There is a threat to performance in the tasks if people are in bad moods or seek to repair bad moods (Erber and Erber, 2001).

According to Yerkes-Dodson's (1908) Inverted U Hypothesis an increase in arousal cause an increase in performance until a certain level beyond which arousal increase cause degradation of performance. Various other studies like Chang & Wilson (2004), Lowther & Lane (2006) and Lane et al. (2005) also found that moods have significant impact on performance. Moods also affect activities like creativity (Russ and Kaugars, 2001; Kaufmann, 2003), memory tasks (Lewis, Critchely, 2003), reasoning (Chang and Wilson, 2004), behaviour (Kirchsteiger, Rigotti, et al, 2005), cognitive processing (Rusting, 1998), information processing (Armitage, Conner and Norman, 1999), learning (Weiss, 2000; Ingleton, 1999) and decision making (Gardner and Hill, 2004).

Research suggests that moods affect reasoning, which is an important cognitive process for programming (Jones, 2003). Jones argued that programming tasks alone are not sufficient for a valid computer program without reasoning. Impact of moods on reasoning is also evident in other studies (Blagrove and Akehurst, 2001; Zarinpoush, Cooper and Moylan, 2000). Furthermore various researchers like Damasio (1995) found emotions/moods and reasoning to be quite interrelated. Therefore, an impact of moods on reasoning might directly or indirectly have an effect on a programmer's performance.

As moods might have an impact on programmers' performance, there is a need to view mood as an important human factor affecting performance. Traditionally, software psychology focuses on human factors such as ease of use, simplicity in learning, improved reliability, reduced error frequency and enhanced user satisfaction (Rugaber and Tisdale, 1994). The personality of a programmer can also be factor. Iron (1982) stated that by considering personality in cognitive operations of programming, it can be easier to understand cognitive abilities and their utilization to improve the performance of programming tasks. Programming or coding is only one of several tasks in the software development process which also includes: designing, debugging and testing etc. Work has also been done towards understanding the cognition behind the programming task. Examples include Shneiderman and Mayer's (1979) work. They proposed a model of programmers' behaviour to explain basic programming tasks in terms of cognition. These tasks include:

1) Composition: writing a program
2) Comprehension: understanding a given problem
3) Debugging: finding errors in a given program
4) Modification: altering a given program to fit a new task
5) Learning: acquiring new programming skills and knowledge

Debugging as mention above is a task that focuses on finding and correcting errors in a program. Debugging is a difficult task and requires large part of a programmer's effort (Brooks, 1975). It is a complex but important activity and is a combined process of testing and code correction (Shaochun and Rajlich, 2004). Debugging has received some attention from software psychology (Rugaber and Tisdale, 1994) but there is still a lack of research in the scope of cognitive abilities and debugging performance. Therefore the main focus of this study will be on debugging. Shaochun and Rajlich (2004) identified three levels of knowledge for a

programmer and these are helpful in understanding effective debugging. They are: program knowledge, domain knowledge and specific bug knowledge. Debugging therefore requires skills to utilize this stored knowledge from memory on all the three levels which in some sense can also be said for program comprehension.

Similar to the three levels of knowledge required for debugging, Shneiderman (1980) also divided program comprehension into three levels, i.e. low level comprehension of the function of each line of code, mid level comprehension of the structure of the algorithm & data and high level comprehension of overall program function. He organized these three levels into the form of a model. According to Shneiderman, programmers store the multilevel knowledge in their long-term memory and extract and use this knowledge to convert program code into some internal semantic form. In addition to memory, other well established cognitive factors are also known to have relationship with different programming variables. For example, Iron (1982) found that induction and syllogistic reasoning are strongly related with program composition and its production. He also found that spatial scanning and debugging are interrelated.

Similarly Lui and Chan (2004) proposed a Cognitive Programming Model (CPM) based on the studies of psychology of programming and cognitive models for learning. The purpose of the model is to describe how problems in natural language are transformed into symbolic executable programs by programmers. CPM work on six key areas: 1) Definition, 2) Representation, 3) Model, 4) Schema 5) Algorithm and 6) Code. Lui and Chan (2004) found that repeated tasks (repeating a similar task several times) initiate cognitive processes of learning and memorization which in turn cause reduction in development time. They found that in the start of the repeated tasks programmers applied different reasoning and logic but after learning and memorization, produced similar solutions in terms of program logic. Xu (2005) worked toward creating program comprehension model based on Bloom's (1956) six learning levels. He considered cognitive activities absorption, denial, reorganization and expulsion at six learning levels of Bloom taxonomy.

The above literature suggests that various cognitive abilities are of importance for different programming tasks. For example Shneiderman (1980) related program comprehension with memorization. Iron (1982) found relation between induction and syllogistic reasoning. He also found a strong relationship of induction with program composition and its production. Next, Iron also found that spatial scanning and

debugging are interrelated. Similarly Lui and Chan (2004) related learning and memorization with efficient task processing as well as a mean for the production of various solutions in repeated tasks. Therefore the Programming Tasks (PTs) which literature suggests that are related with Cognitive Tasks (CTs) can be mapped with each other in a two-dimensional model as illustrated below in Figure 2.2.

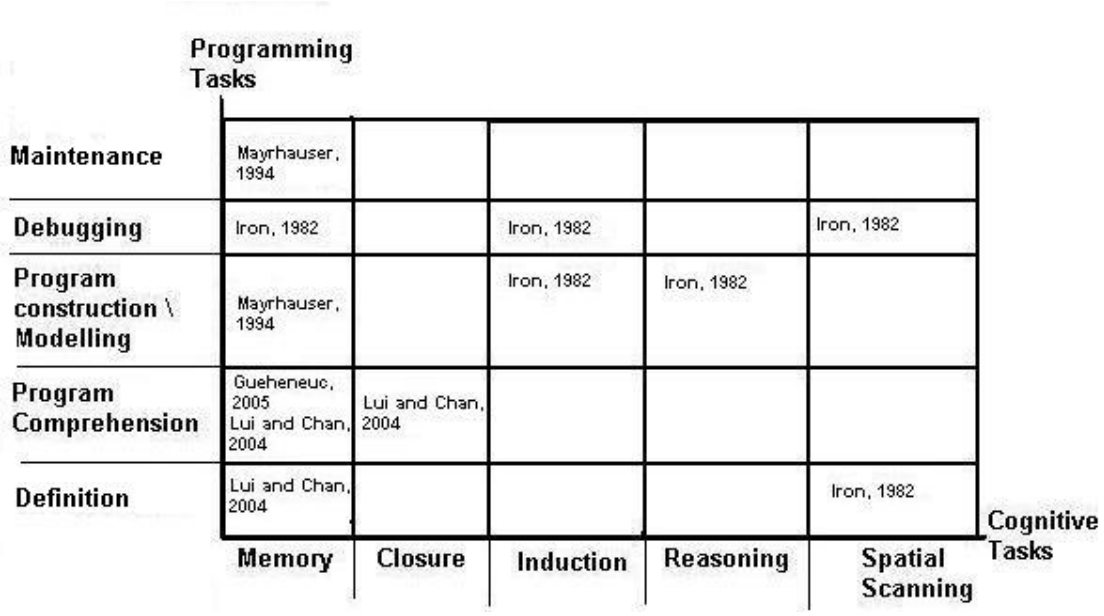


Figure 2.2: *A two-dimensional CPTM (Cognitive Programming Task Model). Some cells show examples from the literature that address the specific cross-section of programming task and cognitive element. Note: this table does not present an exhaustive overview of the literature, instead it just provide some examples from the literature for some PT-CT cells.*

The programming tasks as well as cognitive elements used in the model above were based on the literature like Mayrhauser (1994), Lui & Chan (2004), Iron (1982) and Gueheneuc 2005. This CPTM (Cognitive Programming Task Model) model represents only a few of the important programming tasks as well as some of the cognitive tasks derived from the literature indicated earlier in this paragraph. The programming tasks used in the model (Figure 2.2) along the y-axis are parts of the normal software development life cycle as discussed by Pressman (2005). The programming tasks include:

1. *Definition*; Elementary term required to perceive problems.
2. *Program Comprehension/Program Understanding*; Ability to read and to understand a computer program to plan, design and implement modifications in it.

3. *Program Construction / Modelling*; A stepwise transformation of real world
4. relations into computer language coding
5. *Debugging*: An activity of finding and correcting errors within a computer program
6. *Maintenance*; Modification of the software product to correct faults and to improve performance of the software product.

Cognitive tasks used in the model include:

1. *Memory*; An organism's ability to store, retain and subsequently retrieve information.
2. *Closure*; The ability to 'hold in mind' a particular percept (configuration) and find it embedded in distracting material
3. *Induction*; A Process in software development in which developers deal with elementary terms that are required to perceive problems
4. *Reasoning*; cognitive process of looking for reasons for beliefs, conclusions, actions or feelings
5. *Spatial Scanning*; The ability to quickly survey a complex field to find a particular combination representing a pathway through the field

Paragraphs above relate programming tasks and cognitive elements. Literature discussed earlier in this section also suggests that various moods (valence and arousal) have impact on cognitive abilities. For example with respect of human memory MacWhinney et al. (1992) as well as Lewis & Critchely (2003) found that moods dimensions valence and arousal are related to better memory performance. Mujica-Parodi et al. (2004) found that arousal had a significant impact on cognitive performance. Similarly Chang and Wilson (2004) found that valence and arousal affect reasoning capabilities. Zheng (2008) argued with the help of literature that valence and arousal are related to implicit and explicit memory. Similarly Delplangue et al. (2006) found that emotional content of the stimulus may modulate the reorientation of attention and the subsequent updating process in a specific way. Therefore, it can be argued on the bases of literature that moods (valence, Arousal) and cognitive tasks (CTs) are related and can be mapped to each other in a two-dimensional model as illustrated in Figure 2.3. This two-dimensional model suggests the possibility of mapping moods (valence, Arousal) with cognitive tasks, which could help to classify existing literature and direct future research.

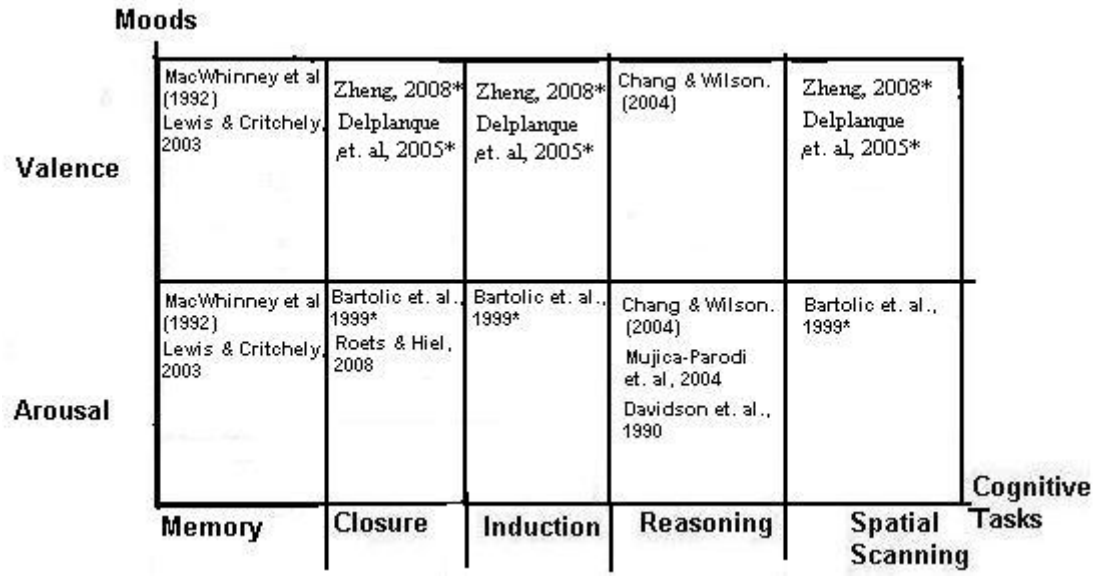


Figure 2.3: *A Cognitive Mood Model (CMM), showing moods (valence, arousal) impact on cognitive tasks (CTs). Note: * indicates that a specific study does not focus exclusively on only valence or only arousal, but on mood in general***.**

Considering the relation of cognitive elements with programming tasks as illustrated in (Figure 2.2) and that of moods with cognitive tasks from the literature (Figure 2.3) it can be argued that moods are related with programming tasks. Thus another two dimensional model with one axis representing moods (valence, arousal) and other axis representing programming tasks can be formulated to form the model represented in Figure 2.4. This MPM Mood Programming Model) model supports the first hypothesis H_1 by indicating a relation between moods and programming tasks. Mood dimension is further divided into valence and arousal. The symbol in the cell represents the impact of one dimension on the other dimension, formulated as an indirect relationship from the two earlier models. For example Lewis and Critchley (2003) presented a case to show that valence (positive or negative) has an effect on memory. Similarly Shneiderman (1980) found that memory plays a vital role in programming tasks like definition (D), program comprehension & understanding (PC), program construction & modelling (PM) and debugging (DG). As valence effect memory and memory effect programming tasks (D, PC, PM, DG), there might be an indirect relation between programming tasks (D, PC, PM, DG) and valence. Thus a symbol in the respective cell shows indirect relationship between the two dimensions. The CPTM model presented earlier also shows that all cognitive tasks mentioned here (induction, closure, memory, induction and reasoning) are related with debugging,

thus indicating impact of moods on debugging. This formed an additional reason to consider debugging as main focus for this study.

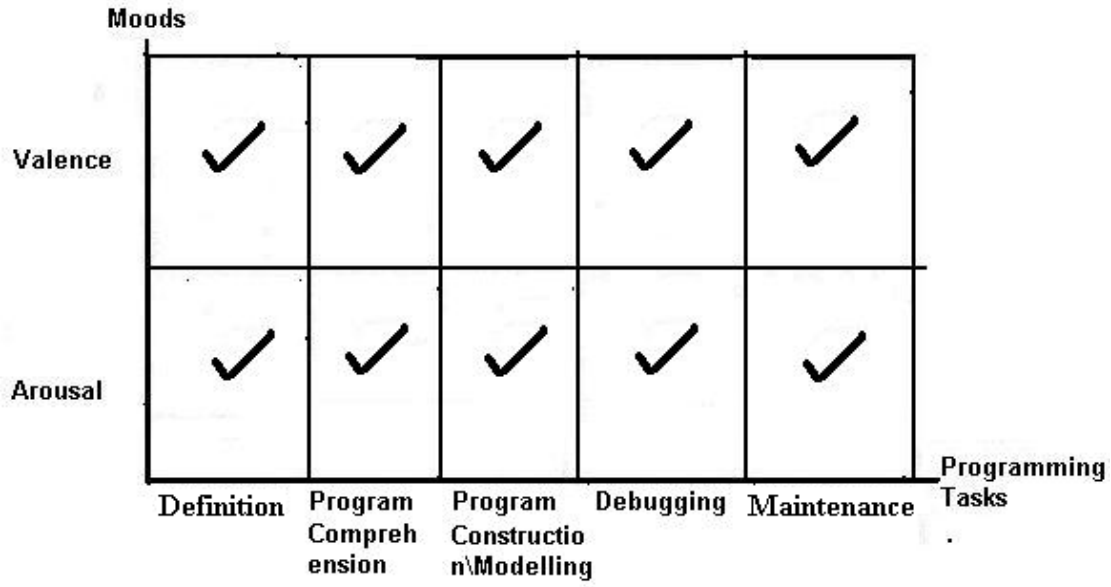


Figure 2.4: *A Mood Programming Model (MPM) based on models in Figure 2.2 and Figure 2.3 showing moods might have an impact on Programming Tasks. The relations are represented by ticks in the respective cell.*

In conclusion, earlier sections established that moods might have an impact on various cognitive abilities. Furthermore cognitive abilities play an important part in programming skills and in programming performance. Therefore the literature provides indirect support for a relationship between mood as a factor affecting various cognitive abilities and thus programming tasks and programmers' performance. This leads to refined formulation of the first hypothesis for this research that states that "*Programmers' moods affect their debug performance in respect of finding and correcting problems"*.

2.5 Measuring Moods from Behaviour

In the previous section it was established with the help of the literature that moods might have an impact on programmers' programming tasks and hence on their performance. However, in order to help programmers to improve their performance, there is a need to measure their mood. The objective of this section is to consider the possibility of measuring programmers' and computer users' mood from their use of a keyboard and a mouse.

As use of keyboard and mouse is a specific kind of behaviour, this section commences with the possibility of measuring moods from programmers' and

computer users' behaviour. Therefore, in the second part of this section the effect of moods on general behaviour will be discussed, followed by the discussion on the effects moods have on computer related behaviour in the third part. The fourth part will be a discussion on different ways to report moods and to measure moods along with their advantages and disadvantages. What could be used in order to have non-interruptive and non obtrusive measurement of moods in a computer related environment? This concern will be discussed in the last part of this section. In addition, the second hypothesis of the study will also be devised in the last part.

Mood might have an effect on behaviour and there are various studies to support this idea. For example, Armitage, Conner and Norman (1999) showed that positive moods promote risky decision making and heuristic strategies. Similarly, they found that people in negative moods focus more on specific outcomes and other attributes associated with behaviour when making a decision. Bohner et al. (1992) showed that people were uninfluenced by the content as well as context information when in a good mood but used both types of information while in a bad mood. Dow (1992) argued that moods are diffusely related to behaviour.

As reports in the literature show that different moods/emotions might be a reasons for various behaviours, this research is particularly interested in measuring moods based on the behaviour. A considerable amount of research has focused on computer systems that can recognize user's emotions and can adapt them accordingly in order to improve social presence (Nasoz et. al, 2003). There are also continuing attempts to measure moods from behaviour.

There are various methods devised for measuring moods/emotion from behaviour including both self-reporting and detections and measurements by the use of software. Methods like SAM (Self Assessment Manikin) and EPI (Emotion Profile Index) are for manual measurement of moods and are widely used in different researches that require measuring moods and emotions of the subjects. These measurements are also called self-reporting measurements (Swindells, 2006; Poels, 2006).

Poels (2006) divided self-reporting measurement scales into verbal self-reporting scales, visual self-reporting scales and moment-to-moment ratings. Some commonly used examples of verbal self-report scales are Plutchik's emotion profile index (EPI) and Izzard's differential emotion scale. Visual self-report scales include SAM (Self Assessment manikin), PrEMO (Product Emotion Measurement

Instrument). Moment to moment ratings includes warmth monitor which is shown to provide reliable measure of warmth.

Verbal mood measuring methods have various advantages such as being simple and easy to quickly investigate large scale emotional responses. Similarly, these methods also have various limitations such as scales containing a large list of emotion adjectives that might be cumbersome and a cause of fatigue in respondents (Poels, 2006). However, these moods/emotions measuring methods might not be suitable for measuring moods/emotions of the programmers due to their distracting nature and requirement to rate moods constantly (Zimmerman et al., 2003).

Visual reporting or autonomic measures include various techniques like facial expressions, skin conductance and heart rate (Nasoz, 2003; Backs, 2000). Measurements of facial expressions might not be suitable for measuring moods of programmers at work because they rely on the visible facial expression. Measuring moods from facial expression is not always a good choice because affective states are internal and involve cognitive thoughts as well as physical changes (Picard, 1997). Internal cognitive thoughts may not always impact facial expressions.

Physiological measures such as skin conductance, heart rate measurement etc require equipment to be attached to different body parts and this may cause disturbance to a programmers working in a professional environment. (Fahrenberg and Wientjes, 2000; Zimmermann et al., 2003). The mentoring of keyboard and mouse use might reduce these overheads as these are the most basic input devices used by computer users. Computer users are familiar with these devices and there are no risks of external device disruptions such as the loss of concentration due to attached equipment and devices. The use of keyboard and mouse is also cheap as they are available with almost every computer (Zimmerman et al., 2003).

Keyboard and mouse have already been used in various studies such as in personality (Mikkelsen et al., 2007) and usability (Coutaz et al., 1995) domains. For instance Meunier (1996) found that personality and gender had a significant effect on keyboard control. However there is limited literature on the possibility of measurement of moods by the user's use of keyboard and mouse. There are some examples where researchers strived for this, like Mahr et al. (2005) used mouse motions to detect emotions with some significant correlations in his thesis. Another example is Zimmerman et al. (2003) who designed experiments to store keyboard keystrokes and mouse movements in log files and later intended to find correlations of

these events with affective state. Zimmerman (2003) results might not be published yet but the study described in this thesis was set up to conduct the empirical examination on the possibility of mood measurement from computer users' keyboard and mouse usage.

Although there is some research, which utilized keyboard and mouse as indicated above, the direct use of keyboard and mouse as a mood measurement instrument was simply not found. However research suggests that keyboard and mouse behaviour is influenced by social factors and work environments (Peres et al., 2004). This might lead us to the second hypothesis of the study that indicates that "*Moods can be measured from the computer user's interaction with keyboard and mouse"*.

2.6 IDE to Improve Performance in Context of Moods

Previous sections looked at the impact moods may have on a programmer coding and debugging performance as well as how programmer and computer user mood can be measured from their interaction with a keyboard and a mouse without interrupting them. Implementing a mood-measuring tool, however, requires a software environment. As almost every programmer uses some kind of Integrated Development Environment (IDE) for software development, an IDE could be a suitable platform for such mood-measuring instrument.

This section will discuss programming environments, also called IDEs, and their role towards assisting programmers in their programming tasks. The first part of this section of this chapter explains programming as complex cognitive tasks and therefore needs to divide them into subtasks as well as to provide tools for each task in order to reduce its complexity. The second part looks at IDEs and some of the tools embedded in IDEs and ways in which they help programmers will be explained by looking at the underlying cognitive functions they support. The third part will discuss the possibility of an affect recognizing tool embedded in an IDE and how it can implement an intervention system to aid programmers. The last part of the section will talk about different interventions that can be used to change moods of people. The position will be taken that these interventions can be implemented in an IDE and thus can serve to improve performance of programmers.

Programming is a complex cognitive activity with limited time to complete the tasks (Yildiz, 2007). Therefore, it is suitable to divide programming tasks into various

subtasks as well as to introduce helper modules to aid programmers. Such a tool which holds different helper tools together at one place is called an Integrated Development Environment (IDE). An IDE improve performance by targeting various problems programmers may face during development. Pane and Myers (2000) identified that programmers face difficulties in memorizing because of increasing vocabulary of computer languages as well huge syntax requirements. They also identified that mental plans of the programmers are not appropriately mapped into programming languages. Therefore Pane and Myers identified that there is a need for tools that can make relevant information available and accessible as well as provide support transforming mental plans into a specific computer language. As memorizing, expressing and representing mental plans are cognitive function, there is a need for cognition support tools.

As explained earlier with the help of Figure 2.2 programming tasks and cognitive elements are directly related. There is some research conducted toward developing software that could provide cognitive support. For example Green and Petre (1996) studied the possibility of designing a development environment that should be able to adapt according to mental representations. They laid a basis for development of such environments by developing a framework they called cognitive dimensions framework.

CTAT (Cognitive Tutor Authoring Tools) is software used to author intelligent tutor behaviour. This tool is now being used in research related with development, learning, and complex problem solving. For example Aleven et al. (2006) used CTAT in preliminary small-scale controlled experiments involving basic cognitive tutor development tasks. Some of the modern IDEs are also providing tools for cognitive support in order to help a programmer to increase their productivity. For example Koedinger & Aleven (2004) found preliminary evidence that CTAT software substantially reduce the time needed to build and test cognitive models and tutors. Support tools provided by IDEs could be mapped onto two dimensions representing programming tasks on horizontal axis and cognitive element on vertical axis. The conjunction of programming tasks and cognitive element form a cell displaying the relevant tool present in the IDE to support programmer in their work. This model is illustrated in the Figure 2.5.

Spatial Scanning	Colour Syntax Error locator	Colour Syntax	Colour Syntax	Colour Syntax
Induction				
Closure		Intellisence		
Reasoning	Error Identification Cause of error			
Memory	Error Identification	Intellisence	Colour Syntax Indention	Class browser Explorer Source Safe
	Debugging	**Construction**	**Comprehension**	**Maintenance**

Figure 2.5: *IDE cognitive support model, including some examples of support tools that support specific cognitive element of the programming task.*

The model illustrated above uses four out of five programming tasks discussed earlier in section 2.2. The definition task is left out because it is part of software requirements and specifications and may not be actively related with development environments. As indicated in the figure above, each cell represents a possibility of where support tools could help programming with various cognitive element of programming. For example the cell at the junction of Memory and Debugging indicates that a debugger can support programmers by displaying error information thus eliminating the need to remember what type of error they made. Similarly tools can also help to decrease memory load while creating programs in IDEs by highlighting the error terms. This model can help to categories existing support tools and suggest the development of future support tools.

Various modern IDEs like Visual Studio and Eclipse try to avoid overloading the user memory, by providing context sensitive help. This help automatically appears at the point of using a class or object or other language syntax or structure and enables programmers to complete the word as well as provide information about the type of object. Figure 2.6 shows an example of such IDE with different colour words. The words in blue are an indication of keywords, whereas words in cyan indicate classes.

The popup helps display all methods, properties and events related with a class thus removing the need to memorize all of them. At the same time debug window at the bottom of the image instantly provides help on the errors and can take the user to the error on a click, thus enabling fast access for fast development.

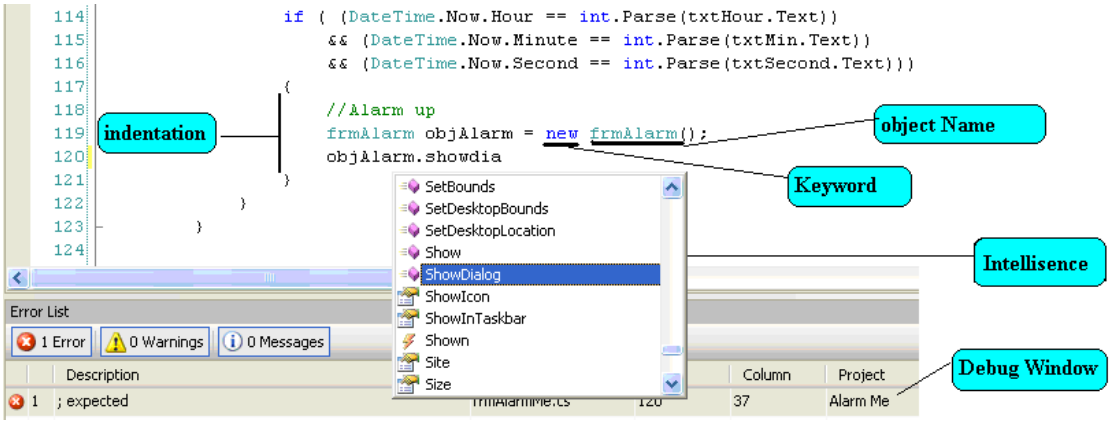


Figure 2.6: *An example of cognitive support provided by IDEs in various programming tasks. This is a screen shot of Visual Studio express edition editor***.**

In addition to providing cognitive support to programmers it is also important to recognise that IDEs are used in a much larger support context. For example, IDEs are now being used for concepts like requirements traceability e.g. Bell-Northern Research's IDE (Macfarlane and Reilly, 1995), for automatic design and implementation of Fuzzy Logic Controllers (FLC) (Kim and Cho, 1997) and for simulation and modelling (Kennedy and Raistrick, 2000).

Although the IDEs are helping programmers to improve their performance by reducing their cognitive load, there are still various cognitive processes such as reasoning and logic for which help does not seems to exist. Affects are also one of the cognitive processes for which IDEs do not seem to provide support for. To improve programmers' performance the IDE concept should also include affective support facilities. Related work is being done in the field of affective computing. However it is not directly related with development environments but to computers in general. There is a need to take a step forward toward developing or modifying existing development environments that not only measure moods but should also be able to provide some suggestions to improve performance. These suggestions could be in the form of appropriate message as an intervention to change programmers' mood. These interventions could include for example the advice to take coffee (Watters et al.,

1997) or to do some exercise (Jeanick, 2002) to increase the performance of the programmer.

This concept of improving performance after introducing an intervention is not new and is well tested. For example O'Hara and Payne (1999) introduced change in the cost structure of operations as an intervention to help expose aspects of cognitive support. They observed that raising the delay for compilation would cause users to make more systematic passes through error lists. Jarvis (1992) found significant improvement in simple reaction time, choice reaction time, incidental verbal memory and visio-spatial reasoning after the intake of coffee. Similarly Smith et al. (1993) found significant effects of intake of coffee on alertness and performance. Robelin and Rogers (1998) also found significant effect of coffee on energetic mood as well as on psychometric performance. Shepard (1997) found that children after substantial physical activity within their curricular time increased their learning ability. He also found physical activity related with increased cerebral blood flow, higher arousal changes in hormone level, changes in body built etc.

Various studies also utilized computers as a means to intervene and found positive impact on people's behaviour and on their performance. For example Wade et al. (2006) used computer-delivered interventions group and a control group of women to examine psychological health differences. They found significant improvement in self-esteem, social support and empowerment in computer-delivered intervention group. Brug et al. (1996) found that tailor made computer interventions according to participants' individual characteristics significantly caused decreased fat consumption as well as intentions and attitude toward decreasing fat consumption. Similarly Dijkstra & De Vries (1999) researched on developing computer generated tailored interventions for self-help. All of the above studies demonstrate the possibility for computers and their software being involved in successful intervention toward changing people's behaviours and improving their performance.

Various studies also targeted the use of computer in mood regulation and improvement of performance. For example Silvestrini & Gendolla (2007) used computer to display film on the computer screen to change the mood of participants. The above literature suggested that if computer can provide such help and suggestions, and enable programmers to take some time out and to do what was suggested, their performance might improve.

In conclusions this research targets programmer debugging tasks and the debugger tool the mood measuring and intervention. From the discussion above we can conclude that it might prove useful to design such a tool for integrated development environments that can help programmers to improve their performance. This refined the third main hypothesis of the study as: *A development environment with an intervention and suggestion could help to improve programmers' performance.*

2.7. Conclusions

In this chapter it was argued that moods might have an impact on programmers' debugging performance. This idea was put forward in the first hypothesis of the study. Studies suggesting that moods have an impact on different activities of people were discussed. Indirect support was found for a relation between moods and programmers' performance by showing a direct relation between programming tasks and cognitive elements and between moods and cognitive elements. These relations were presented in the form of models. However, to help programmers in the context of moods there is a need to measure their moods first. This mood measuring process should be non obtrusive and non interruptive. Therefore in the third section of this chapter, literature suggesting the possibility of measuring mood from programmers' and computer users' behaviour on keyboard and mouse was discussed. To practically implement the mood-measuring instrument, a software environment is required. The last section of the chapter discussed literature related to development environments and the cognitive support they provide was discussed. This information was utilized to analyze the possibility of mood recognizing Integrated Development Environments capable of helping programmers to improve their performance.

The next chapters will explain the empirical examination of these three main hypotheses. Chapter three will explain the first experiment that was carried on in the support of the first hypothesis. Chapter four and five will present the empirical studies that were conducted to test the second hypothesis of the study. Chapter six will discuss the experimentation that explained the third hypothesis of the study. The last chapter concludes as well as discuss limitations and future work.

Chapter 3: Impact of Moods on Programmers' Performance[1]

3.1 Introduction

This chapter tests the first hypothesis of this research stating *"Programmers' moods affect their debug performance in respect of finding and solving errors in the code"* and will be represented as H_1. There is already a large amount of literature that shows that mood affects various cognitive processes in humans as discussed in detail in chapter 2. In addition indirect support was established for a relation between moods and programming tasks. However little empirical work has been reported about effect moods have on the programmers' debugging performance. This chapter therefore aims to examine this relationship empirically. This chapter also forms the motivation to the study of hypothesis H_2 and H_3.

A detailed literature review and the rationale for this experiment were presented in Chapter 2. This included basic frameworks termed as models in this study that were derived from the literature. The CPTM model (Figure 2.2) indicates that there is a direct relationship between cognitive tasks and programming tasks. The CMM model (Figure 2.3) indicates a direct relationship between mood (valence, arousal) and cognitive tasks. This led to the MPM model (Figure 2.4) that forms a relationship between moods and programming tasks. Following this indirect support that moods affect programmers, this chapter presents work that directly examined this effect. In order to do this, two studies were conducted. The first study was conducted over the internet and focused both on the valence and arousal mood-dimension. The second study focussed primarily on the arousal dimension and was conducted offline. In the first study, programmers were asked to complete a debugging test after watching a video clip that was selected to induce a specific mood. Before this experiment, a validation study was conducted to select these mood-inducing videos, and to develop a debugging test. In the second experiment, the setup was very much similar to the first experiment. However this time participants watched both low arousal videos and high arousal video after which their debugging performance was measured. The next part of the chapter explains and examines the validity of the

[1] Much of the material in this chapter is already published as a paper in Khan, I.A., Hierons, R. and Brinkman, W.-P., (2007) Moods and Programmers' Performance, *Proceedings of PPIG 2007* , pp. 3-16,.

material used to study the hypothesis. This includes validation of the movie clips used to induce moods of the participants as well as debug questions used to test participants. In addition the feasibility of the first study that was conducted over the internet will be discussed. This will follow with the description of the experimental methodology used to investigate the hypothesis H_1. The participants, experimental procedure, design and the results section will follow on to the methods section. After this the second experiment will be explained looking at the experimental methodology, procedure, design and results. The final section will discuss conclusions, limitations and any future research work.

3.2 Experimental Material

The experiment[2] involved five movie clips to induce moods and a Multiple-Choice Questionnaire (MCQ) to measure the programmer's performance on recognizing errors as well as on understanding program flow in a piece of software code. The first step towards the experiment was the validation of the movie clips and the debugging questions. Pre-validation of material used in mood studies is often required. For example Thayer, Newman and McClain (1994) validated the categories of behaviour that covered methods of changing bad moods. Kirchsteiger, Rigotti and Rustichini (2006) induced moods of the subjects and validated the required effect of mood induction using an 8-point scale. The following sections will discuss how the validity of the experimental material was examined for the experiment reported in this chapter.

3.2.1 Validating the Debugging Test

A collection of 24 C/C++ multiple-choice questions[3] from a textbook on C++ (Deitel, Deitel and Neito, 2000) were used in order to assess the performance of the programmers. These 24 questions were selected with the intention of further dividing them into three levels: easy, medium and difficult, each level containing a set of 8 questions. All the questions were multiple-choice questions with three or four answers. The use of MCQ tests in the Information Technology (IT) field is considered to be suitable as they are used extensively for assessment purposes in courses such as computer programming and physical education (Roberts, 2006). MCQs have also

[2] The experiment can be done by visiting http://uxisfyp1.brunel.ac.uk/cspgiak/
[3] Download debug questions from http://mailtoiftikhar.googlepages.com/moodvalidation

been the focus of considerable research effort. For example, Lister et al. (2004) used MCQs to investigate the reading and tracing abilities of students. Fitzgerald, Simon and Thomas (2005) studied strategies student used in answering code based MCQs. Roberts (2006) explained the importance of MCQs in formative and summative assessments in details.

Due to the common use of MCQs in computer science education it was decided to use MCQs, as potential participants would be familiar with this type of questioning. Including questions of different levels of difficulty ensured that the test was able to measure debugging performance of programmers with different levels of expertise. The selection of questions into three levels (easy, medium, and difficult) was based on the assumptions that questions from the initial chapter of a book might be easier than later chapters' questions because of the introductory nature of the initial chapters. Similarly a question appearing at the start of the exercise might be easier than one at the end of the exercise. In addition three computer science lecturers also rated the level of the selected 24 questions from easy, medium and difficult.

Table 3.1 shows the selection strategy of questions for final debug tests. The two sets each of 6 selected questions were used to create two debugging tests, both including three easy and three medium level questions. As too few of 24 questions were consistently rated as difficult this level was not included in the debug tests. Finally, the questions were translated into C#, Java and Visual Basic dot net, allowing programmers to take the debugging tests in their preferred programming languages.

Table 3.1: Criteria for the selection of questions for the debugging tests

Question ratings	Number of Questions	Selected (level, Selection basis)
Number of questions rated as easy unanimously	4	4 (Easy, All lecturers rated as easy)
Number of questions rated as easy by two & medium by one lecturer	7	2 (Easy, 2 questions selected randomly from 7)
Number of questions rated as medium by two and easy by one lecturer	6	6 (Medium, All 6 selected as medium)
Number of questions rated differently from each other by all three lecturers	7	0 (Due to wide differences in rating by lecturers none selected)
Total Question	24	12

3.2.2 Validating the Movie Clips

Movie clips were used to induce moods in this study. Mood induction procedures are widely used in various studies including Kirchsteiger, Rigotti and Rustichini (2006), Mayer, Allen and Beauregard (1995), and Nasoz et al.(2003). There are various mood-inducing methods like Velten Mood Induction Procedure

(VMIP, mood induction by positive and negative statements), Music Mood Induction Technique (MMIT, mood induction by using different pieces of music) and Movie Mood Induction (MMI, mood induction using different movie clips). Moods were induced using movie clips as Westerman, Spies and Hesse (1996) found presentation of a film or story to be most effective in inducing both positive and negative mood states. Various studies used movie clips to induce moods like Shapiro, Maclnnis and Park (2002), Ambady and Gray (2002) and Rooijen and Vlaander (2006). For a detailed discussion of the mood induction procedures and their comparisons see Westerman, Spies and Hesse (1996).

Studies have used various different movies for mood induction purposes. Examples of research in which movie clips were used to induce moods include Sanna (1998) and Ambady (2002) etc. However this research did not utilize the movie clips that were already used in the literature as they might not be suitable for a group of programmers. Wynekoop & Walz (1998) compared three groups of programmers, system analysts and managers based on standardized personality test. They found that personality profiles of system analyst and mangers differed widely from programmers but not from one another. Therefore it can be presumed that programmers might be a different group, and thus it was important to select movie clips validated for the target population in this case the programmers. A set of 14 movie clips were selected from different online sites and sources based on their contents to represent a specific quadrant of the two-dimensional mood model (see chapter 2). For example a selected clip from the movie "Dead Poet Society" shows a death scene and mourning by the dead person's friends. It was expected that this movie clip would represent low valence and low arousal. To represent high valence high arousal quadrant a funny clip was selected. A clip from the movie "Fascinating Nature, The Most Spectacular Landscapes in the World" was selected to represent high valence low arousal and a clip from "Amityville Horror" was selected to represent low valence high arousal as it contains a scary scene. The neutral movie clip "Clouds model" contained information about designing clouds in computer games and was expected to have a neutral impact.

To rate the movie clips on the valence and arousal dimension, the SAM (Self Assessment Manikin) scale was used. SAM scale is often used in studies that apply self-reporting measures and has been presented as a promising solution to the problems associated with measuring emotional responses (Morris, 1995). SAM represents PAD (Pleasure Arousal Dominance) along a nine-point scale using

graphical characters. Pleasure or valence is the degree of happiness or sadness. Arousal is the degree of calmness or excitement. Dominance measures controlled or submissive to in-control and powerful feelings. However, only two dimensions, valence and arousal, were used in this study to make the self-reporting simpler. For valence the figures range from smiling happy figure to frowning unhappy figure. For arousal the figures range from excited open eyes to sleepy closed eyes.

To test the mood induction nature of the movie clips a mood validation study was carried out. A total of 10 participants (age M = 26.4 years; SD = 2.41; age range from 24 to 30; 2 females; 8 PhD students; 2 Masters Students; 8 with programming experience) participated in the study. The duration of each movie clip was from 3 to 4 minutes, and after each clip, they rated their mood on the two dimensional nine-point SAM scale (Lang, 1980).

The average time taken to watch all 14 movie clips and to rate them on valence arousal scale was approximately 60 minutes. All 14 movie clips were in the range of 3 to 4 minutes. All movie clips were named according to a sequence i.e. Movie 1, Movie 2 … Movie 14. The application started by generating a random number between 1 and 14. The generated number movie was then played by the application. After watching a movie clip, participants rated their mood on valence arousal scale and their ratings were saved in the database. The process continued until all the movie clips were rated.

Based on the high average ratings towards specific valence arousal quadrant, 4 movie clips out of the 14 were selected for the study. Each clip represented one of the four quadrants by having the largest Euclidean distance from the mid point. Additionally a neutral clip was selected, which had the smallest Euclidean distance from the mid point (valence=5, Arousal=5). The rating of these 4 clips was compared with each other using paired-samples t-test. The neutral movie clip was excluded from this comparison as it was used as a control reference point in the experiment. The 95% confidence intervals as apparent from Figure 3.1 indicate that there is no significant difference between movie clips of similar valence or arousal quadrants. However there was a significant difference between movie clips in opposite valence or arousal quadrants.

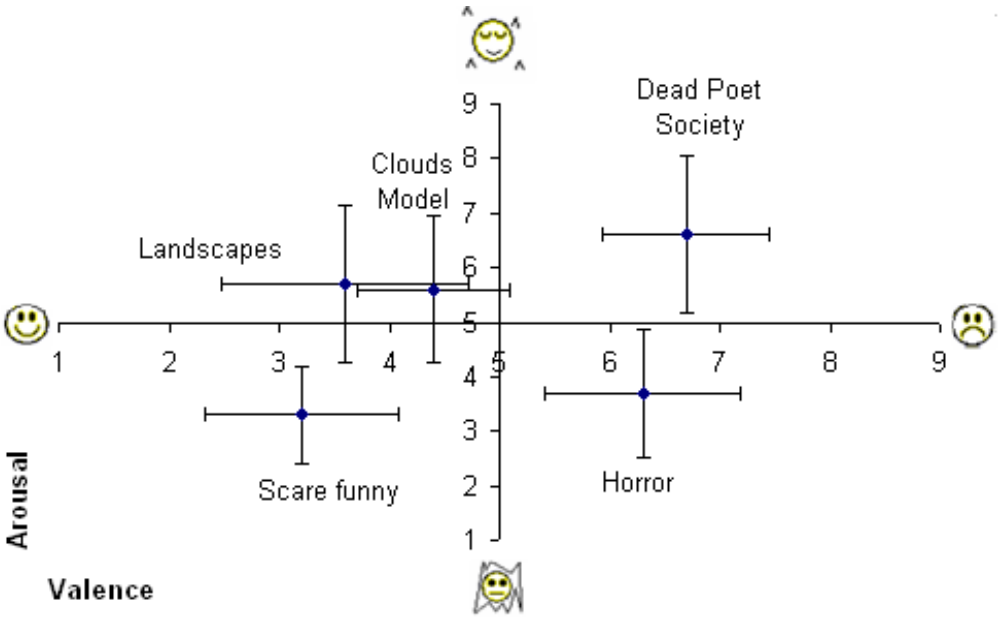


Figure 3.1: *Average mood ratings and their 95% confidence interval of the 4 selected mood inducing video clip and the mood neutral video clip (Clouds Model*).

3.3 Experiment 1

This experiment was conducted over the internet. Results from web experiments and surveys have been reported to be quite similar to those produced in laboratory environments despite great differences between the subject characteristics (Krantz and Dalal, 2000). Conducting the experiment on the web provides additional benefits like: 1) participants are recruited locally and globally and therefore more representative of the population (generalizable demographically); 2) web studies have high external validity and depict original behaviour; 3) the experiment can be taken at any time at participant convenience (time generalizeable); and 4) the costs are low in web studies and there are greater statistical powers as the participant pool is almost unlimited in size (Krantz and Dalal, 2000). Similarly participation in web experiments is completely voluntary (Reips, 2000).

There are also some disadvantages to this type of experimentation like little control of the environmental conditions, a high drop out rate from internet experimentation, etc. Although there are disadvantages in internet experimentation, data gathered in this way is authentic because of the high correspondence in the results from internet and laboratory experiments. Similarly a high drop out rate is an indication that participation in the experiment were voluntary (Krantz and Dalal, 2000). It was therefore decided to perform this experiment on the internet which also

increased the chances of involvement of professional programmers required for the study, a task relatively difficult to gain in laboratory settings.

3.3.1 Participants

Data was obtained from 372 cases in which people started the experiment; however in only 75 cases was the experiment completed forming a dropout of 80%. At the start of the experiment, participants were asked to indicate the number of times they are taking this experiment. Three out of 75 participants indicated they took the experiment more than once. Therefore the analysis was done on 72 cases in which participants took the experiment for the first time. Five participants were females and 67 were males. Participants' age ranged from 18 to 44 years with a mean of 26.3 and a *SD* of 5.2. Programming experience ranged from 0.5 to 25 years with a mean of 4.8 years and a *SD* of 5.6. Participants included 81% professionals, 8% postgraduates, 6% undergraduates, 4% PhD students, and 1% hobby programmers. In the experiment 58% participants took the debug test in C/C++, 24% in C#, 11% in Java, and 7% in Visual basic dot net.

3.3.2 Procedure and Experimental Design

The experiment was conducted on the internet. Invitations of the experiment were sent via various programming forums. At the start of the experiment participants saw an introduction page, which explained the nature of the experiment and requested the participants to give their consent for participating in the experiment and for the collection of data. After this the participants went through a training session so that they should get familiarized with the sequence of the test, which consisted of watching a video clip, followed by a series of debug MCQs that they had to answer within a fixed time set, and finally participants were asked to rate their level of arousal and valence on the SAM scale. After completing the training session, participants continued with the actual test, which consisted of two cycles of the movie, debugging test, and mood-rating sequence. There were two fixed debugging tests of six questions. The movie clips were either the neutral movie clip, or one of the four mood inducing movie clips. To control for potential training effects, the order of the neutral movie and the mood inducing movie clip as well as the selection of the specific mood movie clip was systematically controlled. The system assigned participants in cycles of eight to the eight possible sequences (four clips, two orders

for each). Furthermore, the assignment of the two versions of the debug test to the first or the second cycle was counter balanced. The online experiment sequence is also illustrated in the Figure 3.2 below.

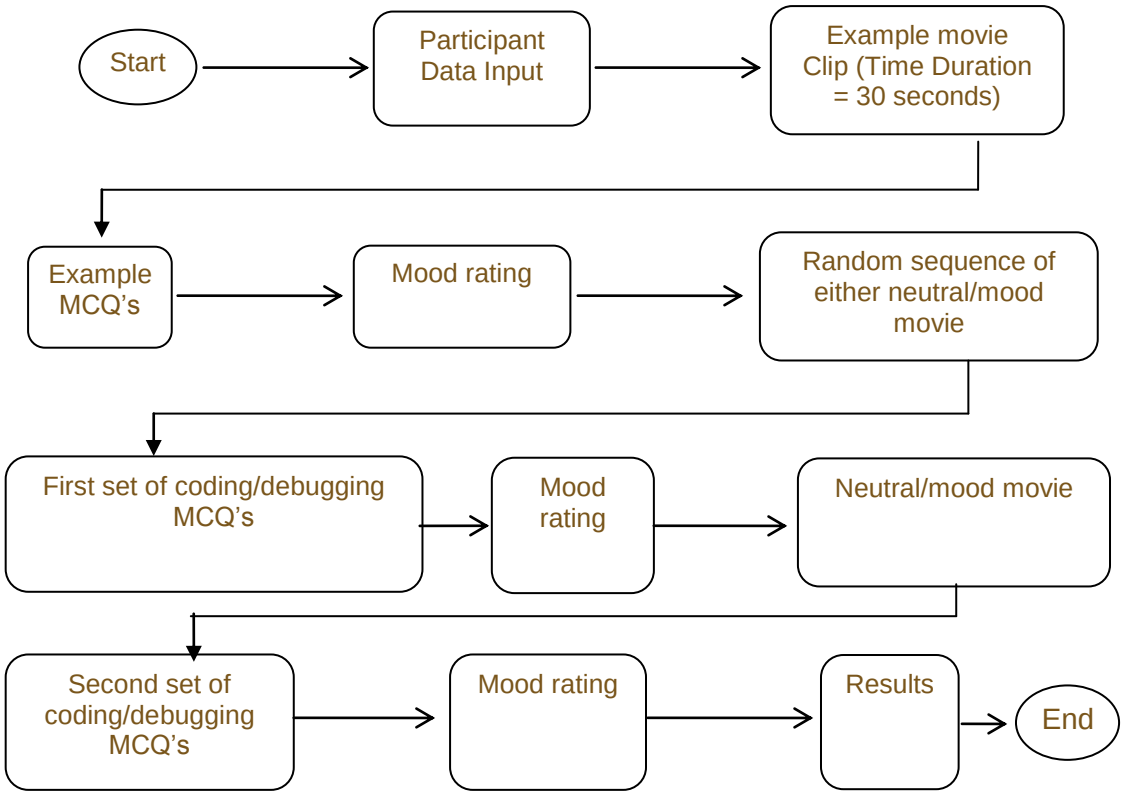


Figure 3.2: *Online experiment flow sequence*

The design of the experiment was considered for a 2x2 factorial design with valence and arousal each with two levels as independent variables. Variables 'number of correct answers' and 'number of debug questions answered within the allocated time' was used as dependent variables. The numbers of correct debug answers in the neutral mood condition was used as baseline covariate condition. The overall time to complete the debug test was limited in order to ensure that the effect of the movie clips remained persistent throughout the test. Every question had a separate timing, which was allocated on the basis of the length of the text and the number of calculations involved in a question. For example if a question contained 7 lines of code then this question was allocated 70 seconds. This can be further clarified from the following example.

```
1. int x = 3;
2. cout << x++ << " ";
3. x--;
```

```
4. --x;
5. cout<<x <<" ";
6. cout<< --x<<" ";
7. x+=2;
```

The piece of code above was included in debugging test and contains 7 statements. While solving this problem, a programmer first had to read and understand the code. This also meant stepping through the code and seeing how values of variables changed. Take the previous example, programmers will first assign value 3 to variable x, and by keeping it in memory will either increment or decrement the statements below. At each step they could get a new value, which they either have to remember or write down. In the process they may have to refer to previous statements again and again. This study assumed this to be a time taking process and therefore assigned 10 seconds to each statement. However another question involving computation like (x = 2 + 2 * 10 / 2 – 2) was allocated 50 seconds whereas it only contained one statement. The mathematical statement above could be divided into five sub statements as the example below illustrates.

1. 2*10 = 20
2. 20/2 = 10
3. 10+2 = 12
4. 12 – 2 = 10
5. x ← 10

The minimum time allocated to any question was 30 seconds. There were 12 questions divided into two parts of six questions each. Each part was counter balanced by the computer. Each succeeding participant was assigned the opposite sequence as their predecessor, avoiding any systematic bias later in the analysis that might be caused by a difference in the difficulty of the two parts of the tests. Each part had the same time allocated to it, having approximately 250 seconds to answer. Questions for the programming language C/C++ can be found in Appendix I.

If participants answered the question before the time deadline, they were moved automatically to the next question in the test. About 63% of the questions answered by the participants in the experiment were completed in less than the maximum allowed time. The remaining 37% either answered the question taking the maximum time possible (9%) or left the question unanswered (28%).

3.3.3 Results

The first preparation step towards the analysis was to examine a potential self-selection bias between the conditions of the experiment, e.g. less motivated people that dropout of the experiment after seeing the low arousing movie, leaving only highly motivated people in that specific condition. Especially for web experimentation, with its truly voluntary nature, it is important to examine the distribution of the dropout rates across the experimental conditions (Reips, 2002). A systematic difference between the conditions would confound the interpretation of any causal effect between the experimental conditions and the programmers' performance.

A total of 72 (20%) participants completed the study and the remaining 80% decided to dropout. An analysis on the dropout data showed that 93% participants decided not to participate after reading the project description while only 7% decided to terminate participation half way in the experiment when they were assigned to an experimental condition. Table 2 shows the distribution across the experimental conditions of the 7% dropout and the number of people that participated. A Fisher Exact Test on this 7% dropout rate revealed no significant ($p > 0.05$) variation between the experimental conditions, and therefore cancelling out self-selection as an alternative explanation for any causal effect found. Still examination of Table 2 did reveal a significant (χ^2 (3, N = 72) = 17.2, p = 0.001) deviation from an equal distribution of participant that completed the experiment across the conditions. Although this does not confound any potential statistical results, it constrains possible statistical analyses such as a full factorial 2×2 analysis.

Table 3.2: The number of participants that complete the test (NC), the number (ND) and percentage (PD) of dropouts after being allocation to experimental conditions

	Arousal								
	Low			High			Total		
Valence	NC	ND	PD	NC	ND	PD	NC	ND	PD
Low	7	1	13%	14	5	26%	21	6	22%
High	31	7	18%	20	4	17%	51	11	18%
Total	38	8	17%	34	9	21%	72	17	19%

A number of multivariate covariant analyses (MANCOVA) were conducted to examine the effect of videos on the performance of the debug task. The individuals' ability to complete these debug tasks was controlled by looking at performance after seeing the neutral movie, i.e. the neutral mood condition. The covariant was therefore

the number of correct debug answers in the neutral mood condition. As the numbers of participants in the low valence low arousal (LVLA) quadrant was too small to conduct a 2×2 analysis, a MANCOVA was carried on only the movies from the other three quadrants (HVHA, HVLA, and LVHA).

The analysis took as independent between-subjects variable the videos (3 levels, excluding low valence / low arousal, LVLA, condition) and as dependent variables the number of correct answers and the number of tasks completed within the time set. The multivariate analysis results showed significant main effect (F (2, 65) = 3.54, p = 0.036) for the videos. Examining the results of the univariate analyses showed that this effect could be found back in the number of correct answers (F (3, 65) = 6.35, p = 0.001) and in the number of tasks completed within the time set (F (3, 65) = 5.18, p = 0.003). These results suggest that mood could affect participants' performance in the debug tasks.

The next step was to examine this effect on both the arousal and valence dimension. To study the arousal dimension, the previous analysis was repeated. However, this time the independent variable movie had only two levels ((1) high valence/ low arousal HVLA, and (2) high valence / high arousal HVHA). In other words, valence was kept constant at the high level condition while the arousal level varied. The results showed no significant effect of the movies (F (2, 65) = 3.13, p = 0.051) still there was a trend toward significance, and this main effect was again found in the univariate analyses on the number of correct answers (F (5, 65) = 4.27, p = 0.002) and on the number of tasks completed within time (F (5, 65) = 3.33, p = 0.01). Examination of the averages, corrected for the effect of the covariant, showed that the participants on average gave 2.37 correct answers in the low arousal condition and 3.14 correct answers in the high arousal condition. Similarly, they completed 3.10 tasks within time set in the low arousal condition and 4.24 in the high arousal condition. This result suggests that arousal level can affect person's debug performance, and in this experiment performance was improved with an increased arousal level[4].

A similar analysis was conducted for the valence dimension. This time the dependent variable only included the low valence / high arousal (LVHA) condition

[4] Note that the similar analysis as above was not conducted keeping valence constant at a low level because of the small number of participants in low valence condition.

and the high valence / high arousal (HVHA) condition, thus arousal was kept constant. The results of the analysis this time found non significant effect for movies (F (2, 65) = 2.49, p = 0.09) with a trend toward significance. Univariate analyses also found non significant effect in number of correct answers (F (2, 65) = 2.71, p = 0.075) again with a trend toward significance. However there was a non significant effect in the number of tasks completed within the time set (F (2, 65) = 1.21, p = 0.3). An examination of the averages, corrected for the effect of the covariant, showed that the participants on average gave 2.57 correct answers in the high valence condition and 3.21 correct answers in the low valence condition.

Potential violations of the assumption for the covariance analyses were also tested such as (1) the linearity of the relationship between the covariate and the dependent variable[5] and (2) homogeneity of covariate-dependent variable slopes[6]. No indication of a violation was found.

The analysis above indicated that arousal is a factor in a two-dimensional valence arousal model having a significant effect on programmers debug performance. However in the preliminary analysis, a MANOVA conducted on the same data revealed a non-significant effect of valence. Arousal was not significant with p = 0.057 levels. As this was not a clear outcome, a similar study was conducted to replicate the findings considering only the arousal dimension. The study was designed to use movie clips that induce only high and low arousal as within-subject factors. The second experiment therefore was conducted on the bases of inconclusive results from the initial analysis. The following section contains the details of the procedural, experimental design, and participants of this experiment. This section is followed by a discussion of the results.

3.3.4 Effect of Debugging Tasks on Mood Ratings

In the design of the experiment, participant self reported their moods *after* the debug task. Reinecke and Trepte (2007) suggested after a literature review that level of performance in a task alters effects of low or high arousal. A low level performance results in low arousal which in turn causes boredom. Similarly a high level

[5] Significant correlations were found between the number of correct answers in the neutral condition (covariant) and in mood condition (dependent) (r(72) = 0.42, $p.$ < 0.001) and between the number of correct answers in the neutral condition and number of tasks completed with in time (dependent) in neutral condition (r(72) = 0.45, $p.$ < 0.0001) and in mood condition (r(72) = 0.32, $p.$ < 0.01).

[6] No significant interaction effect was found between the covariant and other independent variable (mood)

performance results in stress because of keep performing better which results in high arousal. In this study moods (valence, arousal) were measured after the coding\debugging tasks. An analysis of covariance was used to assess whether movie clip induced moods have an impact on mood induction even after debug task was completed. The overall impact of mood inducing movie clips was analyzed on the mood ratings obtained from the participants. First an ANCOVA was carried on with valence rating after completing the debug task as dependent variable. Movie clips from three dimensions were used as a fixed factor leaving out LVLA dimension because numbers of participants in this dimension were very low as discussed before. Valence rating after completing the debug task in the neutral movie clip condition was used as a covariate. The analysis found was not significant (F (2, 63) = 1.19, p = 0.31) for valance. A similar ANCOVA was carried also on arousal rating. Again, movie clips from three dimensions were used as a fixed factor leaving out LVLA dimension. Arousal rating in the neutral condition was used as a covariate. The results revealed also not significant (F (2, 63) = 0.76, p = 0.47) for arousal.

The non significant p value might be an indication that after debugging task the mood induced movie clips no longer had a specific effect on the participants' mood as the debugging task had changed participants' mood to non specific mood levels for each experimental condition. As mood movie clips were validated and found to be mood inducing, the non significance most likely can be attributed to coding\debugging tasks. Analysis above revealed coding\debugging task might have changed the impact movie clips had on moods. However as coding and debugging task was constant in both mood and neutral conditions, the significant improvement in performance as discussed in the analysis above could only be attributed to variable moods induced by movie clips. Another .possibility of this non significant effect could be time rather than coding\debugging task itself. As coding debugging task was little above three minutes duration, effect of induced mood by movie clips could have reduced in this time span.

3.4 Experiment 2

3.4.1 Experimental Design

In addition to its difference as being a lab study, some other notable differences in experiment 2 from experiment 1 were:

1. Mood inducing videos from arousal quadrant were selected to induce moods as within-subject factor.
2. Some extra measures based on participants subjective observations were introduced in order to further understand the impact of mood on programmers.
3. In experiment 1 a neutral movie clip was used as a covariant factor. This experiment was designed considering arousal as a within-subjects factor.

The setup of the experiment was to measure only main effect and no interaction effect. The experiment was setup to measure effect of arousal on performance, but it was also possible to measure main effect for valence without the possibility of measuring the interaction effect between arousal and valence as not all conditions for a full 2x2 factorial design were included in the experiment. Therefore it was not a 2x2 factorial design. It consisted of arousal rating after watching high arousal and low arousal movie clips as an independent variable. The dependent variables used were: the numbers of correct answers, the average time in which debug questions were answered and the four perception questions.

3.4.2 Material and Procedure

For the experiment a windows application quite similar to web version was developed. The start included introduction to the experiment as well as consent form for participant. After obtaining the consent, a sample video was played followed by a couple of sample questions with multiple answers options. This was followed by a mood rating form including SAM (Self Assessment Manikin) for rating valence and arousal level. In addition, the following subjective questions were also included on the mood rating form.

1. I found this quiz easy
2. I think I made no errors in this quiz.
3. I think I performed well in this quiz.
4. I found that there was enough time to answer the quiz questions.

Participants were able to rate their opinion on a 10-point Likert scale ranging from ‘strongly agree’ starting from 1 to ‘strongly disagree’ representing by 10. Because of the training phase, participants understood what they could expect and thus eliminating the element of surprise. The sequence of experiment 2 was similar to experiment 1 as explained in Figure 3.2. As indicated earlier movies varying on arousal dimension and was therefore introduced as a within-subject factor. The movie

clips were displayed in a pre set format as is explained in the Table 3.3 below. To reduce the complexity, the experiment was primarily setup to study a main effect for arousal and secondarily a main effect for valence. Examining a potential 2-way interaction between these factors was not pursued.

Table 3.3: Preset format of Movie Sequence, each row indicating one complete test with a participant.

Quiz	Movie Clip 1	Quiz	Movie Clip 2	N
A	HVHA	B	HVLA	2
B	HVHA	A	LVLA	3
A	LVLA	B	LVHA	3
B	LVHA	A	HVLA	3
A	HVLA	B	HVHA	3
B	LVLA	A	HVHA	1
A	LVHA	B	LVLA	3
B	HVLA	A	LVHA	2
	Total Participants			20

***Note**. Example movie clip remained the same for all participants, N = Number of participants that took the test in the sequence*

In the experiment participants were asked to complete the debug tests used in the previous experiment. Like experiment 1 participants had an option of taking the test in their preferred programming language out of C++, C#, Java and Visual Basic. The result of the participants quiz was presented at the end, with the combination of correct answers from both quiz A and B.

3.4.3 Participants

Participants were invited for the experiment in two ways. Firstly the test application was packaged and sent to one of the participants via his email. This nominated person along with taking the test himself also organized the experiment to be conducted on his PC by 5 other participants as well. The rest of the 14 participants were contacted personally and took the experiment on the laptop of the experimenter. However instructions were presented to all of the participants in only one language (English) as well as all participants used the same English version of the application. A total of 20 participants participated in the study out of which there were only two female participants. The mean age of the participants' was 26.6 years with a standard deviation of 2.6 years. The mean experience of participants in the programming field was 2.9 years with a standard deviation of 2.7. About 30% participants took the tests in C#, 40% took the tests in C/C++, and 30% took the tests in Java. Ten percent

programmers were professionals, 40% were post-graduate students, 35% were undergraduate students and 15% were PhD students.

3.4.4 Results

The basic aim of the experiment was to observe the effect of arousal (high, low) induced by movie clips on the debug performance of the programmers and on their subjective observation. However the effect for valence (high, low) was also analyzed in the case of participants where participants watched varying low and high valence movie clip. The first analysis carried out was a MANOVA with as independent with-subject variable arousal with two levels (low arousal and high arousal). The dependent variables were: the numbers of correct answers, the average time in which debug questions were answered and the average score on the four opinion questions. The response on the four questions was taken together as Cronbach's alpha examination indicated a high level of consistency between the responses (low arousal condition $\alpha = 0.89$, and high arousal condition $\alpha = 0.89$). The multivariate analysis results (Table 3.4) showed a non-significant main effect (F (3, 17) = 1.13, $p = 0.36$) for arousal.

Table 3.4: Results multivariate and univariate analysis on arousal

Data Item	*Mean in HA*	*Mean in LA*	*Hyp. Df*	*Error Df*	*F*	*p*
Joint MANOVA			3	17	1.13	0.36
Total correct answers	3.30	2.65	1	19	3.77	0.07
Average time taken to answer the debug question with in time	28.65	28.30	1	19	0.03	0.87
Combined subjective observations	5.92	5.80	1	19	0.24	0.63

Note: *HA = High Arousal, LA = Low Arousal.*

The univariate analysis (Table 3.4) showed non significant effect of arousal on the programmers' subjective observations of their performance. Similarly average time taken to answer questions also emerged to be non significant. The number of correct answers in high and low arousal condition was also not significant (F (1, 19) = 3.77, $p = 0.067$) however showed a trend toward significance with an average of 3.3 correct answers in the high arousal condition and an average of 2.65 correct answers in the low arousal condition.

A second MANOVA was carried out with as independent with-subject variable valence with two levels (low valence and high valence) on the 9 participants that were assigned to sequence (Table 3.2) in which the two video clips had different valence levels. The dependent variables were again the combined questions score, the

number of correct answers, and the average time in which a debug question was answered. The multivariate analysis results (Table 3.5) showed a non-significant main effect (F (3, 6) = 1.35, p = 0.42) for valence. Examining the results of the univariate analyses showed that valence had no significant effect on the number of correct answers (F (1, 8) = 3.77, p = 0.088) but have a tendency. Note that potential order and arousal effects were controlled by counterbalancing these two factors in the selected sequence in valence analysis.

Table 3.5: Results multivariate and univariate analysis on valence

Data Item	*Mean in HA*	*Mean in LA*	*Hyp. Df*	*Error df*	*F*	*p*
Joint MANOVA			3	5	1.35	0.36
Total correct answers	2.25	3.12	1	7	3.94	0.09
Average time taken to answer the debug question with in time	7.37	8.29	1	7	0.13	0.73
Combined subjective observations	5.31	5.15	1	7	0.61	0.46

Note: *HA = High Arousal, LA = Low Arousal.*

3.5 Discussions, Limitations and Future Research

The result of the first experiment suggests that programmers' mood can affect their debugging performance. More specifically, the first experiment shows that the arousal mood-dimension could effect how the participants complete debug tasks. However second experiment failed to show earlier experiment findings but showed tendency toward that. The reason for these findings will also be discussed in this section. There these studies support hypothesis H_1 (*Programmers' moods affect their debug performance in respect of finding and solving errors in the code*) of this study suggesting a direct relation between mood and programmers debugging performance as was already indirectly suggested by literature review in Chapter 2.

The findings of these studies seem to agree with previous research that also suggests that arousal is related to performance, as is expressed with the Yerkes-Dodson Law (Yerkes-Dodson, 1908). Yerkes-Dodson Law suggests an inverted U shape relationship between arousal and performance and thus is also termed as Inverted-U hypothesis. According to this hypothesis a certain level of arousal increase performance and less or over arousal may decrease performance. For a detailed discussion of arousal models see Derryberry and Rothbart (1984).

Taking the inverted U-shape performance hypothesis and the result of the experiments with increased performance that coincide with an increase of arousal level, the low performance-low arousal result can be positioned at the left side of the inverted U shape, and the high performance – high arousal on the right side of this point at a position where the performance is still above the low performance – low arousal point as illustrated in Figure 3.3. The inverted U shape performance hypotheses also put forward a limitation on how the rise of arousal would improve programmers' performance. As the diagram suggests, at a certain point an increase in arousal would worsen the programmers' performance. However, this was not examined in this experiment and therefore would require future research.

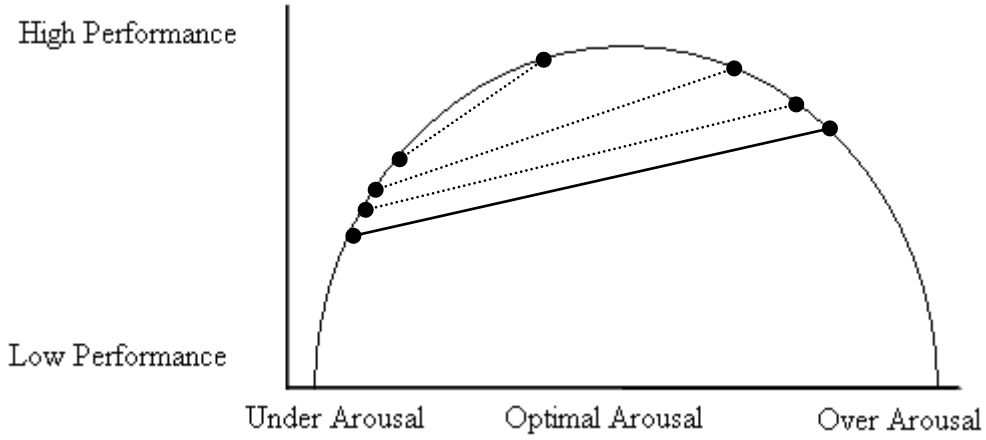


Figure 3.3: *Arousal Performance relationship in terms of Inverted-U shaped hypothesis including the potential places of experimental conditions (left side represents low arousal while the right side represents high arousal***)**

Various studies indirectly suggest that valence (positive and negative moods/happiness or sadness) does have an impact on programming performance along with various other interrelated tasks as is indicated in model represented in Figure 2.3. However in this study a non significant effect was found for valence dimension, which raises the need for additional experiments. Although valence also showed a non significant effect on the performance it showed tendency to have an impact on performance. The mean analysis of correct answers in high valence in both studies reveals less correct answers as compared to mean in low valence. This suggests that the participants performed better in low valence than in high valence mood state.

Experiment 2 resulted in non significant findings as compare to the first experiment. A number of factors might have caused this, for example:

1. Participation in experiment 1 was on a volunteer basis and participant only decided to participate after reading the description. Contrary to the open invitation of experiment 1, participants for experiment 2 were actively recruited by the experimenter. Although these participants participated on voluntary basis, an element of social pressure cannot be excluded which might cause participants to be less willing to be emotionally affected by the video clips.
2. Although participation in experiment 2 was anonymous they knew the experimenter. The feeling of being 'observed' or even 'assessed' by someone familiar might have created a situation where participants were less willing to be emotionally engage in the video clips.
3. Experiment 1 included 72 participants, whereas in experiment 2 included only 20 participants, which affects the power of the statistical analysis.

Some of the potential limitations of the study are that the tasks given to the programmers might not be representative of the entire industrial debugging task where most of the efforts are used to locate and identify relevant code while ensuring that these changes did not create any ripple effect. Similarly, the studies only focussed on the relationship (mood and debugging task) put forward by the CPTM, the CMM and the MPM model discussed in chapter 2. Future research might examine other relationships put forward by models, such as the effect of mood on program construction, or on maintenance tasks. Besides studying these effects in lab, it is also important to study them in an industrial environment. The findings presented here could be regarded as a first step in developing a deeper understanding, as the experiments shows that moods have at least an impact on programmers' debugging tasks.

Two main contributions can be identified in the work presented in this chapter. First, in the preparation of the study, five movies were empirically validated on their ability to evoke specific moods and thus enabled advanced understanding of programmers' mood and their debugging performance especially the relation between arousal level and finding debugging errors. Secondly this study found significant effect for arousal-mood dimension on programmer's debugging performance. This contributes toward support of the first hypothesis.

The studies in this chapter suggest that moods, specifically arousal, could affect programmers' debug performance which addressed the first hypothesis of this research. The next question inline was how to measure these moods during work? In

order to help programmers to improve their performance software\human need to know the mood that caused low performance. Although there are various methods to measure moods, most of these methods are obstructive or intrusive and therefore are not practical in the case of programmers. There was a need for a method that could measure moods in a simple and in an unobtrusive manner. Therefore two experiments were planned to measure moods from interaction behavior of the users with keyboard and mouse and are explained in the next two chapters. The first experiment in this series as explained in the next chapter was a field study. It was designed to record moods and computer interaction in a natural environment over a period of days with out any mood induction.

Chapter 4: Mood Measurement by the use of Keyboard and Mouse[7]

4.1 Introduction

This chapter covers the second hypothesis of this thesis, which states that "*Moods can be measured from the computer users' interaction with keyboard and mouse"* and will be represented as H_2. A large volume of literature exists presenting various approaches toward users' mood and emotion measurement. However all these methods somehow are interruptive or obstructive and therefore are less suitable for mood measurement of computer users and especially for mood measurement of programmers. This chapter will therefore present a new approach toward mood measurement of the computer users from their interaction with keyboard and mouse. This chapter will also form the basis for the creation of a tool that than can possibly be embedded in Integrated Development Environments (IDEs) for programmers' mood measurement and to help programmers to improve their performance.

This chapter will start by briefly looking at relevant literature that supports H_2. A detailed discussion was already presented in chapter 2 about the possibility of establishing the suggested measure. This chapter will therefore mainly discuss the literature to support the setup of the experiment. To test H_2 two experiments were conducted. The third section explains material used in the first experiment. To conduct the experiment an application was developed to record keyboard and mouse events in log files. The position is taken that log file recording is a useful and valid way of obtaining interaction data. Support for this position will be presented from the literature. The set up of the experiment will be presented, as well as results of the analysis conducted on the experiment. Instead of addressing the more challenging aim of establishing a measure that could be applied on any person, the analysis focuses on establishing individualised mood measures. As the analysis suggest that individualised mood measure might be possible for some but not all the participants, the analysis also focuses on factors that distinguish these two groups of users. The chapter finishes with reflection on the findings and suggestions for future research.

---

[7] Major part of this chapter is already published as Khan, I. A., Brinkman, W-P. & Hierons, R.M. 2008. "Toward Computer Interaction based Mood Measurement Instrument", Proceedings of 20th Annual Workshop of psychology of programming interest group, pp. 155-169.

4.2 Literature Review

Here a brief overview of the literature will be presented as was presented in detail in chapter 2 (section 2.3). Reeves and Nass (1996) argues that human computer interactions follow the principles of recognition, interpretations and expressions of emotions that are similar in human-human interaction An affect recognizing computer can communicate in a more natural way and can be more effective in its interaction with humans (Zimmerman et al., 2003). Affect recognizing computers not only make certain tasks easier to do but also provide effective assistant to address user frustration and sympathy by providing encouragement and comfort ant thus significantly reducing frustration levels (Klein, 1999). One group of the computer users - programmers- most often feel frustrations and become moody because of the deadlines, consistency, and expectations to produce (Ross and Zhang, 1997). Programmers therefore might be an interesting group for studying a mood measuring instrument.

Damasio (1995) argued that moods and emotions are necessary for reasoning and rational decision making. Programmers most often need reasoning and decision making to resolve programming problems. There are also some studies like Khan et al., (2007) that suggest that programmers' mood might have an impact on their debugging performance. However the task of affect recognition is complex in nature and involves various overheads. There are various affect recognition methods like self-reports and physiological methods. Still another less explored way of affect recognition and measurement is by considering a person's behaviour.

At the outset, behaviour based mood measurement seems possible because mood is an affective state and has an impact on behaviour (Zimmerman, 2003). Moods last from minutes to hours and days, and their influence on behaviour is more prominent than emotions (Zimmerman, 2003). These influences include: judgements and evaluative judgements (Siemer et al, 1998), intergroup discrimination (Forgas and Fiedler, 1996) and acceptance of certain risks (Isen and Nehemia, 1987).

Although the literature shows that moods have an impact on behaviours, this study focuses on measuring mood from human-computer interaction. The idea of using this type of data for mood measuring is not new. For example Johnson et al. (1998) experimented to extract ordered sets of training data from human behaviour for computer learning purposes. However these mood-measuring methods often require

complex software and algorithms as well as dedicated hardware like video cameras etc. Similarly these methods can be disruptive for users (Zimmerman et al., 2003). On the other hand, the use of keyboard and mouse data might reduce these overheads as these are the most basic, most familiar and cheap devices used by computer users. Furthermore keyboard and mouse logging is relatively easy to implement. This method seems to have received little attention in the literature. This study therefore aims at addressing this gap. The study presented is a first step toward measuring moods with the use of keyboard and mouse. The aim was to examine whether individual tailor made mood measure could be established. This would be a first step toward generalizable instrument that could be used across individual computer use. The next section will explain the experimental material, setup, participants, log files recording, and results. The analysis will also look at a personal factor that would help to predict whether person behaviour would correlate with their mood. The last section will be conclusions with a brief discussion on the similarities and differences of experimental procedures in this study. The next section will present the material used in the experiment.

4.3 Experiment

4.3.1 Experimental Material and Design

The basic idea of the experiment was to record keyboard and mouse events in log files along with self-reported mood data from the participants. The motivation was to use both datasets afterwards to study the correlations between them. A keyboard and mouse event logging application was developed that ran on participants' computer as a background process. This application was able to capture computer user keystrokes and mouse movements and to store them in a log file. A mood rating dialogue box (Figure 4.2) appeared after every 20 minutes asking participants to rate their mood on SAM (Self-Assessment Manikin) scale. The application also allowed participants to pause recording[8] for 5 and 10 minutes or for a self indicated time slot

[8] It was assured to participants that application would not record any personal information. To further increase the trust pause logging functionality was provided so participants could pause logging of events for a specific time interval

(Figure 4.3). In addition participants were able to exit logging[9], stopping pop-up[10] dialogue boxes and uninstalling the application with all the participant data deleted, in case of withdrawal from the study.

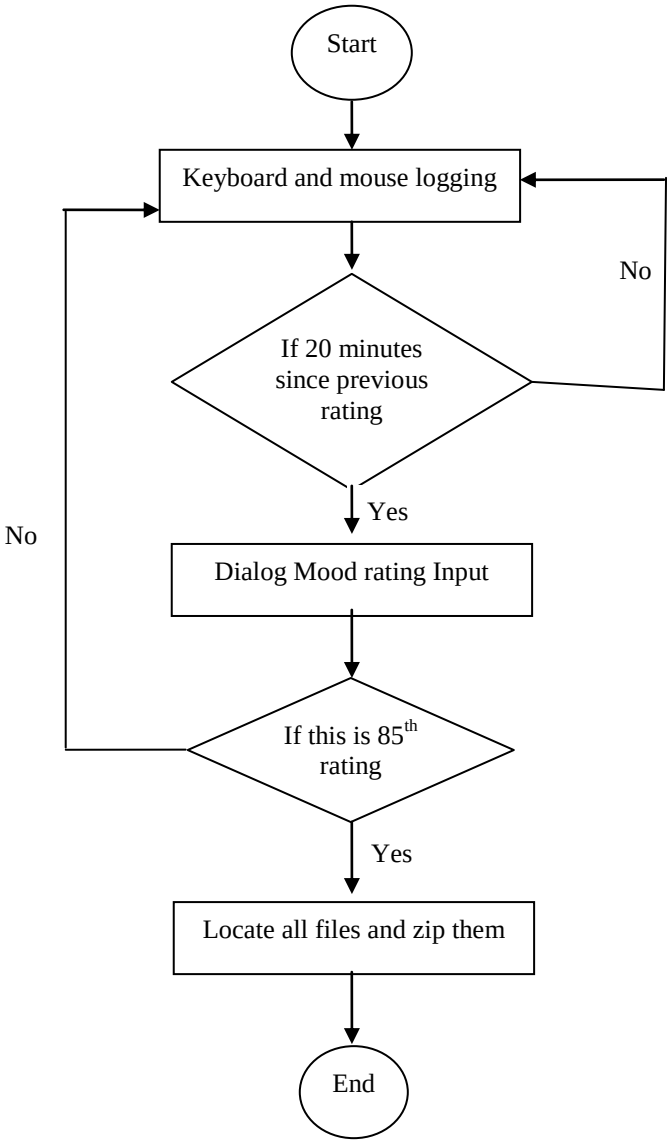


Figure 4.1: *Flow diagram of the application for keyboard/mouse logging*

The functionality of the application is explained in Figure 4.1. Application was designed to constantly run in the background and was accessible via an icon on task bar. This application recorded every key press and mouse click in log file. After every 20 minutes it displays a dialog box to ask participants to rate their mood as well as to

[9] Participants were able to exit logging at any time. The application was developed to restart itself at the next computer boot up process; however participants were able to restart application without rebooting computer whenever they feel so.

[10] The appearance of pop-up dialog box after a fix interval might be teasing specially when doing very important and concentration demanding work. This functionality was provided to stop appearance of pop-up dialogs until participant restart them again.

indicate what type of task they were engaged in (e.g. browsing, word processing or playing games). If the participants had filled out this dialog box for more than 85th times, the application told the participant that the experiment was completed and they could send the data to the experimenter. On average, the application ran on the participant's computer for 8 days before the 86 mood ratings were collected. The study was setup for a sample size of 86 mood rating data points for each participant. This sample size would give an 80% chance of finding at least medium size correlations with an alpha = 0.05 (Cohen, 1992 a), if these correlation existed. Therefore participants were instructed to answer at least 86 mood rating dialogues however some participants did not meet this request. The flow diagram of the application is illustrated in Figure 4.1. The flow starts from by initiating a logging mechanism to log keyboard and mouse events. After every 20 minutes of recording, a dialog box appears before participant for self-reporting of moods. If the rating is not 85th than the flow goes back to logging events whereas after 85th self reported ratings, the experiment finishes and participants could send data to researcher.

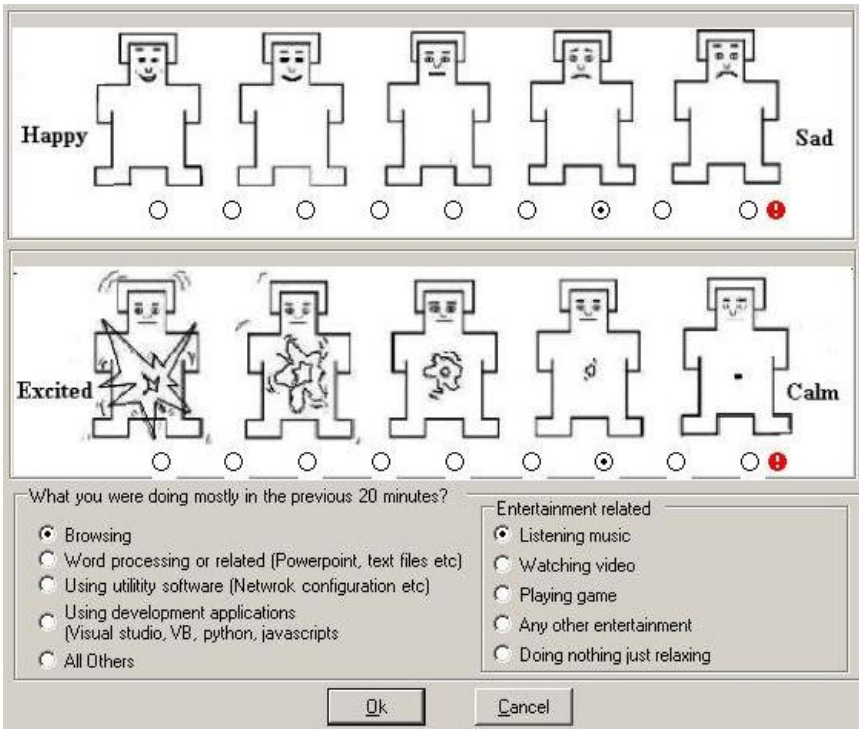


Figure 4.2: *Mood rating dialog with SAM (Self Assessment Manikin) scale*

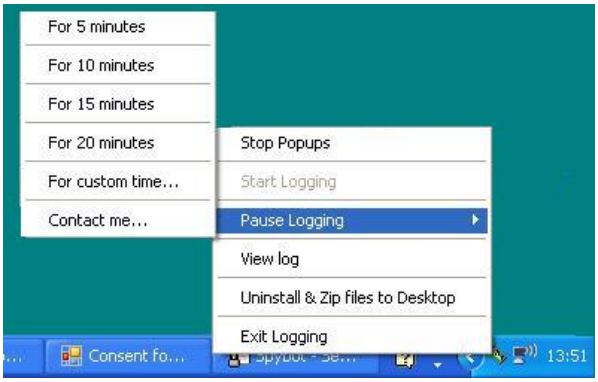


Figure 4.3: *Application menu and its further options*

4.3.1 Log File Recoding

The study used log files to record keystrokes and mouse clicks. Various researchers have used log files in different computer related studies effectively. For example, Kukreja et al. (2005) developed the RUI (Recording User Interfaces) tool to study human robot interaction as well as in human computer interaction. Haigh and Magarity (2000) used log analysis for measuring website use whereas Khan et al. (2008) used log files to study users' personality. Log files used in this study record keyboard and mouse clicks to find a relationship between interaction data and moods. The following events were recorded in the log files: capital alphabets as 'Capital Alphabet', short alphabets as 'Short Alphabet', numbers as 'Numerical' and special characters like /, @ as 'Special Character'. Recording a descriptive code instead of actual event was done to protect participants' personal data and information. Figure 4.4 shows an example of the log file used in this study.

4.3.2 Participants

Application developed was uploaded on the experimenter's website. Participants were informed about the experiment and were invited to download and install the application on their computer. Some people were reluctant to participate because of their negative perception of using log file recording techniques. Because of this trust issue participants were especially recruited within the social network of the experimenter. An opportunity sample of 26 participants participated in the study, which only include two female participants. The mean age of participants was 27 years with SD of 3. Their age ranged from 22 to 34 years. Fifty percent of the participants classified their self as programmers, 42% as expert computer users and

8% as medium computer users. The average computer use was 5 years (SD of 1.6) ranging from 2 to 9 years.

4.3.3 The Experiment Setup

The department ethics committee approved this experiment. Furthermore, participants gave their consent on a consent form before they could use the application. The application was installed on participants' computers as a background running process. Recorded data in the log files were in four different categories named as:

- *Window name*: This data was used to identify the application that was used at a specific event time. This data could help in identifying difference in keyboard and mouse use between applications. Special care was taken not to identify the document names participants were working on. That means that the application recorded only specific application names like Microsoft Word, Internet Explorer, and Visual Studio.
- *Keyboard or mouse event:* This data identifies a particular event. Events could either be of mouse clicks or of key press. Mouse clicks were recorded as "Mouse button" and key press events were recorded as "Key Up" or "Key Down" event.
- *Date and time of event:* This data was used to records date and time of an event and was used to find the time difference between events.
- *Category of event:* This data stores the category of the event. It stores "Left" or "Right" for the mouse button that was pressed, and stores "Short Alphabet", "Capital Alphabet", "Numeric key", "Special Character", if a key was pressed in these categories.

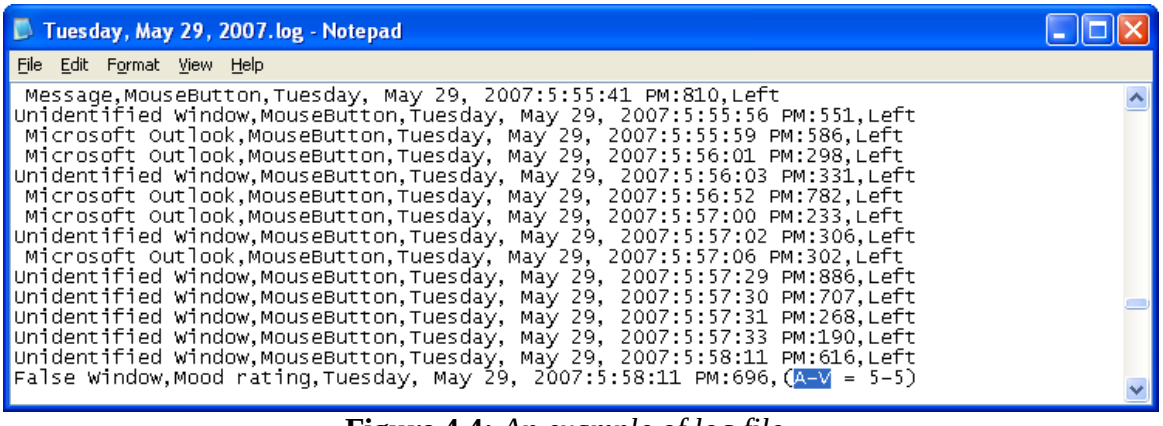


Figure 4.4: *An example of log file*

Log files also recorded mood rating from the mood rating dialogue. In the dialogue box (Figure 4.2) participant could also indicate the category of applications they were working with or the type of entertainment (music, games etc) they were enjoying before the mood rating dialogue appeared. The recording of mood rating can be seen in the last line of log file example in Figure 4.4. Note that A-V stand for arousal-valence rating from the participant.

4.4 Results

4.4.1 Preparation of Data

The participants used the logging application for around 8 days on average. All log files created over these days were merged into a single log file. Each participant log file contained on average of 0.1 million lines of event recordings. An application was developed to extract the needed information from the log files. The application extracted self-reported arousal and valence recorded from these log files and keyboard/mouse behaviour within six and ten minute windows around these mood ratings. Figure 4.4 below illustrates the way application extracted the data around the mood ratings within the six and ten minute windows.

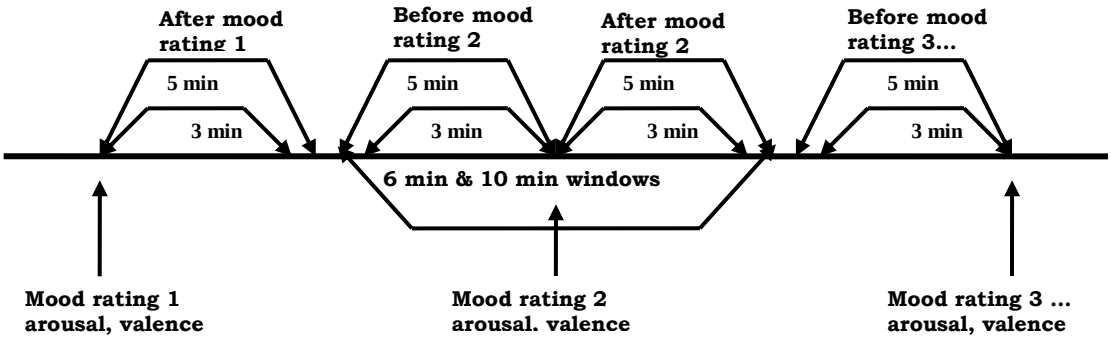


Figure 4.4: *Window for taking events to analyze correlations between events and valence/arousal*

The basic measures taken in each mood rating window were:

1. Self-reported valence and arousal that participants recorded in a mood rating dialogue.
2. Total number of events around a particular mood rating
3. Average time between these events
4. Total windows switched
5. Standard Deviation of the time between events
6. Number of backspace and delete key events

7. Number of alphabetical and numerical key events
8. Number of mouse clicks
9. Number of all other keys

In preparation of the data, window slots were removed where the number of events was fewer than or equal to 10. The threshold value of 10 was selected because participants might not be active interacting with the keyboard and the mouse as they were busy in some other tasks like reading text from websites. Besides ignoring complete data windows, filtering was also conducted on individual event level. For example, one filter was based on the key press and mouse click rates. All the key press and mouse click events with less then 50 milliseconds difference with previous event were filtered. Card et al. (1983) reported that a novice typist have a typing speed of 1000 milliseconds whereas a champion typist has an average speed of 60 milliseconds to type a character. Considering a champion typist a lower limit, 50 milliseconds was set to filter the data having a speed of less than 50 milliseconds. All the key press and mouse clicks with a difference of more than 20,000 milliseconds (20 seconds) with previous event were also filtered out. People waiting more than 20,000 milliseconds to type in a key are likely involved in some other activity besides interacting with the computer.

4.4.2 Analysis

For each participant Pearson correlations were calculated between valence rating and their log variables and between arousal rating and their log variables. Table 4.1 showed that 7 out of 26 (27%) participants had some significant correlations between interaction behaviour and valence rating in a six minutes window. Similarly 7 out of 26 (27%) participants showed significant correlations with arousal in a six minutes window (Table 4.2). The data analysis on the 10-minute window showed that 8 out of 26 (31%) participants had significant valence correlations (Table 4.3) whereas 6 out of 26 (23%) participants had significant arousal correlations (Table 4.4). This suggests that it might be possible to measure moods of some of the participants from their interaction behaviour with keyboard and mouse.

Table 4.1: Correlations between behavioural interaction variables and participants' v*alence rating with* six minutes data around a *valence rating.*

Participants	N	Window Switching	SD Time between events	Backspace delete	Mouse Clicks
1	73			0.26*	
2	49				-0.31*
8	54		-0.30*		
9	56			-0.30*	-0.32*
17	46		0.34*		
19	45	0.32*			-0.31*
20	31	0.38*			

Note: * sign: Correlation significant at 0.05 levels, ** sign: Correlation significant at 0.01 levels, N: Number of mood valence ratings.

Table 4.2: *Correlations between behavioural interaction variables and participants' arousal rating with six minutes data around an arousal rating.*

Partici - pants	N	Average time between events	Windows switching	SD between events	Other keys	Alphabets Number keys	Mouse clicks
1	73				-0.24*		
10	18		-0.50*				
11	69		-0.25*	0.26*		-0.30*	-0.28*
13	37					0.44*	
19	45	0.28*					-0.33*
20	52	0.42*					
25	32						0.35*

Note: ** sign: Correlation significant at 0.05, ** sign: Correlation significant at 0.01 levels, levels, N: Number of mood valence ratings.*

Table 4.3: *Correlations between behavioural interaction variables and participants' valence rating with ten minutes data around a valence rating.*

Partici-pants	N	Events	Window Switching	Average time between events	SD TIME between events	Back-space Delete	Mouse clicks	Others
1	83					0.25*		
3	76		-.24*				-0.31*	
8	56			-0.28*				
9	63	-0.26*			0.34*	-0.34**	-0.39**	
11	72			-0.24*				
14	66		0.28*					
15	73	0.38**		-0.29*	-0.31**	0.26*	0.35**	0.37**
19	57		0.27*				-0.30*	

Note: * sign: Correlation significant at 0.05 levels, ** sign: Correlation significant at 0.01 levels, N: Number of mood valence ratings.

Table 4.4: *Correlations between behavioural interaction variables with participants' arousal rating with ten minutes data around an arousal rating.*

Partici-pants	N	Events	Average Time between events	Window switching	Back-space Delete	Alphabets Numbers	Mouse Clicks	Others
1	83		0.23*				-0.23*	
10	21		0.44*	-0.53*		0.44*		
18	62	0.26*				0.25*		
19	57	-0.32*			-0.30*		-0.36**	-0.28*
20	37		0.37*					
26	77						-0.24*	

Note: * sign: Correlation significant at 0.05 levels, ** sign: Correlation significant at 0.01 levels, N: Number of mood valence ratings.

4.4.3 Regression Analysis

Regression analyses were conducted to see whether individual measurement accurateness could be increased by combining multiple behavioural variables for each participant. Stepwise regression analysis was conducted for individuals that had multiple significant correlations either in the six (Table 4.1 and Table 4.2) or ten minutes window (Table 4.3 and Table 4.4) for both valence and arousal. Table 4.5 and 4.6 shows the participants whose valence can be predicted with two or more behavioural measures in six and ten minute windows. Similarly Table 4.7 shows the participants whose arousal can be predicted with two or more behavioural measures in six-minute windows. No significant multiple linear regression models were found for arousal based on the ten-minute window data.

Table 4.5: Regression model for prediction valence with that of behavioural variables in six minutes window obtained by the use of keyboard and mouse**.**

PID	*R*	R^2	*Adj* R^2	*F*	*Sig.*	*Std. Error*	*Predictor*	*t*	*p*	*B*
19	0.46	0.21	0.17	(2,42) =5.72	0.006	1.8	Number of windows switched	2.5	0.01	0.13
							Number of mouse clicks	-2.4	0.02	-0.01

Table 4.6: Regression model for prediction arousal with that of behavioural variables in six minutes window obtained by the use of keyboard and mouse.

PID	*R*	R^2	*Adj* R^2	*F*	*Sig.*	*Std. Error*	*Predictor*	*t*	*p*	*B*
11	0.38	0.15	0.12	(2,66) =5.72	0.005	1.9	Number of Alphabetic \ Numeric keys pressed	-2.3	0.02	-0.02
							Number of mouse clicks	-2.1	0.03	-0.01

Table 4.7: Regression model for prediction valence with that of behavioural variables in ten minutes window obtained by the use of keyboard and mouse**.**

PID	*R*	R^2	*Adj* R^2	*F*	*Sig.*	*Std. Error*	*Predictor*	*t*	*p*	*B*
19	0.40	0.16	0.13	(2,54)= 5.15	0.009	1.9	Number of windows switched	2.2	0.03	0.08
							Number of mouse clicks	-2.4	0.02	-0.007

Table 4.5 shows that a regression model that included both the 'number of windows switched' and the 'number of mouse clicks' as predictors for valence could explains 21% of the variance in the self-reported valence rating of participant 19. Similar results were found for this participant when fitting a regression model on the 10-minute window as shown in Table 4.7. Prediction of the arousal level was also improved for participants 11 as shown in Table 4.6. The model included as predictors: 'number of alphabetic / numeric keys pressed' and 'number of mouse clicks' which was able to explain 15% of the variations of self-reported arousal level. The result of these regression analyses indicates that for some people it is possible to increase accurateness of their mood measuring by considering multiple behavioural measures.

4.4.4 Conservative Analysis

This study analysed eight different behavioural variables. To test whether all these variables differ significantly, a comparison (post-hoc) test was conducted to insure that possibility of committing a Type I error does not exceed pre-specified alpha value that is reasonably low (in this study $\propto = 0.05$). Introduction of tougher alpha levels control potential biases. Although post-hoc analysis is mostly used in case of ANOVA, this study looked on the possibility of using more conservative levels in case of correlations also.

Researchers are not in total agreement on the suitable way to use these comparisons (Sheskin, 1996). In planned analysis, comparisons are decided in the beginning of the study. In post-hoc tests data analysis is conducted to explore which groups or variables contributed statistically to significant results (Maxwell and Delaney, 2004). Following equation helped to decrease the possibility of committing a Type I error for multiple comparisons:

$$\propto_{Pc} = 1 - \sqrt[C]{1 - \propto_{FW}} = 0.006$$

Here $\propto_{FW} = 0.05$ and C = 8 as number of behaviour variables is eight. The note $\propto_{FW}$ represent family wise type I error rate and is the chance of at least one Type I error in C comparisons. $\propto_{PC}$ is a comparison Type I error rate and is the chance of a Type I error in any single comparison. By substituting values in equation one, we get $\propto_{pc}$ of 0.006. Although reducing the alpha value decreases the possibility of committing a Type I error, it increases the possibility of committing Type II error. Table 4.8 shows the significant correlation on 0.006 levels.

Table 4.8: *Behavioural variables with significant correlations at 0.006 level with valence and arousal.*

PID	N	Mood Dimension	Time Window	Backspace Delete		Mouse Clicks		Number of events		Other keys	
				r	*p*	*r*	*p*	*r*	*p*	*r*	*P*
9	63	valence	10 Min	-0.34	0.006	-0.39	0.001				
15	73	valence	10 Min			0.35	0.002	0.38	0.001	0.37	0.001
19	57	Arousal	10 Min			-0.36	0.006				

Table 4.8 shows that it is unlikely that the significant results obtained from some participants were obtained by chance alone. An interesting observation here is that for all the three participants had a correlation that involved the number of mouse clicks.

4.4.5 Logistic Regression

The analysis found significant correlations between the mood rating and the behaviour for some of the participants only. Therefore the next step of the analysis was to determine potential factors that could help identifying people for which interaction-based mood measuring might be possible. Two Logistic Regression analyses were conducted, one with as dependent variable whether a significant correlation was found between a person's arousal rating and his or her behavioural measures. Other logistic regression analysis with as dependent variable whether a significant correlation was found between a person's valence rating and his or her behavioural measures. The analyses used a logistic regression forward method with likelihood ratio. Independent variables were gender, user type as categorical variables. Independent variables also included age and personality scores on the five personality

main traits and 30 personality sub traits. To get the personality information 20 out of 26 participants completed a short version of the IPIP-NEO personality test (Buchanan et al., 2005).

The significant (χ^2 (2, N = 20) = 9.56, $p < .01$) logistic regression model for valence showed that 85% of participants showing correlations could be correctly classified. Table 4.9 shows variables that might be useful in identifying these individuals. The predicting variables include experience of computer use in number of years approaching an alpha level of 0.05 ($p = 0.06$) and self-discipline. Table 4.10 shows some explanation how predictions can be made. From Table 4.10 it is evident that as experience of the participants' increases it is more likely that their valence correlated with their interaction behaviour. Similarly it might be more difficult to measure valence from the keyboard mouse use of disciplined participants compared to participants with less self-discipline.

The arousal logistic regression model (Table 4.11) shows that 65% of participant could be correctly classified, which is significant (χ^2 (1, N = 20) = 7.05, $p < .008$). The model includes only dutifulness as significant predictor, which explains that participants with more sense of dutifulness are more likely to have significant correlations between their arousal and their use of keyboard and mouse.

Table 4.9: *Summary of logistic regression analysis for variables predicting participants that might show correlations in valence; Have correlation (n = 11) and No correlations (n = 9).*

Predictor	*B*	*df*	*SE B*	e^B
Experience	0.931	1	0.49	2.54
Self Discipline	-0.1*	1	0.05	0.90
Constant	1.23	1	2.41	3.42
χ^{2**}		9.56		
Df		2		
% Correct predictions for participants having correlations			50%	
% Correct predictions for participants with no correlations			35%	
Overall Correct predictions			85%	

Note: *e^B= exponentiated B, *p < 0.05, **p < 0.01*

Table 4.10: *Predictor variables showing possibility of predicting participants who might show correlations or no correlations with use of keyboard and mouse and their valence level.*

Predictor Variable	Correlation (Yes/No)	Mean	Standard Deviation	Out Comes
Experience in years	No	4.5	0.41	More experience might result in correlations
	Yes	5.6	0.53	
Self Discipline	No	64.33	5.70	Not Disciplined to Self Disciplined (1 to 100)
	Yes	48.92	4.92	

Table 4.11*: Summary of logistic regression analysis for variables predicting participants that might show correlations in arousal; Have correlations (n=8) and No correlations (n=12).*

Predictor	*B*	*Df*	*SE B*	e^B
Dutifulness	0.07*	1	0.03	1.07
Constant	-4.26*	1	1.93	0.01
χ^2**	7.05			
Df	1			
% Correct predictions for participants having correlations			20%	
% Correct predictions for participants with no correlations			45%	
Overall Correct predictions			65%	

Note: e^B*= exponentiated B, *p < 0.05, **p < 0.01*

Table 4.12: *Predictor variables showing possibility of predicting participants who might show correlations or no correlations with use of keyboard and mouse and their arousal level.*

Predictor Variable	Correlation (Yes/No)	Mean	Standard Deviation	Out Comes
Dutifulness	No	45.50	6.12	undutiful to dutiful 1 to 100
	Yes	71.63	6.54	

4.5 Conclusions, Limitations and Future Research

The results of this study suggest that it might be possible to measure computer users' mood from their use of keyboard and mouse. There were some significant correlations, which Cohen (1992 b) would classify as medium or large effects, supporting this. Therefore results partially support H_2, which indicate that measuring moods of a computer user might be possible from their interaction behaviour with keyboard and mouse. As not every participant data show similar correlation patterns, therefore these findings cannot be generalized even in this opportunity sample. However the significant correlations found for some participants could be promising for future extensive experimentation. Results also revealed that it might be possible to predict whether or not a participant would show significant correlations between his\her mood and the use of keyboard and mouse.

There were some limitations also in the study. Low standard deviation differences between participants' mood ratings showed that their moods were of less intensity on both arousal and valence dimension. This might be because that natural condition was used without experimentally controlled mood induction. The figures 6 and 7 show that mean of valence is 4.80 with an SD of 0.94 and that of arousal is 5.19 with an SD of 1.15 which is near 5 the centre point of both ratings.

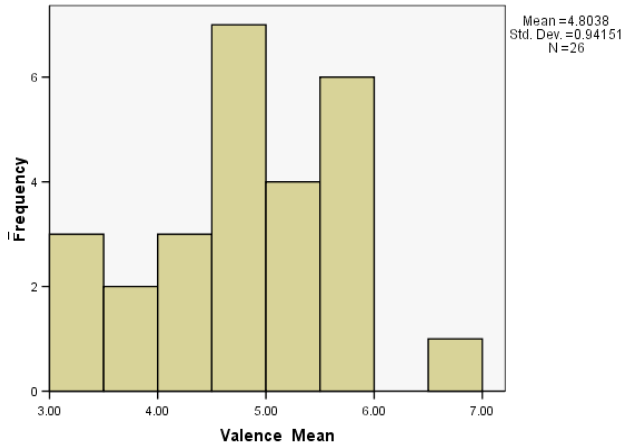


Figure 4.5: *Valence Histogram*

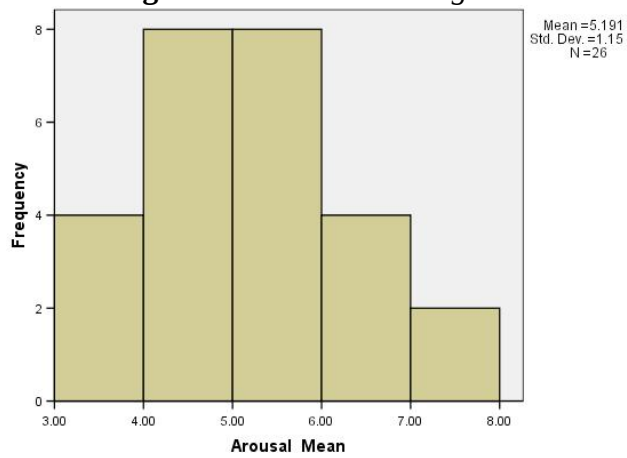


Figure 4.6: *Arousal Histogram*

Another limitation that two of participants mentioned is related to the mood rating dialogue. They considered the mood rating dialogue a distracter as it was designed to appear after every 20 minutes. Yet another limitation could be cancelling the mood rating dialogue by participants in an extreme mood state and therefore possibly less willing to answering the dialogue box. Calculations showed that participant cancelled 27% of their dialogues. Therefore a future study was required which might consider a similar experiment in lab conditions to overcome this limitation of field study. Hence the experiment conducted to overcome this and some other limitations will be explained in the next chapter. Another limitation was that only two female participants took part in this study which is clearly an under representation of the female population. Future work from this study and the study explained in the next section is collectively stated in the conclusions of the chapter 5.

The results from this chapter revealed that it might be possible to measure some of the programmers' and computer users' moods from their interaction behaviour with keyboard and mouse. However these participants form only 37% of

the sample containing 26 participants. These results therefore are not encouraging enough. There could be various reasons for this as explained above; one of them could be that participants stopped recoding their moods whenever their arousal or valence was at peak in both directions. This was evident from the number of mood rating dialog boxes that were cancelled. A further experiment was planned to induce moods and to measure moods without requiring participants to rate them. In other words mood rating was not interruptive. The next chapter explain the design, material, analysis and results of this experiment.

Chapter 5: Measuring Mood by the use of Keyboard and Mouse: Galvanic Skin Response Scenario

5.1 Introduction

This chapter contains results about an experiment carried out to further validate the results obtained in Chapter 4. The results from Chapter 5 seem also to suggest that it might be possible to measure the mood from keyboard and mouse use for some individuals. Therefore this chapter again strengthens the support for the second hypothesis of this research stating, *"Moods can be measured from the computer user's interaction with keyboard and mouse"* which will be represented as H_2.

The study presented in Chapter 4 used self-reported mood ratings during everyday use of a computer over a period of several days. Keyboard and mouse events were recorded in a log file with a 20-minute gap between mood rating dialogs. In this experiment self-reporting was replaced by a galvanic skin response meter to measure response of the skin, which in turn is related to moods and emotions. The benefit is that this makes the measurement more objective. This study continued for only an hour. In the previous study no mood induction was applied and participants rated their moods under general working conditions whereas in this study moods were induced with high arousing and low arousing music.

GSR (Galvanic Skin response) is a physiological method in which response of the skin is measured between two points. A fair amount of literature exists presenting various approaches toward user mood and emotion measurement including psycho physiological measures. However as discussed earlier, most of the methods have an interruptive or obstructive element and therefore are not suitable for mood measurement of programmers in actual daily life setting. Therefore GSR is only used in this chapter as a baseline mood measure to examine the possibility of establishing a mood measure that is based on keyboard and mouse interaction data. This chapter with Chapter 4 will lay the basis for the creation of a tool that can be embedded in Integrated Development Environments (IDEs) for measuring programmer mood and to help programmers to improve their performance.

The second section of this chapter starts by reviewing research in the support of mood measurement from psycho physiological measures. The third section

explains material used in the experiment like GSR meter and application for measuring skin responses as well as an application for logging keyboard and mouse interaction data. Selection of the mood inducing music tracks is also explained in the third section. Participants and experimental setup are explained in the last subsection of section three. Analysis and results are discussed in the fourth section of this chapter. The fourth section also discusses data preparation and data smoothing along with basic analysis carried out on the data. The fourth section also contains discussions on a conservative analysis and logistic regressions to identify personal characteristic that could indicate whether keyboard and mouse data from specific individual can be used to measure their mood. The last section of this chapter deals with conclusions, limitations, and future research.

5.2 Literature Review

Affective computing is one of the emerging fields of computer science and is yet in its early phases (Picard, 2000). Efforts are being made now to develop affect recognizing computers which can help in decision making, reasoning and emotional interactivity. Various affect recognizing methods are being developed by researchers in the field of computer science like affect recognition from video responses and images, body movements and postures and psycho physiological measures like blood pressure measurement, heart beat, and skin response.

Psycho physiological measures like electroencephalograms (EEG) or electromyograms (EMG) are used to study unconscious functions of human body like affects (Sakurazawa et al., 2003). According to Backs and Boucsein (2000) three-arousal model devised by Gray (1975) and further refined by Fowles (1980) may be used as indicator for respective arousal system. The three-arousal model is comprised of affect arousal, effort and preparatory activation and is termed as Arousal System 1, Arousal System 2 and Arousal System 3 respectively. They further argued that "Amygdala[11]" is mainly responsible for Arousal System 1. On a certain change or on stimulation "affect arousal" can be measured from phasic heart rate (HR) or amplitude of Electro-dermal response (EDR).

There is a substantial amount of research which indicates an increase or decrease of various psycho physiological measures like additional heart rate,

[11] A part of the brain responsible for arousal, autonomic response related with fear, emotional responses and Hormonal secretions etc. (Fink, G. (2000). Encyclopaedia of Stress, Published by Academic Press, p 247

respiration rate, finger pulse volume amplitude, eye blink rate and finger temperature under conditions like strain (Backs and Boucsein, 2000). There is also field, simulation and laboratory research available on psycho physiological measures and on the level of arousal. For example Gundel et al. (1995) conducted a field study and found an increase in EEG alpha rhythm de-synchronization when pilots feel sleepy. With respect to bio-signals from skin various researchers focused on emotional aspects like Boucsein and Thum (1997). They found an increase in NS.SCR (non Specific Skin Conductance response) frequency while examining emotional strain during prolonged interruption of patent examiners.

Electro-dermal activity (EDA) is considered to be the most convenient measure taken from the skin and is measured as SCR (Skin Conductance Response) or as SRR (Skin resistance response) (Backs and Boucsein, 2000). Researchers found psychophysiology parameters like increase in heart rate (HR), respiration rate related with human-computer interaction (Haider et al, 1982; Gao et al., 1990) and computer related data entry work (Schleifer and Ley, 1994). However skin response is also used widely for studying emotional responses related to computers. For example Sakurazawa et al (2003) utilized galvanic skin responses in computer gaming. Lissetti and Nasoz (2002) proposed an interface to sense a user's emotional and affective states by using visual (images, videos), Kinesthetic (Automatic nervous system signals) and Auditory (speech).

Although there is research available on galvanic skin response (GSR) in relation to different aspects of computing and measuring affects, research on studying GSR and use of keyboard and mouse for the purpose of mood measurement is still lacking. This chapter will target this gap as well as strengthen the results obtained from the previous Chapter 4 that indicates that it might be possible to measure moods of some of the people from their use of keyboard and mouse. The next section will explain the experimental setup applied in this experiment, followed by a discussion of results and any possible outcomes. The last section will explain conclusions, limitations in the research and the future research.

5.3 Experimental Material and Design

The basic design of the experiment is similar to the earlier experiment explained in Chapter 4. The keyboard and mouse events were being recorded in a log file. However instead of self reported mood, skin resistance measurements from the

GSR meter were being recorded in the form of images with date and time stamp. Mood inducing music was used in background to induce moods. The aim was to find correlations between keyboard and mouse events and GSR measurements. However to confirm the validity of GSR as a mood measure, the impact of mood inducing music was also analysed on the GSR ratings. A 2x2 factorial design was considered with arousal as independent variable with two levels High and Low and music type (Used as literature indicates vs. brought by participants) as an independent variable. GSR measurements were used as dependent variable. The material used in this design is explained below:

5.3.1 Galvanic Skin Response Meter

For measuring skin response of participants this study utilized a GSR meter. This is a psychometric research biofeedback monitor designed to measure the electrical conductivity of the skin. To measure conductivity, this meter passes a small unnoticeable current through one finger electrode to another. Electrical activity of skin is affected by various factors like: 1) changes in salt in the sweat gland duct and 2) changes in water in the sweat gland duct. The changes are a response to certain emotional reactions. Increase in the activity (high arousal) results in the increase of electrical conductivity whereas relaxations (low arousal) results in the decrease of electrical conductivity. The output shown by the meter is the conductance in units called micro Siemens (µS). The attachment of the GSR meter with the fingers is illustrated in Figure 5.1.

Figure 5.1: *The GSR Meter and it electrodes attaching position with hand as* used in the experiment. The output shows a measurement of 31.4 micro Siemens (µS)

5.3.2 GSR Recording and Keyboard Mouse Interaction Logging Application

The idea behind the experiment was to record skin response as well as keyboard and mouse usage of participants in parallel. A GSR meter was used to measure galvanic skin response by attaching electrodes to the base of the left hand middle finger and ring finger. As the GSR meter used in this research had no facility to record measurement directly into the computer, a separate application was required and therefore developed for this task. In addition keyboard key press and mouse clicks logs were also required. The application used in Chapter 4 was modified to serve this purpose. First the application recorded measurement from the GSR meter by taking an image of the GSR measurement every 10 seconds and saving it with date and time. The other application recorded keyboard key press and mouse clicks. Original data typed were not recorded because of the privacy reasons. Instead the following coding was used:

1) Keys 'A-Z' were recorded as 'Capital Alphabet'.
2) Keys 'a-z' were recorded as 'Short Alphabet'.
3) Number keys as 'Numeric Key'
4) Special characters' as 'Special Character'
5) All other keys were recorded in their original context like 'Home', 'Page up', 'Page Down' etc.

In addition to key press and mouse click events, other data recorded in the log files were date and time of the event, the window in which participants were working on and the type of event (Keyboard key press or mouse Click').

5.3.3 Mood Induction

For mood induction two audio clips, one audio clip for high arousal and one for low arousal were selected. The two audio clips, Mozart Sonata for two pianos K448 (1985, track 1) for high arousal and Adagio by Albinoni (1981) for low arousal, used in this research were validated for their arousal induction nature by Thompson et al. (2001). Thompson et al. observed that performance of the participants who listened to the music by Mozart (high arousal music) was better in spatial tasks as compared to silent condition. They further observed that both piece of music induced different responses on enjoyment, arousal and mood measures. Silvia and Abele (2002) suggested the neutral audio clip "Hymn" by Moby (1995) and this was also selected.

In addition participants were instructed to bring their own high and low arousal audio clips, which they thought would put them in a low or high arousing state. Participants were not asked to bring neutral music.

5.3.4 Participants

Experiment took place in each individual's home with a quiet room. The basic aim of this setup was to avoid the anxiety of coming to the lab. In addition it was assumed that they would be more relaxed in their own home environment. Another reason was the lack of participants' time to come to the lab. Therefore it was more convenient to visit to participants' preferred place as well as keeping the same experimental conditions of quiet room with no interruptions. A total of 16 participants participated in this study. All the participants were male. Participants in this study were computer users with a mean age of 26 years and with standard deviation of 3.1. The participants had an average computer usage experience of 7.3 years with standard deviation of 2.8 years. Among participants 38% rated themselves as medium computer users, 31% rated themselves as expert computer users and 31% rated themselves as programmers.

5.3.5 Experimental Setup

The basic aim of the experiment was to record GSR measurements taken by GSR meter in parallel with programmers doing programming tasks. In addition it was also an aim to record their keyboard and mouse interaction data to find a relationship between the GSR readings and keyboard and mouse interaction data. Consent was obtained from participants before conducting the experiment. Consent explained all the procedure and equipment used in the experiment.

The participants were asked to complete a number of programming tasks in Alice. Alice is an object-oriented language used to teach programming to children and was developed by Carnegion Mellon University. It is free to use and has some interesting aspects like visual object and visual construct and structures. It can be downloaded from www.alice.org . All participants completed four learning tutorials present in Alice Interface. The tutorials were about using Alice and its various functionalities as well as introduction about simulating stories. Two Aesop stories ('The Hare and the tortoise' and 'the cow and the Frog') were assigned as tasks to each participant.

Experiment was composed of two sessions of 30 minute with a 10 minute break between them. Each session was further composed of three sub sessions. One sub-session was of neutral music, and two sub-sessions of either high arousal or low arousal music. High or low arousal music sub-sessions were counter balanced. Similarly personal music and music from the literature were also counter balanced. Neutral music sub-session started with five minutes of no music at all followed by five minutes of neutral music. This was followed by either high arousal (brought by participants or taken from the literature in sequence one after another) or low arousal (brought by participants or taken from the literature in sequence one after another) music. There was a break of 10 minutes or more between two main sessions. The sequences of sessions as well as music sequences are illustrated in Table 5.1 below.

Table 5.1: *The four session/music sequencing permutation participants were allocated to.*

Session 1 (30 min)				**Break**	**Session 2 (30 min)**			
Sub-Session 1 (10 min)		**Sub-Session 2 (10 min)**	**Sub-Session 3 (10 min)**	**(10 minute) break**	**Sub-Session 1 (10 min)**		**Sub-Session 2 (10 min)**	**Sub-Session 3 (10 min)**
NoM 5min	NeM 5min	HAP	HAL		NoM 5min	NeM 5min	LAP	LAL
NoM 5min	NeM 5min	LAP	LAL		NoM 5min	NeM 5min	HAP	HAL
NoM 5min	NeM 5min	HAL	HAP		NoM 5min	NeM 5min	LAL	LAP
NoM 5min	NeM 5min	LAL	LAP		NoM 5min	NeM 5min	HAL	HAP

Note: *NoM = No Music, NeM = Neutral Music, HAP = High Arousal Brought by person, LAP = Low arousal brought by person, HAL, High arousal indicated in literature, LAL = Low arousal indicated in literature.*

This music was played in the background on the participants' headphone. In this way they were able to listen to the music while completing the tasks. The use of a headphone along with using a separate room made sure no external interruptions affected the participants. Participants keyboard key press and mouse clicks were also logged in a file during the sessions along with their galvanic skin response from the GSR meter.

5.4 Results

5.4.1 Data Preparation

Experiment was divided into two sub sessions separated by a gap of 10 minutes or more. The whole experiment took approximately an hour and ten minutes. Galvanic skin responses (GSR) were recorded every 10 seconds in both of the sessions, thus recording a total of about 180 responses in each session of 30 minutes with an average of 60 responses in each of the neutral, personal and literature music sessions. Similarly the second session also had 180 responses forming a total of 360 responses from a single participant. The intention of the experiment was to find correlations between the GSR responses and the use of keyboard and mouse. There were a number of 10-second time slots with no keyboard or mouse data.. This could be the result of participants involved in a thinking process or using pencil and paper and not actually interacting with the keyboard or mouse. Therefore, it was decided to take the average of GSR responses in a minute and also take average of behavioural data recorded in that minute. This resulted in 30 responses from each sub-session forming a total of 60 responses for experiment. An application was developed to count and analyse all the behavioural data within the target minute of GSR response. The behavioural variables considered for analysis were number of events, time between events, standard deviation of time between events, number of windows switched, total alphabetic and numeric keys pressed, total mouse clicks, total backspace/delete keys pressed and total of all other keys pressed within that time. Each GSR measurement was than linked to the average of each behavioural measure recorded within that minute of GSR measurement. This resulted into a total of 60 GSR responses each with their associated eight behavioural measurement values.

The purpose of GSR meter was to record electrical conductivity of the skin, which changes according to changes in emotional reactions. However when initially connected with the fingers, the reading started from minimum and gradually increased up representing internal response. Similarly a rapid and fast movement of hand also caused an increase or decrease in measurements. To reduce this effect the first step carried on the data was to smooth it. This was done using a macro function called "Smooth"[12]. This function performs all the standard smoothing methods of exploratory

[12] http://www.quantdec.com/Excel/smoothing.htm

data analysis explained by Tukey (1977) with a high degree of flexibility. A command passed to the smoothing function to smooth was "3R[13]S[14]SH[15]". Smoothing is included in the Exploratory Data Analysis (EDA) and is used in various studies of psycho-physiological nature to reduce the noise (e.g. Baumgartner et al., 2000).

5.4.2 Analysis

The first part of the analysis was to test whether the music chosen from the literature as well as of participants' choice had an effect on participants' GSR. Each participant's average GSR score in high arousal or low arousal session was calculated. A relative score was calculated by subtracting GSR measurements obtained when participants were listening to high arousal or low arousal music from the GSR measurements when participants were listening to neutral music or no music at all. The equation for this can be represented as follows

$$GSR_{Relative} = GSR_{Neutral} - GSR_{Low/High\ Arousal}$$

A repeated measure MANOVA was conducted to assess if there was a difference between high arousal and low arousal music, and music selected from the literature or brought by the participants. No significant main effect (F (1, 16) = 2.30, p = 0.15) was found for high arousal versus low arousal. This means that together these two types of music did not induced uniformly either high or low arousal. However a significant effect (F (1, 16) = 13.43, p = 0.002) was found for the music type. Examining Figure 1, it seems that on average the music from the literature induced an arousal level close to the arousal level obtained in the neutral sessions (high arousal with a relative mean of 11.3 µS and a low arousal with a relative mean of -4.75 µS), whereas the music brought by the participants resulted in much more high arousal (high arousal with a relative mean of -78.4 µS and low arousal with a relative mean of -24.4 µS). Also the analysis found a significant interaction effect (F (1, 16) = 13.43, p = 0.002) between music type and arousal level. It seems that only the music brought by the participants significantly induced the expected relative levels of arousal. The following analyses

[13] When a median smooth is immediately followed by an "R" (repeat) command, then continue to apply the median smooth until no more changes occur

[14] "Split" the sequence. This process dissects the sequence into shorter subsequences at all places where two successive values are identical, applies a 3R smooth to each subsequence, reassembles them, and polishes the result with a 3 smooth.

[15] "Hann" the sequence: This is convolution with a symmetrical kernel having weights (1/4, 1/2, 1/4). That is, each value x[i] is replaced by x[i-1]/4 + x[i]/2 + x[i+1]/4. The end values are not changed.

therefore only included data obtained from the sessions in which the music was brought by the participants.

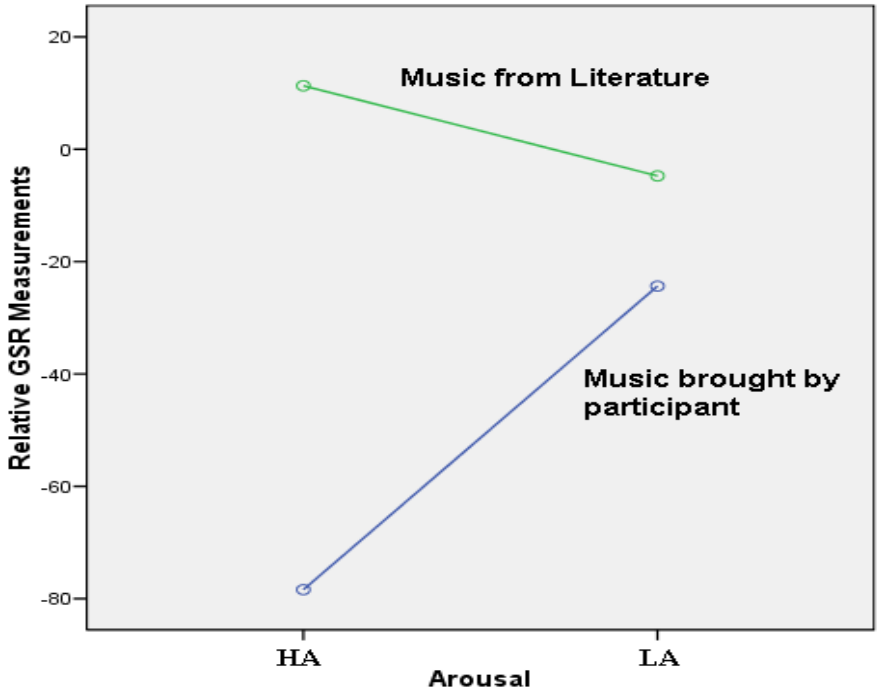


Figure 5.2: *Relative Galvanic Skin Response ($GSR_{relative}$ = $GSR_{neutral}$ – $GSR_{low/high}$) in low (LA) and high (HA) arousal conditions. Upper line represents music from the literature whereas line near the bottom represents music brought by person.*

The next step was to see whether a similar effect would be found in the behavioural measures. This would give an insight into whether a uniform mood measure across the participants would be possible. A repeated measure MANOVA was conducted with as with-subjects variable arousal —the two levels for music brought by the participants. The dependent measures were the behavioural measure derived from keyboard and mouse data, which were:

1. total number of events around a particular mood rating
2. average time between these events,
3. total windows switched
4. standard deviation of the time between events
5. number of backspace and delete key events
6. number of alphabetical and numerical key events
7. number of mouse clicks
8. number of all other keys

The analysis found no significant main ($F(8,8) = 1.03$, $p = 0.48$) effect for arousal on the behavioural keyboard and mouse data. The univariate analyses also did not show a

significant effects of arousal on the number of alphabetic and numeric keys pressed ($F(1,15) = 1.42$, $p = 0.08$) but only a trend toward that. This suggests that arousal difference did not have a clear uniform effect on keyboard and mouse behaviour across the participants. Therefore the following analysis focussed on the idea of individualised mood measures. Correlations were examined between high/low arousal music brought by the participants and the behavioural measures from keyboard and mouse. For each participant Pearson correlations were carried out between GSR responses and the behavioural variables for all those 20 average GSR measurements over a minute where music induced high or low level of arousal and the music was brought by participants. 50% (8 out of 16) of participants showed significant correlations of GSR measurements with that of keyboard and mouse interaction variables as shown in Table 5.2.

Table 5.2: *Correlations of different behavioural variables with participants GSR responses*

PID	N	Avg Time b\w Events	Window Switched	SD Time between events	Backspace delete keys	Others	Mouse Clicks
3	20		0.5*				
5	22			0.42*			
7	20				0.59**	0.6**	0.6**
8	20		0.45*	0.47*			
9	20	0.52*					
11	20			0.49*			
12	20		0.48*				
16	20	0.67**					

*Note: **sign- Correlation significant at 0.01 levels, *sign- Correlation significant at 0.05 levels, N- Number of mood valence ratings*

As in Chapter 4, regression analyses were conducted to see if a combination of behavioural measure could increase mood prediction. The regression analyses used a stepwise method for selecting the predictors in the model. GSR measurements were modelled as a function of keyboard and mouse interactions for each participant having multiple significant correlations between GSR measurements and keyboard/mouse interaction (PID 7 and 8). However, the analyses did not result in significant models with multiple predictors. This suggests that for these participants with these behavioural measures, combining the behavioural measure did not improve mood prediction beyond than already presented in Table 5.2.

5.4.3 Conservative Analysis

As in the experiment described in Chapter 4, eight different behavioural variables were used for the post hoc analysis to test whether all these variables differ

significantly. Post hoc tests ensure that possibility of committing Type I error does not exceed pre-specified alpha value that is reasonably low (for example $\propto = 0.05$). More detail about Post hoc analysis can be found in Chapter 4. The formula to calculate a conservative level is:

$$\propto_{Pc} = 1 - \sqrt[C]{1 - \propto_{FW}} = 0.006$$

Note: *Here $\propto_{FW} = 0.05$ and C = 8 as number of behaviour variables is eight.*

Table 5.3: *Behavioural variables with significant correlations with GSR measurements at 0.006 level.*

P-ID	N	Backspace Delete		Mouse Clicks		Average time between events		Other keys	
		r	*P*	*r*	*P*	*R*	*p*	*r*	*P*
7	20	0.6	0.006	0.6	0.005			0.6	0.005
16	20					0.67	0.001		

Table 5.3 above shows that the significant results obtained from participants 7 and 16 is unlikely to be obtained by chance alone. With correlation around 0.60 and 0.67 is seems that for these participants it is possible to explain 37% to 49% of the variation in their GSR measurement, which suggest that behavioural data can give some insight into the mood state of some participants.

5.4.4 Logistic Regression

Analysis above found significant correlation between arousal level and behaviour for 8 out of 16 participants. One question of interest could be why only some of the participants' correlations were significant and how these individuals can be identified. Therefore the next step of the analysis was to determine potential factors that could help to determine the likelihood that the mood of a specific person could be measurable by looking at his or her keyboard and mouse use. Logistic regression analyses were conducted with as dependent variable whether or not a significant correlation was found between a person's arousal rating and his or her behavioural measures. Independent variables were gender and user type as categorical variables. Independent variables also included age and personality scores on five personality main traits and 30 personality sub traits. To get personality scores 15 out of 16 participants completed a short version of the IPIP-NEO personality test (Buchanan et

al., 2005). The analyses used a logistic regression forward method with likelihood ratio.

The model from the analysis showed that 80% of the participants could be correctly classified with a significance of (χ^2 (4, N = 15) = 20.73, $p < .001$). Table 5.4 shows variables that might be useful in the classification. The predicting variables include personality traits cooperation and achievement striving but also computer users' expert and programmer groups.

Table 5.4: *Summary of logistic regression analysis for variables predicting participants that might show correlations; Have correlation (n = 7) and No correlations (n = 8)*

Predictor	*B*	*Df*	*SE B*	e^B
Cooperation	0.12*	1	413	1.13
Achievement Striving	-1.56	1	470	0.90
Type of Computer User (Expert)	23.65	1	71654	2^e+010
Type of Computer User (Programmer)	-15.18	1	72390	0.00
χ^{2**}		20.73		
Df		4		
% Correct predictions for participants having correlations			71%	
% Correct predictions for participants with no correlations			88%	
Overall Correct predictions			80%	

Note: *e^B= exponentiated B, *p < 0.05, **p < 0.01*

Table 5.5: *Predictor Variables showing possibility of predicting participants who might show correlations or no correlations with use of keyboard and mouse and their valence level*

Predictor Variable	Correlation (Yes/No)	Mean	Standard Deviation	Out Comes
Cooperation	No	38.75	17.52	Not Cooperative to Cooperative (1 to 100)
	Yes	68.43	12.50	
Achievement Striving	No	39.63	22.37	minimal workers to hard workers (1 to 100)
	Yes	18.00	13.26	
Type of Computer User (Expert)	No	4	-	80% Experts might not show correlations whereas 20% might
	Yes	1	-	
Type of Computer User (Programmer)	No	2	-	60% programmers might show significant correlations whereas 40% might not
	Yes	3	-	

Table 5.5 show some explanation how prediction can be made. For example, participants who had no significant correlation were less cooperative as indicated by the lower means whereas participants with higher means were more cooperative. Participants that scored high on achievement striving were less likely to have significant correlations than participants that scored low on achievement striving. Computer users were divided into three types: Medium, Expert and Programmer. Logistic regression included experts and programmers in prediction model. According

to this model 80% of experts in the sample did not show any significant correlations as compared to 20% who showed significant correlations. 60% of programmers showed significant correlations as opposed to 40% programmers who did not show significant correlations.

5.5 A Comparison of Mood Measurement Methods

There are various methods available to measure moods such as: 1) Psychological measures 2) Psycho-physiological measures and 3) Behavioural measures. Table 5.5 compares these mood measuring methods based on different criteria. These comparisons are not suggesting which measure is better as suitability of these measures depend on the situations. Some of the comparison criteria given below in the table are not based on the literature and therefore are subject to further thorough research. Each criterion is represented by a unipolar scale. Unipolar scales start from minimum to maximum. For example criterion 'Ease of Implementation' could be classified as 'not easy', 'easy', 'medium' and 'difficult'.

Table 5.5: *Comparison of mood measuring methods*

Criteria	**Psychological (Self reporting)**	**Physiological**	**Behavioural**	
			Image Processing \ others	**Keyboard \ mouse**
Ease in implementation	Easy	Difficult	Difficult	Medium
Time requirement to measure mood	Medium (more accurate need more time)	Low	Low	High
Maturity	Highly Mature	Mature	Mature	Not mature
Dependence on human perception	High	Low	Low	Low
Invasive	Most often	Mostly invasive	Most Non Invasive	Non Invasive
Difficulty in data gathering	Low	High	Low	Medium
Complications in mapping to emotional constructs	Low	High	Medium	High
Hardware Requirement	No	Yes	Yes	No
Accuracy	Accurate mostly	Most researchers term it accurate	Mostly accurate	Not accurate (immature)
Obtrusive \ obstructive	Yes	Yes	No	No

Explanations of the comparison criteria given in the table above are as follows:

1. **Ease in implementation:** This criterion explains which of the mood measuring method is easy to implement. Psychological involve self reporting and need human input, therefore is easy to implement. However psycho-physiological and behavioural (Image processing) mood measures are difficult to implement because of extra hardware requirement as well as complex algorithms to capture data. However mood measuring from keyboard and mouse is moderately difficult as there is a need to write a background application to constantly log and monitoring.
2. **Time requirement:** Psychological measures can generate quick outputs. However quick measures contain lesser items to rate and therefore are not always accurate. Psycho-physiological and behavioural (Image processing) methods require less time to measure mood provided that equipment is set beforehand. Keyboard and mouse method however, require more time as data need to be logged for some time before software starts mapping interaction with mood.
3. **Maturity:** Maturity here is taken in context of how well researched a particular method is? Psychological, psycho-physiological and image processing methods have long history of research. However mood measurement from keyboard and mouse is very immature in this context.
4. **Dependence on human perception:** A psychological measure depends on human perception (Cook and Campbell, 1979) as well as cognitive elements like memory (Schacter, 1999). However other three methods are relying on objective observations and are not subjective.
5. **Invasive:** Psychological measures are invasive as participants directly rate their moods. Psycho-physiological measures are also invasive as there is a need to attach equipment with participants. However other two methods are not invasive.
6. **Difficulty in data gathering:** It is easy to collect data using Psychological measures depending upon availability of participants. Similar is true with image processing and keyboard and mouse interaction methods. However psycho-physiological method involves hardware that requires constant monitoring or special monitoring applications and therefore data collection is relatively difficult.
7. **Complications in mapping to emotional constructs:** Psychological methods are relative easy to map with mood constructs. Similar is also true for image processing methods. However mapping data from psycho-physiological and keyboard\mouse interaction seem more difficult to map with moods.

8. **Extra hardware requirement:** Psychological and keyboard\mouse interaction methods require no extra hardware. However other two methods require extra equipment like GSR meters or cameras.
9. **Accuracy:** Psychological are most accurate in mood measurement. Psycho-physiological and Image related depend on the circumstances and control and could be accurate to highly accurate. Keyboard/mouse interaction method as is not mature therefore its accuracy is in doubt.
10. **Obtrusive\obstructive:** Psychological and psycho-physiological are obtrusive\obstructive as they require rating or other hardware attached with participant. Other two methods do not require such connections and therefore are not obtrusive\obstructive.

The comparison above indicates advantages of different mood measuring methods in different conditions. As indicated earlier, no method is superior over other and a single method or combination of methods can be selected based on criterion, requirements and scenario. As mood measurement from interaction behaviour of keyboard and mouse is very immature, yet comparison indicates it is of low cost as it requires no extra hardware and therefore is also not obtrusive and obstructive. It does not depend on human perception thus could depict correct internal state of moods. It is easy to implement and data gathering is also not difficult to implement in this context. Its accuracy might be increased by the use of trained neural networks.

5.6 Conclusions, Limitations and Future Research

The results of this experiment seem consistent with the experiment presented in Chapter 4 and suggest that for some computer users it might be possible to measure their mood from their use of keyboard and mouse. There were some significant correlations which Cohen (1992 a) would classify as medium and large effects to support this. In the study described in Chapter 4 a maximum of 31% participants showed significant correlations. In this experiment these statistics seems to have improved to 50% of the participants. Again as few of the participants did not show significant correlations therefore these findings cannot be generalized even in this opportunity sample. However the significant correlations found for some participants could be promising for future experimentation. Conservative analysis indicated that the correlations obtained from at least 2 participants in this experiment were unlikely to be the result of chance alone. Furthermore various common data items were found

to be significantly correlating with GSR measurements in both experiments 4 and 5. The results of the logistic regression analyses point towards the possibility of predicting whether a person would show significant correlations with GSR measures and their use of keyboard and mouse or not.

The first limitation of this study was the very restricted sample of participants. All the participants in the study were male thus there was no representation of females. Other limitations related to psycho-physiological measures as discussed by Gillard and Kramer (2000). For example, a disadvantage of psycho-physiological measurement is the large variability in responses. An attempt was made to remove this variability by using data smoothing algorithms but variability could not be removed completely. Another disadvantage Gillard and Kramer mentioned that psycho-physiological measures are affected not only by workload but also by stress, strain and emotions. The opposite could also be true and therefore variations in GSR might also be caused by other factors than arousal like the workload and the strain of the task participants were performing. Yet another limitation could be the constant movement of hands while typing the keys or mouse movement. It is suggested to instruct participants not to move hands as it can produce noise in the recordings. However no such instruction were given to participants in this study, as doing so would obstruct their use of keyboard and mouse. Later smoothing algorithms were applied on the data to minimize the noise impact. Furthermore these potential noise factors were not systematically manipulated.

This experiment found a relationship between moods and keyboard\mouse usage patterns for some of the participants. Putting these patterns into neural networks and other learning algorithms might increase the percentage of affective state measurement and recognizing people that could show a possible significant relationship between keyboard mouse interaction and their moods. Zhai and Barreto (2006) considered such work. They extracted some features from Skin Responses (GSR), Pupil Diameter (PD), and Blood Pressure Volume (BPV) and put these features into three learning algorithms named as: Naïve Bayes Classifier, Decision Tree Classifier and Support Vector Machine (SVM). They found an accuracy of 79%, 88% and 90% respectively towards measurement of affective states. Future studies with neural networks can take a step further in not only classifying the states but also rate their affective states in a specific dimension of valence and arousal; something this study suggest might be possible for at least some people.

The findings presented in this chapter along with the findings of Chapter 4 support H_2 suggesting that it might be possible to measure the mood of some people from their interaction with keyboard and mouse. As, there is enough ground now that it might be possible to measure moods from interaction behaviour of users with keyboard and mouse, the next question inline for this research is whether an IDE generated intervention can help programmers to improve performance? This experiment was therefore planned and its design, procedure, material, analysis and results are explained in the next chapter.

Chapter 6: The Impact of a Computer-Support Intervention on the Programmers' Performance

6.1 Introduction

In previous chapters it was established that moods and particularly arousal have an impact on programmers' performance. It was also established that it might be possible to measure moods of computer users, such as programmers, from their interaction with keyboard and mouse. Assuming for a moment that a computer could measure different moods, would this information allow software to provide help or suggestions to programmers in order to improve their performance? This software component of course would have to be embedded in a suitable environment. In case of programmers such an environment could be an IDE (Integrated Development Environment). Before actually implementing such a component it is important to establish whether this component could really help to improve programmers' performance. This chapter will therefore present an experiment that examined whether a suggestion software system could help to increase a programmer's performance or not.

This chapter commences with a brief introduction to the underlying theoretical framework for this experiment. The next section of the chapter will discuss the experimental material used in the experiment. Participants' information will be provided in the next section. After that the experiment setup will be discussed which will be followed by the results. The results section is divided into two sub sections: a data preparation part and a data analysis part. The chapter ends with conclusions that can be drawn including the limitation of this experiment and suggestions for future research.

6.2 Background

An interactive work system has two main actors (Dowell and Long, 1989): the user and the computer, which together interact with the domain of application. To improve work performance, focus can be on the system (e.g. detecting when errors have been made), the user (e.g. detecting when person is in state where he/she is likely to make errors) or the interaction between the system and the user (e.g. adjust

the user interface to avoid errors). The focus in this experiment was on the user. Performance of the user, in this case a programmer, depends on various factors, such as cognitive capabilities as was discussed in Chapter 2. The CPTM model (Figure 2.2) represents a direct relation between programming tasks and cognition, whereas the CMM Model (Figure 2.3) represents a direct relation between cognition and moods. Together they present relation between moods and programming tasks as illustrated by the MPM (Figure 2.4). This implies that performance of the programmers might also be related to their moods as was demonstrated in the experiments presented in Chapter 3.

Moods can be considered as multidimensional with two prominent dimensions valence and arousal. Both of these dimensions are known to have an impact on performance. For example Yerkes-Dodson law (1908) indicate an inverted U shaped relation of arousal and performance. This law states that with an increase in arousal performance of the individuals' increases but only up to a point. A very high level of arousal also causes a decrease in the performance. Similarly positive moods or high valence are known to inhibit performance on analytic and detailed oriented strategies as well as increased creativity whereas negative moods are known to decrease creativity levels (Schwarz & Bless, 1991).

As indicated above, various studies show an impact of moods on performance with difference in terms of positive and negative moods and low or high aroused states. These differences are also visible in the behaviour of people as was discussed in detail in Chapter 4 and 5. People regularly try to self regulate their moods from negative to positive state. For example, people often get caught up in the moment following their moods and emotions. However interference in these moments from some one could cause a person to quickly attempt to pull out of these moments to get composed to interact with the other person (Eerber & Wegner, 1995). Social interaction therefore can be regarded as an intervention that causes people to regulate their mood. Interventions are often being used in research to achieve certain goals. For example Neighbors and Larimer (2004) evaluated a computer delivered intervention in reducing alcohol consumption in college students.

As Neighbors and Larimer (2004) used computer delivered interventions to evaluate and to reduce the drinking behaviour in college students, Watters et al. (1997) used caffeine to arouse subjects in an experiment to test the validity of Yerkes-Dodson law. Similarly, physical exercises are also known for their positive impact on

performance. For example McMorris and Graydon (1997) found that exercise could significantly increase the speed of visual search, speed of decision making and accuracy of decision making. Physical exercises are also known to have an impact on moods. For example Steptoe and Cox (1988) found that moderate physical exercises results into positive moods. Gowans et al. (2001) after an experiment involving physical exercises concluded that exercises had improved the moods and physical functions of their participants.

The experiment described in this chapter set out to analyze the performance of programmers before and after an intervention. But before looking at their performance, the next section will discuss the material used in the experiment.

6.3 Experimental Material and Design

The design of this experiment is based on the hypothesis that debugging or tracing performance of the participants will improve after an intervention suggested by the system. Therefore the design of this experiment considered analysis on the variables change before and after intervention. Dependent variable was performance and independent variable was time of measurement before and after intervention.

6.3.1 Algorithms

The basic aim of the experiment was to determine the impact of an intervention in the form of some physical exercises, on the programmers' program understanding and debugging performance. Therefore a total of 24 algorithms were selected and divided into three categories of easy, medium and difficult. Different levels of difficulty were selected to ensure that programmers with different programming skill could participate in the experience. The variation in difficulty would reduce potential floor or ceiling effects in programmers' performance. The algorithms were selected mainly from Parberry and Gasarch (2001) along with a basic algorithm book "Data structures and Algorithms in Java" from Lafore (1998). The algorithms taken from Parberry and Gasarch (2001) were already classified by the authors into these three categories. The algorithms taken from Lafore (1998) were categorized into three difficulty levels by the experimenter according to type of tracing, reasoning involved and data structures used in the algorithm. For example an algorithm with single or nested loop were categorized as 'easy' if it involve simple tracing and if no complex computations were involve. An example of such an

algorithm is given below. The algorithm contains only three nested loops. This means that only a relative simple tracing table is required.

Example Code 1 (Easy)

Consider the following algorithm

```
1.    for i := 1 to N
2.         for j := 1 to i
3.              for k := 1 to j
4                    Change line
5              End for k
6          End for j
7     End for i

Q.    Create a trace table where N = 10
```

Similarly, a medium algorithm was a mixture of loops with some basic reasoning and logic required in order to create the trace table. An example of such an algorithm is given below which has several loops. Further there are some mathematical computations that again increased the complexity of algorithm and tracing steps. In addition the participants have to reason and trace the output also.

Consider the following algorithm

```
1.    i := 0, j := 0, v := 1, N = 7
2.    for i:= 1 to N
3          for j := 1 to N – i
4               Output " "
5          End for loop j
6          for k := j to j + v
7               Output "*"
8          End for loop k
9          v := v + 2
10         Change Line
11    End for loop i
```

Q. *Create a trace table from the algorithm below.*

Output

```
   *
  ***
 *****
*******
```

****

A difficult algorithm was composed of algorithms with un-orthodox looping styles like recursion. These algorithms also contained some complex data structures, which might prove to be difficult to trace. An example of such an algorithm is given below.

Create a trace table of the array after each time this function call itself. Consider that the following array is passed in the function: 1, -1, 2, 3, 4, 5, 6, and 9.

```
1     function CallItself(array, arraySize)
2     if (arraySize > 1)
3          for i:= 1 to arraySize – 2
4               if array(i) > array(i+1)
5                    swap (array(i), array(i+1)
6               End if
7            End for loop
8            CallItself(array, arraySize – 1)
 9   End if
10   End Function
```

Note: *A complete list of algorithms can be viewed in Appendix II*

6.3.2 Intervention Exercise Video

The approach taken in the experiment was to decrease the arousal level over time by asking participants to trace various algorithms for a period of at least 16 minutes, hoping that overtime some element of boredom would set in. The intervention generated by the computer was in the form of a video in which participants were asked to participate in some simple warming up exercise. After the intervention the participants again continued with tracing algorithms for about another eight minutes. Analysing the performance before and after intervention provided an insight of the impact of the computer-based intervention on participants' performance.

A special exercise video was prepared with duration of 2 minutes and 17 seconds. The video contained some very simple exercises like warming up by moving hands, legs and jumping. Background music was also being played with the exercise instruction video.

6.4 Participants

Invitations to the participants were sent via email. Participants were also approached via personal contacts. There were a total of 23 participants. However, four

participants left in the middle of the experiment because of other commitments. The mean age of the remaining 19 participants was 28.1 years with a standard deviation of 4.5. There was only one female participant. Asked to categorise themselves, 79% of the participants categorized themselves as programmers, 16% categorized themselves as expert computer users and 5% categorized themselves as medium computer users. The mean computer experience of the participants was 8.3 years with a standard deviation of 2.9 years.

6.5 Experimental Setup

As mentioned previously, the expectation was that giving the participants algorithm tracing tasks for at least 16 minutes would decrease the arousal level of the participants. Participating in some physical exercise as suggested by the computer would again increase the arousal level and consequently their performance level. To implement this set up two applications were developed: 1) the main application based on the design of the experiment and 2) a training application that enabled participants to familiarise themselves with the experimental task. The main software program was comprised of an initial screen to input some of the participants' information. Consent was also required at this first screen, without consent it was not possible to continue with the experiment. The second screen contained some of the instructions to take this test as well as examples of how to dry run and trace the algorithms. There was also a link to a video showing a demo of the tracing format in which to dry run the algorithms. Finally algorithms were presented for about 16 minutes followed by a mood dialog box and an intervention containing instructions to do physical exercise as described in the video. The screen that displayed an algorithm contained three frames or text areas. The first frame was used to display the algorithm selected from a database with 24 algorithms. The second frame displayed information about what is required from the participant. The third frame was used to input the answer. All the answers were saved into a database after the participant pressed the 'Next button' or automatically in case the maximum time of 4 minutes to complete the task was passed and the participant was directed to the next algorithm. A sample screen of the training application can be seen in Figure 6.1.

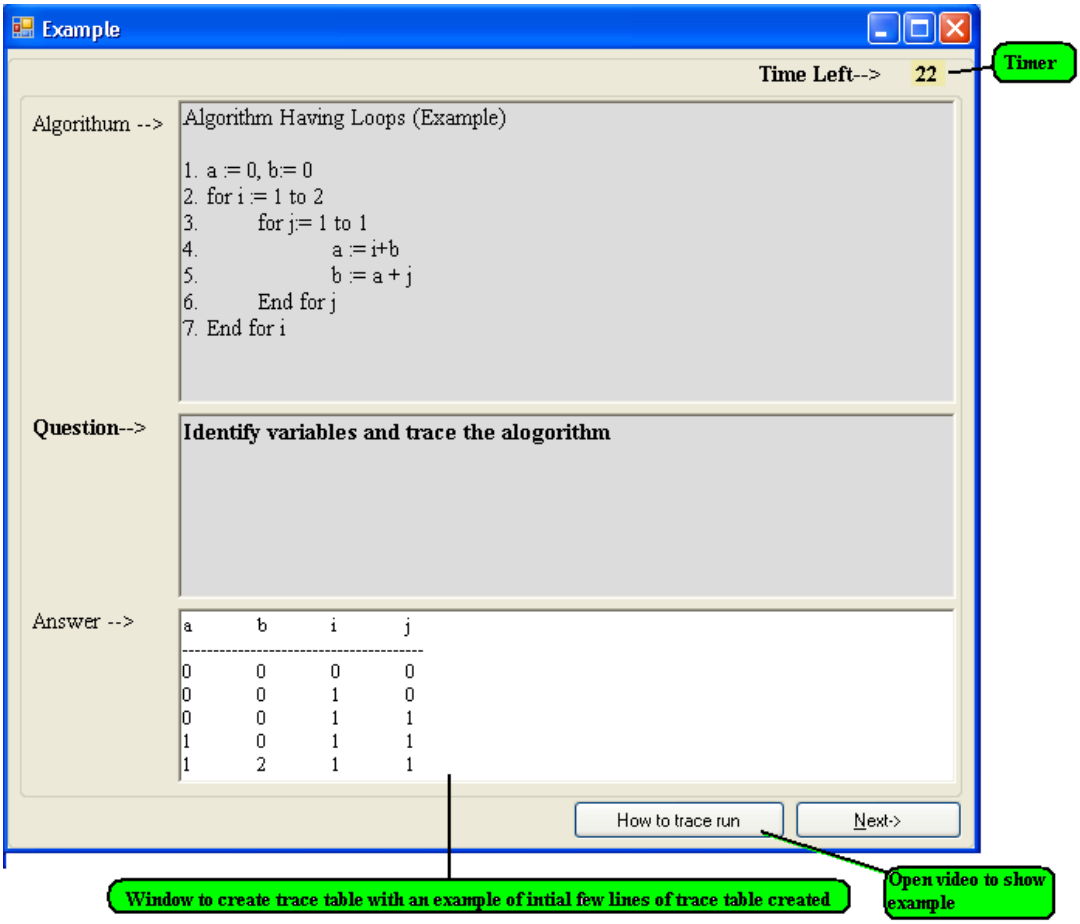


Figure 6.1: *A screen shot of the training application*

6.6 Procedure

The department ethics committee authorized the experiment. This experiment used a controlled setup, which included a separate room with no one going in or out during the experiment. Similarly all the other distracting devices like mobiles phones were switch off. Selected algorithms as explained in section 6.3.1 appeared on the screen in proper indentation and formatting so that it could be easy to read. Each question related to an algorithm required the participant to produce a trace table. Participants could write their answer in a separated text box in the application. Participants had four minutes to complete each algorithm. Participants were able to go to the next algorithm by clicking the 'Next button' if they wanted to or if they completed tracing the algorithm. If participants were not able to complete the algorithm within the required time any input in the answer text box was automatically saved in the database and participants were presented with the next algorithm. Algorithms kept on appearing, after 16 minutes, however, participants were not given a new algorithm, instead they were asked to rate their mood on the self-reporting two

dimensional SAM (self assessment Manikin) scale. The scale ranged for valence from 1 -being happy- to 9 -being sad. Similarly, arousal ranged from 1 -being excited- to 9 -being calm and sleepy.

After rating their mood, a video was displayed on the participant's screen. Participants were asked to repeat some simple physical exercises as shown in the video. The exercise lasted 2 minutes and 17 seconds and was followed by another mood rating dialog box. This was followed again by the next sequence of algorithms. Algorithms always appeared in a cycle of easy, medium and difficult before intervention. However, after intervention phase started with the same difficulty level algorithm that was displayed just before intervention. For example suppose that a participant answered an easy a medium and a difficult algorithm. After intervention the sequence s/he would get will be difficult, medium and easy. Similarly a participant getting a sequence of easy, medium, difficult, easy algorithms before intervention would get easy, difficult, medium, easy sequence. This arrangement ensured that performance before and after the intervention was balanced as the level of difficulty was spread equally. This was especially important for algorithm just before and after the intervention.

The procedure of the experiment is explained in the Figure 6.2 below.

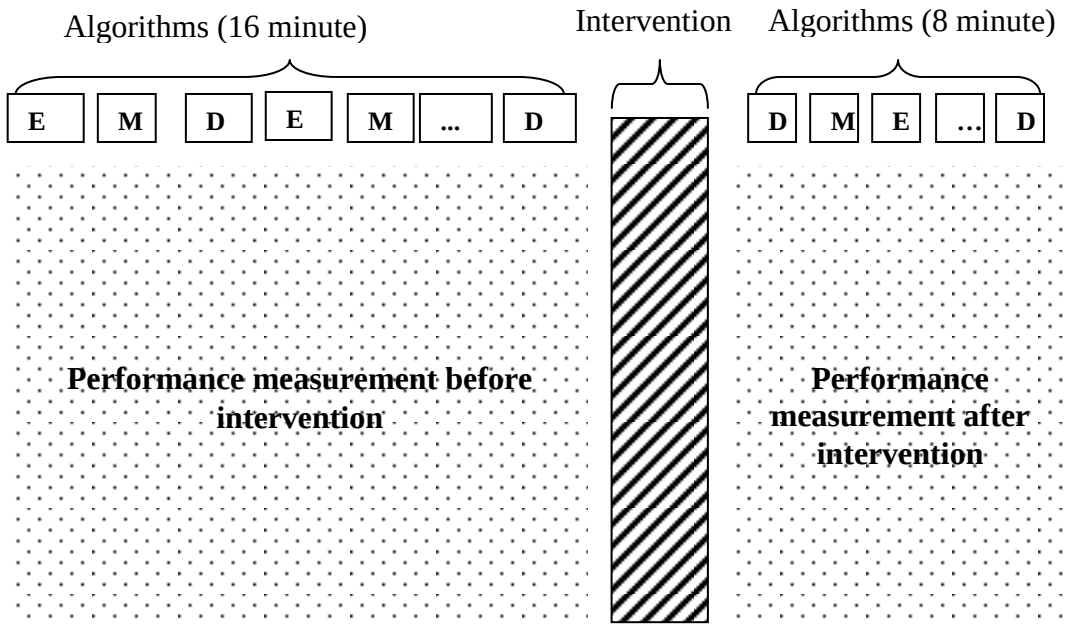


Figure 6.2: *Diagram showing pattern of the experiment*

6.7 Results

6.7.1 Data Preparation

The answers from the participants were saved in the database. For consistent marking of these answers two programmers who did not participate in the study were asked to mark the answers. To help the markers, a special application was developed with three windows on the main screen. The window on the top left side was used to display participants' answer whereas the top right window was used to display the correct trace of the algorithm. The bottom left window was used to display the algorithm along with the question. The bottom right side contained five text boxes. Three text boxes were 'Total variables identified', 'Time left to answer the algorithm' and 'Total lines of output'. The values for these boxes were automatically calculated by the application. The markers were only asked to fill the text boxes of 'Correct flow' and 'Correct output' for each algorithm. The maximum marks for the flow was 3 and the maximum marks for correct output was 4. Markers could mark in decimal points. Markers were instructed to mark on the bases of overall flow rather than exact output. For example if participants started an algorithm with wrong initial values than it was likely that they will have all the other output wrong. So instead of marking based on the initial values markers could use initial values to trace the rest of algorithm and could judge its accuracy. Therefore, markers were instructed to take special care in spotting correct flow. To help them in this, another utility was developed in which a marker was able to input the initiated values. This utility was able to provide possible correct flow and correct output resulting from initiated values. This utility was helpful for the marker to check whether participant maintained consistency in the tracing after initial error(s). The markings from the markers were saved in the database after they clicked the next button.

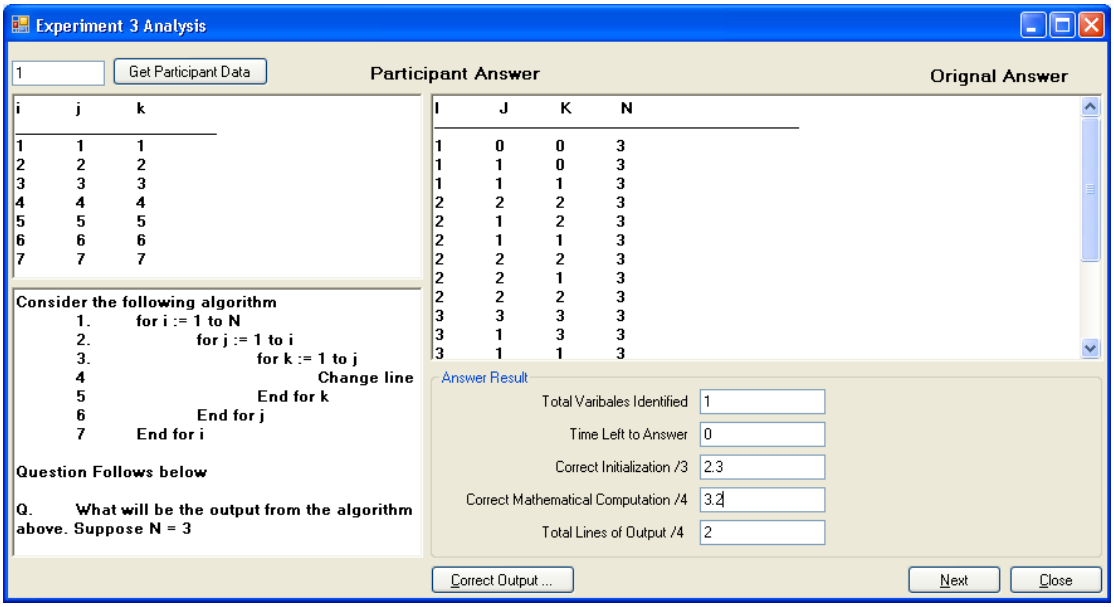


Figure 6.3: *Screen shot of the marking application showing three windows and place to input marks.*

The screen shots of the marking application and tracing utility are shown in the Figure 6.3 above and 6.4 below respectively.

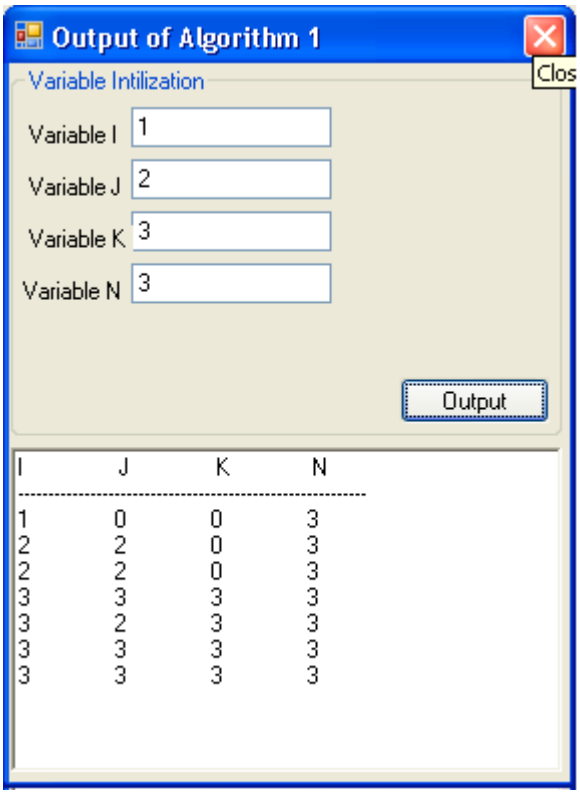


Figure 6.4: *Utility to provide possible correct flow & output from the initializing values provided by participant in the algorithm trace table.*

Two markers marked 147 algorithm traces from the 19 participants. The consistency between the two markers was examined with calculating Pearson

correlations between their marks across all the participants. Paired correlations were carried out between markings of marker 1 and marker 2. The marking outputs were comprised of all the markings of algorithms by all the participants as 'Correct flow' or as 'Correct Output'. The results revealed that there was significant correlations for the correct output (r (146) = 0.88, $p < 0.0001$) and for the 'Correct flow' (r (146) = 0.72, $p < 0.0001$). As correlations between markings of maker 1 and marker 2 were acceptable high, averages of the markings of the both markers were taken as final markings for these two measures. The other performance measures that were used in the analysis were 'Time left to complete tracing', 'total variables identified' and 'total lines of output'. These measures automatically calculated by application ensuring consistency in the marking.

6.7.2 Analysis

A Paired samples *t*-test conducted on arousal rating before intervention and after intervention indicated that that participants' arousal level significantly improved after doing exercises, $t(18) = 6.7$, $p < 0.0001$. Similar tests conducted on valence rating before intervention and after intervention indicate that valence level also significantly improved after doing exercises, $t(18) = 6.9$, $p < 0.0001$. The mean rating of arousal before intervention was 6.3 and after the intervention 4.4 on the 9-point Likert scale. A ratings near 1 indicated higher arousal whereas near 9 indicated lower arousal. Similarly mean of the valence rating before the intervention was 6 and after the intervention was 4.2, where scores near 1 meant happier whereas near 9 meant sadder. These findings suggest that the computer-based intervention had a significant impact on the participants' mood -the mediating factor that was expected to influence the participants' performance.

To examine the effect of the intervention on performance, a repeated multivariate analysis was conducted on the measures: 'total variables identified', 'total lines of output', 'correct flow', 'correct output' and 'time left' just before and after intervention. Results showed that participants improved significantly after intervention (F (5, 19) = 3.51, $p = 0.03$).

Follow up univariate analyses indicated that this significance could only be found in the correct output (F (1, 19) = 13.81, $p = 0.002$). Figure 6.5 shows the mean marks given to correct output of the algorithms just completed before (I) and after (I+1) the intervention. The marks are presented in z-values. The differences in marks

between easy, medium and difficult levels algorithms were relatively small with means of 2.32, 2.29, 2.34 and standard deviations 1, 0.75 and 1 respectively. Figure 6.5 shows z-values standardized for each difficult level. This removed any potential distortion in the figure of the performance over time as for example 74% of participants completed a medium level algorithm just before the intervention (I) and before that an easy level algorithm (I-1). The mark for the correct output was slightly below the average mark (–0.07) just before the intervention (I) whereas participants got the highest mark of 0.60 SD above the average marking for the first algorithm they complete just after the intervention (I+1). Looking at Figure 6.5 is seems that up to I-1 performance was still improving as part of learning effect, and after this point possible fatigue or boredom might have set in. The effect of the intervention also seems temporary as performance seems again to drop in the second (I+2) and the third (I+3) completed algorithm after the intervention.

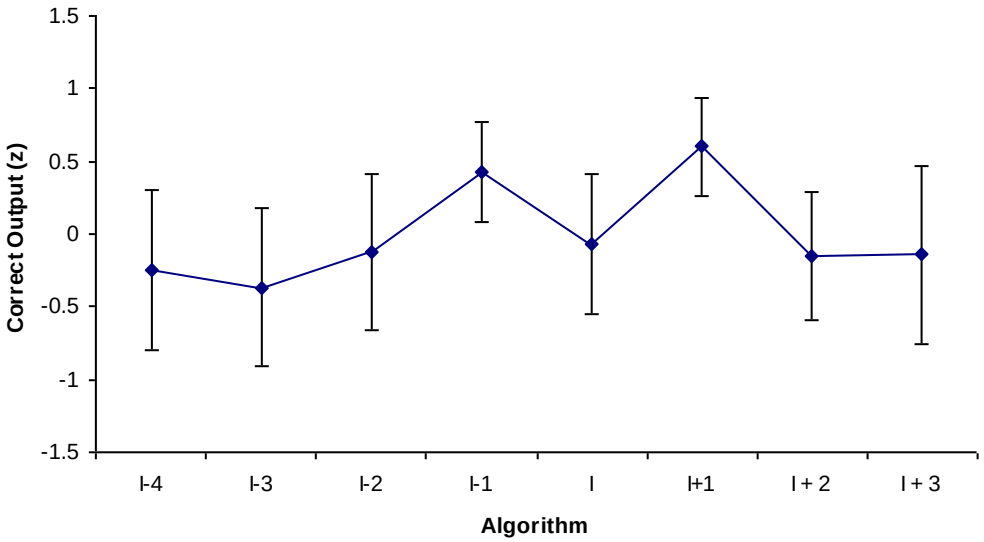


Figure 6.5: *Correct output performance standardized by difficulty level including 95% confidence interval, whereby 'I' stands for the algorithm complete just before the intervention and, (I+1) is the first algorithm completed just after the intervention. Note that algorithms (I-3) to (I+2) were complete by all 19 participants, algorithm (I-4) by only 17 and (I+3) by only 12 participants.*

6.8 Conclusions, Limitations and Future Research

The findings of the study suggest that an increase of arousal and valence after computer-based intervention in the form of physical exercise coincided with an

increase of the performance in the algorithm-tracing task. This is an effect that could not simply be explained by learning effect over time as a decline in performance seems to have set in just before the computer-generated intervention. These findings therefore seem to support H_3.

Reflecting on limitation of the study, the sample of participants had a gender bias with only a single female participant. This study had within-subject design and consequently learning effects might have been a confounding factor. A potential alternative interpretation is that participants performed better after the intervention as they were more familiar with the type of algorithm as they were kept constant before and after the intervention. Although in theory this might have caused some improvement, it still would not explain the entire improvement from a –0.07 SD to 0.60 SD of the mark when comparing it with the variation in performance of the other algorithms. To be more precise, 74% participants completed a medium level algorithm before and after the intervention, but also at point I-3 participants obtained a mean mark of 0.37 SD below the average mark for algorithms with medium difficulty level. Another limitation could be the time pressure as participants had to complete the algorithms within 4 minutes. However time pressure is not unrealistic for an industrial environment. Still this study was done in a controlled lab environment, stretching ecological validity. As a first step to study this effect, this setting was however considered as appropriate.

The limitations indicated above may lead to some possible future research based on this experiment. For example this study could be repeated with a between-subject design where one group receives an intervention and another group does not. Of course to have a similar level of statistical power such as study would need a far larger sample of participants. Another option would be to study the effect of computer generated intervention in the field. Although appropriate performance measures would have to be selected, the intervention could simple be time-based instead of mood-based, avoiding the need for measuring programmers' mood. The findings presented in this chapter would suggest that programmers' performance would benefit from such as setup, an idea well worth investigating.

Chapter 7: Conclusions, Limitations and Future Research

7.1 Overview

This chapter gives the conclusions, limitations and possible future researches ideas that emerged from the research discussed in this thesis. The chapter commences by recapping the motivation behind and conclusions drawn from the research. The research focused on three hypotheses, therefore after a general recapitulation each hypothesis will be discussed in detail. This then leads to a reflection of the overall hypothesis of this research that states that 'an integrated development environment could measure the programmer's moods in a non-interruptive and un-obstructive manner and generate effective interventions to alter the programmer's mood in order to improve his/her performance. This reflection has to be seen within the context of a possible implementation of such a system but also the limitations of the research as will be discussed in the next section. This and new insights also leads to suggestions for possible future research that could arise from this thesis. Both the theoretical and practical contributions of this work are discussed and some final remarks conclude this chapter.

7.2 Recapitulation

The thesis started by indicating that the quality of software depends on how well programmers perform their programming task. The position was taken that one of the factors affecting a programmer's performance was their mood. Consequently, it was suggested that an Integrated Development Environment (IDE) could help programmers to improve their performance by considering their current mood. With this motivation in mind Chapter 2 set out to examine the literature on programmers' performance and their mood. The literature study explored the relationship between cognitive functions and programming by putting forward the Cognitive Programming Model (CPTM) as well as a relation between cognitive functions and moods by putting forward the Cognitive Mood Model (CMM). Combining these two models led to the Mood Programming Model (MPM), which provided indirect support for the notion that moods have an impact on programming performance. Chapter 2 also indicated the lack of reports in the literature that examine a direct relation between

moods and programming in an empirical setting. The Chapter 2 also described various mood measurement methods that could be used to measure moods but noted also that they could be interruptive and obstructive. Therefore mood measurement from human interaction with keyboard and mouse was put forward as a method of mood measurement that could also be non obstructive and non interruptive. Chapter 2 pointed out the lack of literature on Integrated Development Environments that could improve programmers' performance considering their moods. However reports from other domains supported the idea that computer generated intervention could change people's mood.

Chapter 3 of this thesis reported an experiment that tested hypothesis H_1 that concerns the impact of moods on programmers' performance and found that arousal could significantly impact debug performance of programmers. However findings suggested there was no significant effect of valence on the programmer's performance. Chapter 4 and 5 tested hypothesis H_2 and found that for some computer users mood correlated with their keyboard and mouse behaviour, supporting the idea of a possible interaction based mood measurement instrument. It was also established that people who had a significant correlation between their interactive behaviour and their mood were partly identifiable from their personality traits. The last study presented in Chapter 6 confirmed the idea that a computer-generated intervention could change the programmer's mood and consequently improve their performance. Together this supports the tenet of an IDE that could measure moods and then provide suggestions accordingly. The next section looks at the hypotheses in more detailed and what conclusions can be drawn from the findings.

7.3 Conclusion from Hypotheses Testing

7.3.1 H_1: Moods have an Impact on Programmers' Performance

This study devised CPTM based on the relation between cognition and programming tasks (Figure 2.2) and CMM based on the relation between moods and cognition (Figure 2.3) thus forming MPM, which suggests a relationship between moods and programming tasks (Figure 2.4). Together this led toward testing H_1. The indirect support from the literature for MPM indicated a need for empirical evidence that specifically addresses a direct relation between moods and programming and therefore the two experiments were conducted toward this goal as explained below.

The first experiment was conducted to test H_1 and used a two-dimensional valence, arousal model for the self-reporting mood measurement. The experiment only focussed on one part of the programming task, which was debugging. The experiment was conducted online, where 72 participants took the test at their preferred time and place. Results of the experiment suggested that an increase in the arousal level coincided with a significant improvement of programmers' debugging performance. Valence showed a tendency to have an impact, although non significant on programmers' debugging performance. Decrease in valence, i.e. less positive, coincided with an improvement in debugging performance. As control of the environment is lacking in any online experiment a second experiment was conducted within a laboratory environment with only 20 participants. This experiment was not able to replicate the findings of the online experiment. The effect for valence and arousal on performance were ($p = 0.09$) & ($p = 0.07$) respectively. Still, the results of both experiments seem to support H_1.

The results for the effect of arousal on performance should also be viewed in relation to Yerkes-Dodson law. This law indicates that an increase in arousal causes an increase in performance but up to a certain level and after this an increase in arousal could adversely affect performance. This however was not studied in these experiments and therefore would require future research. As this study only focussed on one subtask –debugging– of the program task, several further studies are required to strengthen this idea stated in theory H_1. Interestingly the effect of valence on performance might be the complete opposite when more creativity is needed for example in program construction or modelling.

7.3.2 H_2: Moods can be measured From a Computer User's Interaction with Keyboard and Mouse

Hypothesis H_2 was supported by the findings of two experiments. The first experiment logged keyboard and mouse interaction data along with self-reported moods in normal working conditions. Data was collected from individuals over a period of 8 days on average. Whereas the first experiment was conducted in the field, the second experiment was conducted in the laboratory. Here keyboard and mouse events were logged while participants worked on a series of programming tasks. The participant's moods were induced by the use of high and low arousing music. Instead of using a self-report measure, mood was measured using a GSR (Galvanic Skin

Resistance) meter. The findings from both experiments support the hypothesis. In the first experiment about 30% of participants' interaction behaviour correlated significantly with their self reported moods. In the second experiment 50% of participants' interaction behaviour correlated significantly with their moods as represented by GSR readings. An important observation was also the lack of consistency between the participants on which behavioural indicators correlated with their mood. A universal behaviour based mood measure applicable to all types of computer users is therefore not supported by this data. This was even further complicated as the recorded behaviour of some participants did not reveal correlations with their mood. As the findings showed, the difference between these two groups of participants could be partly predicted by personality traits such as dutifulness and self-discipline.

7.3.3 H_3: A Development Environment with an Intervention or Suggestion System Could Help to Improve Programmer's Performance

A single experiment was conducted to test the hypothesis. Participants traced series of algorithms before and after a computer generated intervention. The intervention was an instruction video asking people to participate in some physical exercises. The results of the experiment revealed that after these intervention participants scored their arousal and valence significantly higher than before the intervention. Similarly the experiment also found that the described output of the algorithms after the intervention was significantly better than before the intervention. This lead to the conclusion that an intervention system implemented in a development environment might be able to help programmer to improve their performance. Considering the findings of Chapter 3, future research might study whether an intervention that would decrease valence, instead of increase it, would be more effective. It is Important to consider the programmers' willingness to have their mood changed in this direction. Future work could also consider designing the above experiment in a professional environment with professional programmers.

7.3.4 Overall Experimental Hypothesis

This thesis discussed hypotheses, literature and experiments toward a goal of developing an IDE for programmers that can measure a programmer's mood and then

can help them to improve their performance in this context. Initially it was demonstrated that moods and particularly arousal do have an impact on a programmer's performance. Laterally it was demonstrated that mood measurement with the help of un-interruptive and non-obtrusive method is possible. The method utilized was from the interaction behaviour of the computer users with keyboard and mouse and correlating with self-reported moods on SAM scale as well as from skin resistance measures. Although evidence in this study showed that it is possible to measure a computer user's moods from their use of keyboard and mouse, it does not support a universal behaviour-based mood measure. Possible future research might increase the chances to measure moods by considering other behaviour-based indicators. However, the logging and correlating mechanism tested in the experiments could be integrated in a development environment. The measured moods could allow IDE (software) to make decisions like what kind of suggestion or advice it should generate for programmers in order to improve their performance. The developers of such an IDE might also like to consider the following points:

1. Anger, sadness and excitement are within two-dimensional model of moods. These mood conditions along with others in the model cause different behaviours and therefore further experimentation is required for these different conditions and their effects on performance. Specific moods could be tested from two dimensions of moods as is illustrated by Russell (1980) in his circumplex model of affect.
2. One of the important aspects in this study is the impact of moods on the performance of programmers. Therefore an IDE should also know how to measure programmer performance. This thesis did not discuss computer-based measuring of performance in a non-specific task setting. Instead participants received specific debug tasks for which the desired outcomes were known. A substantial amount of literature exists on the measurement of performance of programmers. For example Takada et al. (1994) devise a method for measuring programmer performance from keystrokes. The idea has also been implemented in the software called "SourceKibitzer EyeQ" (Kofman, 2007). Performance measurement based on keystrokes seems aligned with the mood measurement from keyboard. Measuring both performance as well as mood might therefore be done by using the same software to collect this interaction data. An IDE therefore can compute correlations to measure mood as well as

performance, and thus can predict what type of mood results in high or low performance. This software thus could suggest ways to improve the performance personalised for that specific user. Neural networks as discussed in the conclusion section of Chapter 5 might prove to be very useful in this case, as they can learn which mood correlate with what kind of performance.

3. After knowing the mood, performance and the cause and effect of moods on performance software need to suggest some intervention or measure to improve programmer performance. However to do this software needs to know what kind of suggestion or intervention could regulate the mood. There is already some literature that shows what kind of interventions cause's specific moods. For example Steptoe and Cox (1988) showed that moderate exercise is related to higher valence (positive mood). While doing programming programmers often have to make decisions on choosing the efficient algorithms as well as methodology. Enhancing their performance might be possible as McMorris and Graydon (1997) showed that exercise significantly increase the speed of visual search as well as performance in searching, the speed of decision-making and the accuracy of decision. Similar research could help to formulate a list of interventions and their effect on moods and performance which software (IDE) can utilize to suggest to programmers.

4. In addition to measure moods from interaction behaviour with keyboard and mouse, another non-obstructive way of measuring moods is also possible with wireless psycho-physiological measures like GSR (Galvanic Skin Response). We could connect the GSR meter to a wireless device and in turn connect this to a band or ring to transmit skin resistance and this could form another approach to measure moods. Some work in this field has already being conducted although not published yet. For example Menon and Tashakkor (2005) proposed a design of a wristband that allow free movement anywhere in the room, thus making measurement from GSR simpler and non-interruptive.

7.4 Envisioned Mood Sensitive Component to Enhance Performance

Based on the suggestions in Section 7.3.3 a vision for a component in an IDE is presented in this section. Most of software has three basic parts: Input, Processing and Output. The vision is therefore described using these three basic parts along with additional storage.

a. Input

Input to the system will be key press events, mouse click and mouse movement events. As was discussed before mood measurement from interaction behaviour can not be generalized. Individuals can be identified based on the personality traits as was shown in the results of Chapter 4 & 5. Therefore another input required from the users is to rate their personality. Personality input could take two forms.

i. Presenting IPIP-NEO questionnaire to users in the form of dialog boxes to input traits manually

ii. Measuring personality from the keyboard and mouse interaction behaviour (Khan et al., 2008). In this case a separate processing unit could be implemented which could identify the personality of the user and then a decision could be taken to measure mood or not.

Before mood measuring from interaction data the component first needs to be trained. Therefore in the training phase the component will need mood information from another source. For example this could be from self-report mood measure like SAM or from psychophysical measures such as GSR. To enhance the range of the samples and to increase the learning phase, various mood-inducing background music tracks can be used to obtain samples from various moods.

b. Storage

To store the key and mouse patterns either a database system or XML files could be utilized. The patterns of storage can be of two types:

i. If each user has a single unique computer, than the interaction behaviour could easily be stored and can be related with that specific user. These stored patterns could also be input into neutral networks to increase the mood measurement rate.

ii. If multiple users are using a single computer then each user could be identified by their user name or from their personal identification numbers. Each

behaviour measure can be related to their ID and thus can be retrieved for the mood measurement purposes.

c. Processing

Processing involve various functions to be executed, from correlations to forming relations between users and their personality traits or between their moods and their interactive behaviour. Processing could also be divided into various parts like:

i. Forming patterns from the keyboard key press and mouse click events.

ii. Identification of patterns that could be related to different moods.

iii. Correlating performance of the programmers and their interaction behaviour with keyboard and mouse.

iv. Relating performance with moods based on the stored results of interaction behaviour.

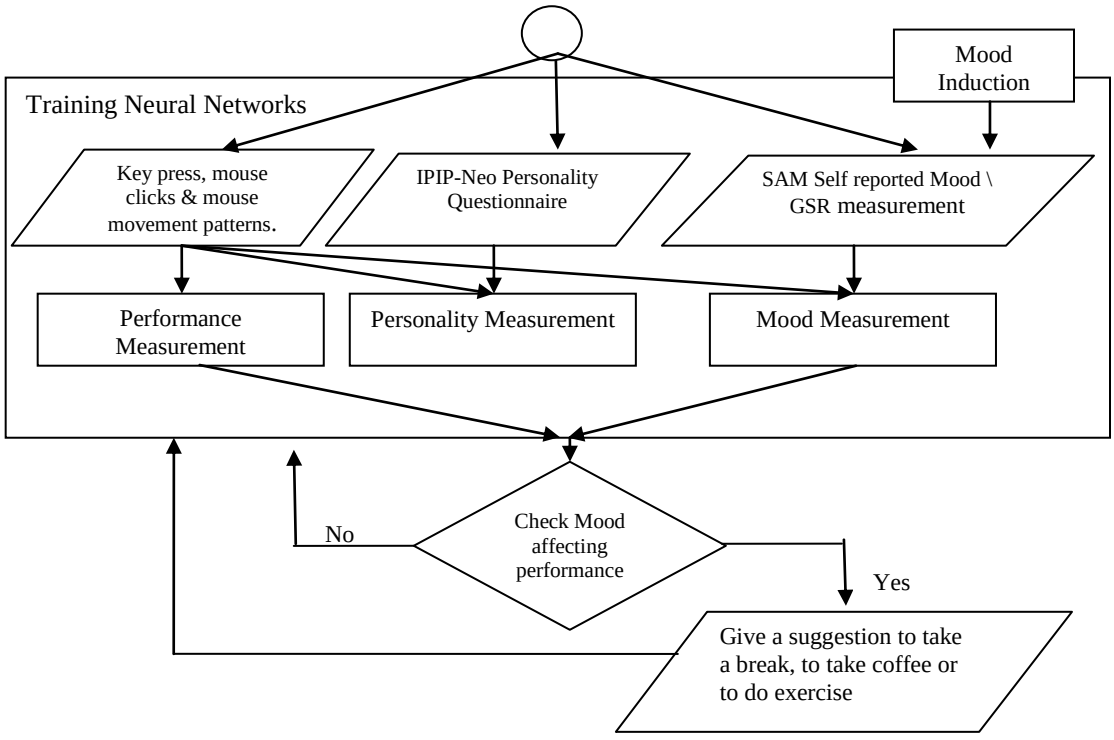


Figure 7.1: *Flow diagram of envisioned component*

d. Output

The required processing as discussed in the earlier section will result in a system that could measure moods as well as performance from interaction behaviour with keyboard and mouse, and thus can present some suggestions based on the performance predicted and mood measured. This suggestion could be presented in

various ways. However the suggested method could be displayed in the form of side screen popup messages, as they are not interruptive but still able to catch the user's attention.

7.5 IDE a Step Ahead

A design of the IDE as explained in the previous section aims to aid programmers based on its ability to recognise moods and suggest appropriate actions. Such an IDE could make potential contribution not only in improving performance of the programmers, but might also be used in regulating mood, cause by various factors such as limited time to deliver, development and performance pressures. This kind of IDE can help programmers in various ways like:

1. Will let them know about the mood they are in – People most often are not aware of their moods, therefore making them aware of their mood might allow them to take step to regulate it. Studies like Wingrove and Bond (1996) stated that mood and mood awareness are important factors in controlling impulsive behaviour.
2. Will help them to regulate their mood – On finding a mood effecting performance, the system can suggest suitable action or intervention to alter their mood.
3. Will let them know their personality – This kind of environment can also incorporate personality measuring and thus can let the users know about their personality. Again this will give individuals information they use for example selecting task in project teams. Further research could be conducted on the possibility of designing IDE that might also focus on measuring personality and then assigning tasks accordingly to members of a development team as mentioned by Dick et al, (2002), Lewis and Smith (2008).

7.6 Limitations of the Study

Limitations of each study have been discussed in their respective chapters. However some of the limitations concern the approach taken in this thesis. For example SAM (Self Assessment Manikin) devised by Lang (1980) was used in this research for self-reporting of the moods. Along with various advantages of this multi-dimensional approach some of the disadvantages include representation of moods and emotions in the form of two or three dimensions is an abstract expression with which not everyone is comfortable (Herbon et al., 2005). Another limitation visible in all the

studies is the very low number of female participants, making the findings biased towards a male population.

In this study various people from different backgrounds participated. In almost every experiment professional programmers were required. However getting access to this group and getting them to participant was found to be relatively difficult. Various strategies were applied to obtain an adequate level of involvement from professional programmers and conducting the experiment over the internet was one of the strategies applied. Still in the other experiments the sample consisted not exclusively of professional programmers, but included a mixture of PhD students, professionals and undergraduate programmers.

Although research mainly focused on debugging as a subtask of programming, the debugging tasks used might not be the representatives for entire debugging process where most of the efforts are used to locate and identify relevant code while ensuring that any changes do not create any ripple effect.

7.7 Future Work

Chapter 3 described two experiments conducted in order to find the impact of moods on programmers' performance and these focussed only on the debug task. However, the MPM established from the literature indicated that other programming subtask might also be affected by the mood of the programmers. Future research might test this empirically, which would help to assess the validity of MPM. Instead of just relying on the self reported moods, other mood measures should also be considered and practically applied in parallel.

Chapter 4 and 5 explained two experiments conducted to measure moods from the interaction of users with keyboard and mouse. Future research could continue on measuring both valence and mood from the interaction behaviour. However, in addition research might also aim at identifying specific moods such as anger and fear within this two-dimensional mood model.

Chapter 6 is about the use of interventions and its impact on programmer performance. The experimental results demonstrate that a suggestion or intervention could help programmers to improve their performance. Future work might like to replicate this finding outside the laboratory, and also look at other possible interventions and their effect on mood and performance.

In addition research on the presentation of the interventions, their usability, their social acceptability, and side effects also need to be studied in details in order to implement a development environment that could really help programmers to improve their performance in the context of their moods.

Three methods were used for mood manipulation in this research: 1) mood manipulation by using movie clips; 2) mood manipulation using music; and 3) mood manipulation using exercise. There are various mood manipulation or mood induction procedures apart from the ones used in this research and they are generally referred to as Mood Induction Procedures or MIP's (Westermann et. al, 1996). Westermann et. al. (1996) evaluated 11 important mood induction procedures by meta analytical procedures. They found film or story as the most important medium to induce both positive and negative moods. The other MIP's used were: 1) mood induction using film or story along with explicit instructions to enter into the specified mood; 2) imagination; 3) velten mood induction procedure; 4) social interaction; and 5) feedback MIP's. Future research could consider all of these methods separately or in combination to induce moods in order to analyze their effectiveness on the experiments described in this thesis.

Similarly various mood measurement methods could be used in parallel with mood measurement from the use of keyboard and mouse such as self-reporting, physiological, and behavioural measures (Table 5.5). This would help in validating the accuracy of mood measurement using other methods for example self reporting moods and mood measured by the use of keyboard and mouse. Future research could look into this possibility also.

All the experiments explained above were conducted online, with mostly professional participants, in the field, or were conducted in the lab on a mixed group of participants with a professional background or students. Future research can consider conducting all of these experiments in industrial environment with professional programmers. This requirement is important as IDE is targeted toward professional programmers in an industrial environment. Some of the experiment explained in chapters 3 & 4 might not require a lot of time. However experiments explained in chapters 5 & 6 require special preparations like compensating time and pay for the programmers in this scenario.

The outcome of this research is a proposed IDE to measure programmers' mood and then help them to improve their performance in context of their moods.

Future research can also consider developing such an environment and then testing it in the field and lab. This research could be conducted either by qualitative approach or quantitative approach. Qualitative approach (Questionnaires, interviews etc) could be used to understand further the outcome and perception of programmers about the tool. Quantitative approach could involve experiments on participants in different conditions at work environment. The next section will discuss theoretical and practical contributions made by this work.

7.8 Contributions

7.8.1 Theoretical Contributions

What will HCI be like in year 2020? This was an interesting question and a topic of a report that emerged from a meeting organized by Microsoft research (Harper et al., 2007). This meeting also discussed moods and emotions as important factors in effective human computer interaction and thus it being important to embed that in future computers. The work presented in this thesis seems clearly relevant in this context with an aim of developing an effective interactive development environment to help programmers to improve their performance. The work presented has investigated three experimental hypotheses toward development of a system. The investigation of these three hypotheses resulted in the following theoretical contributions:

The first contribution made was the development of three models that explain the relationship between mood and programming performance. These models were based on the literature findings and can be used as a framework for further empirical investigation. The work presented was empirically validated for one possible link between mood and programming which was for debugging and valence and arousal. Although there is a substantial amount of work reported on the impact of moods on performance, no reports in the literature were found up to now that specifically focussed on the impact of moods on IT professionals (programmers) supported by systematic empirical observations.

The second hypothesis studied was that moods can be measured from the interaction behaviour of computer users with keyboard and mouse. Although there is literature on the measurement of moods from the use of software, discussing techniques such as psycho physiological devices, none of these reports yet present

empirical results to validate these ideas. The results from the experimentation revealed that for some computer users it is possible to measure their moods from their interaction with keyboard and mouse. Besides the contribution made by this new insight, the work will also benefit other researchers interested in methods that are relatively inexpensive and non-interruptive.

The last contribution comes from the gained insight that physical exercise can improve programming performance. Where traditional program support focuses on the usability of the development environment or the detection of errors in the program code, this approach directly focuses on the human to improve performance.

7.8.2 Practical Contributions

The research presented in this thesis contributed toward the development of an affect recognizing and suggestion system for programmers. Although the user group was programmers the suggested support system might also benefit other computer users performing task with similar underlying cognitive functions. A blueprint presented for a mood sensitive component in Section 7.3.4 has been design to be independent from the programming environment, implying that it is possible to apply it with other types of software as well.

The finding that physical exercises can improve dry running of algorithms can help programmers with improving this task. This especially the case when they are performing this task for some time and boredom sets in, programmers should consider interrupting the task with some mild physical exercises.

7.9 Final Remarks

Affective computing is one of the rapidly growing areas of HCI research that provides means for affective interactions between computers and humans, thus making relationship more personal and helpful. In addition an emotional interaction may also produce a sense of responsibility toward maximum performance and caring approach like other humans have to other humans. Emotions help toward recognizing a win or loss, and thus people strive to win, which in turn make them happy (Damasio, 1995). The creation of affect recognizing development environment therefore is work toward this goal of increasing performance by knowing the moods of a person.

The studies in this thesis stated that moods (arousal, valence) do have an impact on programmer performance. In this context an Integrated Development Environment has been proposed which could measure the mood of programmers and computer users and provide them with help or suggestions to improve their performance. Improving programming performance and thus the quality of software could have far reaching effects as almost every human being is in some way affected by information technology in its daily life.

References

Albinoni, T.G., (1981). Adagio in G Minor for organ and strings [Recorded by SolistiVeniti, Conducted by Scimone, C.] On Albinoni's Adagios [CD]. Perivale, England Warner Classics, (1996)

Aleven, V., Sewall, J., McLaren, B. M., and Koedinger, K. R. (2006). Rapid Authoring of Intelligent Tutors for Real-World and Experimental Use, In Kinshuk, R. Koper, P. Kommers, P. Kirschner, D. G. Sampson, and W. Didderen (Eds.), *Proceedings of the 6th IEEE International Conference on Advanced Learning Technologies* (ICALT 2006), (pp. 847-851). Los Alamitos, CA: IEEE Computer Society

Almagor, M. and Ben-Porath, Y.S. (1989). The Two-Factor Model of Self Reported Mood: A cross cultural replication, *Journal of Personality and Assessment*, vol. 53, no. 1, pp. 10-21.

Ambady, N. and Gray, H.M. (2002). On Being Sad and Mistaken: Mood Effects on the Accuracy of Thin-Slice Judgments, *Journal of Personality and Social Psychology*, Vol. 83, No. 4, p. 947-961

Armitage, C. J., Conner, M. and Norman, P. (1999). Differential effects of mood on information processing: evidence from the theories of reasoned action and planned behavior, *European Journal of Social Psychology*, vol. 29, pp. 419-433.

Backs, R.W. and Boucsein, W. (2000). Engineering Psychophysiology as a Disipline: Historical and Theoretical Aspects in Engineering Psychophysiology, eds. *R.W. Backs and W. Boucsein, 1st edn, Lawrence Erlbaum Publications*, London, pp. 3-30.

Barr, A. and Tessler, S. (1996). Good Programmers Are Hard To Find: An Alternative Pespective on the Immigration Of Engineers, *Press briefing*, October 1996.

Bartolic, E. I., Basso, M. R., Schefft, B. K., Glauser, T. and Titanic-Schefft, M. (1999). Effects of experimentally-induced emotional states on frontal lobe cognitive task performance, *Neuropsychologia*, Vol. 37, pp. 677-683.

Baumgartner, R., Ryner, L., Somorjai, R. and Summers, R. (2000). Exploratory Data Analysis Reveals Spatio-temporal Structure of Null fMRI Data, Proc. Intl. Sot. Mag. Reson. Med. 8.

Blagrove, M. and Akehurst, L. (2001). Personality and the modulation of effects of sleep loss on mood and cognition, *Personality and Individual Differences*, vol. 30, no. 5, pp. 819-828.

Bloom, B. S. (1956). Taxonomy of Educational Objectives, the classification of educational goals, *Handbook I: Cognitive Domain*, New York: McKay

Bohner, G., Crow, K. and Hans-Peter, E. (1992). Affect and persuasion: Mood effects on the processing of message content and context cues and on subsequent behaviour, *Journal of Social Psychology*, vol. 22, no. 6, pp. 511-530.

Boucsein, W., and Thum,M. (1997). Design of work\rest schedules for computer work based on psychophysiological recovery measure, *International Journal of Industrial Ergonomics*, Vol. 20, pp. 51-57.

Bradley, M.M. and Lang, P.J. (1994). Measuring emotion: The self assessment Manikin and the seman-tic differential. *Journal of Behaviour Therapy and Experimental Psychiatry*, Vol. 25, pp. 49-59.

Brooks, F.P. (1975). The Mythical Man Month, Addison-Wesley, Reading, MA,

Brug, J., Steenhuis, I., Assema, P.V.and De Vries, H. (1996). The Impact of Computer-Tailored Nutrition Intervention, *Preventive Medicine*, Vol. 25, No. 3, pp. 236-242.

Buchanan., T., Johnson., J. A., and Goldberg., L. R. (2005). Implementing a five-factor personality inventory for use on the internet, *European Journal of Psychological Assessment*, vol. 21, pp. 115-127.

Card, K.C., Moran, T.P. and Newell, A. (1983). The Psychology of Human-Computer Interaction, *Lawrence Erlbaum Publishers*, Hillsdale, New Jersey.

Chang, A., and Wilson, M. (2004). Recalling emotional experiences affects performance on reasoning problems, *Evolution and Human Behavior*, vol. 25, pp. 267-276.

Cheyne J. A, Carriere J.S.A and Smilek D. (2006).Absent-mindedness: Lapses of conscious awareness and everyday cognitive failures, *Consciousness and Cognition*, Vol. 15, pp.578-592

Coetzer, G. H., and Richmond, L. (2007). An empirical analysis of the relationship between adult attention deficit and efficacy for working in teams. *Team Performance Management*, Vol. 13, Issue. ½, pp. 5-20.

Cohen, J. (1992 a). A Power Premier, *Psychological Bulletin*, Vol. 112, No. 1, 1992, pp.155-159

Cohen, J. (1992 b). Statistical Power Analysis, *Journal of Clinical Psychiatry*, Vol. 1, No. 3, pp. 61-69.

Cook, T. D., & Campbell, D. T. (1979). Quasi-experimentation: Design and analysis issues. Boston, MA: Houghton Mifflin Company.

Coutaz, J., Nigay, L., Salbe, D., Blandford, A., May, J. and Young, R. M. (1995). Four Easy Pieces for Assessing the Usability of Multimodal Interaction: The CARE Properties, *Proceedings of the Fifth IFIP International Conference on Human Computer Interaction INTERACT'95* (Lillehammer, Norway), 115-120. New York: Chapman and Hall

Cowie, R., Douglas-Cowie, E., Tsapatsoulis, N., Votsis, G., Kollias, S., Fellenz, W., and Taylor, G. (2001). Emotion recognition in human-computer interaction, *Signal Processing Magazine* IEEE, Vol. 18, Issue. 1, pp. 32-80.

Damasio, A. R. (1995). Emotion, Reason and the Human Brain, *First Ed. New York: Quill, An imprint of Harper Collins Publishers.*

Danaher J.W. (1980). Human error in ATC system operations, *Human Factors*, Vol. 22, pp. 535-545.

Darwin, C. [1872] (1965). The Expression of Emotion in Man and Animals, *University of Chicago Press*. Chicago

Davidson, R. J., Chapman, J. P., Chapman, L. J., and Henriques, J. P. (1990), Asymmetrical brain electrical activity discriminates between psychometrically matched verbal and spatial cognitive tasks, *Psychophysiology*, Vol. 27, pp. 528-542.

Deitel, H.A., Deitel, P.J., and Neito, T. (2000). Instructor Manual for C++ How to program. 3rd Edition, *Deitel and Deitel Associates Inc*.

Delplangue, L., Silver, L., Hot, P., Rigoulot, S. and Sequeira, H. (2006). Arousal and valence effects on Event-Related P3A and P3B during Emotional Categorization, International, *Journal of Psychophysiology*, Vol. 60, No. 3, pp. 315-322.

Derryberry, D. and Rothbart, K. (1984). Emotion, attention and temperament. In *Emotions cognition and Behavior, ed. Izard, E.C., Kagan, J. and Zajonc, B.R.*, 1st Edition, Cambridge University Press.

Dick, A. J., Zarnett, B. (2002). Paired Programming and Personality Traits, *Proceedings of the Third International Conference on Extreme Programming and Agile Processes in Software Engineering*, pp. 82-85.

Dienstbier, R. A. (1984). The role of emotion in moral socialization, in *Emotion, Cognition and Behav-ior, 1st Ed. Izard, C. E., Kagan, J. and Zajonc, R. B.:* Eds. USA: Cambridge University Press, 1984, pp. 484-514.

Dijkstra, A. and De Vries, H. (1999). The development of computer-generated tailored interventions, *Patient Education and Counselling*, Vol. 36, No. 2, pp. 193-203.

Dow, J. (1992), External and Internal Approaches to Emotion, *Psycoloquy*, vol. 3, no. 1.

Dowell, J. and Long, J. (1989). Towards a conception for an engineering discipline of human factors, *Ergonomics*, vol. 32, No. 11, pp. 1513-1535.

Ekman, P. (2003), Changing What We Become Emotional About in Emotions Revealed, *ed. P. Ekman, First edn, Times Book (Henry holt and company) LLC*, New York, pp. 67-68.

Erber, R. and Erber, M.W. (2001), 'Mood and Processing: A view from self-regulation Perspective' in *eds. L.L. Martin and G.L. Clore, 1st edn, NJ: Lawrence Erlbaum*, Mahwah, pp. 63-84.

Erber, R. and Wegner, D. M. (1995). On Being Cool and Collected: Mood Regulation in Anticipation of Social Interaction, *Journal of Personality and Social Psychology*, Vol. 70, No. 4, pp. 757-766.

Fahrenberg, J. and Wientjes, C.J.E. (2000), Recording Methods in Applied Environments in Engineering Psychophysiology, *eds. R.W. Backs and W. Boucsein, 1st edn, Lawrence Erlbaum Publications*, Mahwah, New Jersey, pp. 111-136.

Fiedler, K. (2001). Affective influences on social information processing, In *J.P Forgas (Ed.), Handook of Affect and Social Cognition*, pp. 163-185. New Jersey, Lawrence Earlbaum Associtates.

Fisher, G. (1987). Cognitive View of Reuse and Redesign, *IEEE Software, Special Issue on Reusability*, vol. 4, No. 4, pp. 60-72.

Fitzgerald, S., Simon, B. and Thomas, L. (2005). Strategies that students use to trace code: An analysis based in Grounded Theory, *First International Computing Education Research Workshop ICER'05*, October 1-2, 2005, Seattle, Washington, USA

Forgas, J. P., and Fiedler, K. (1996). Us and Them: Mood Effects on Intergroup Discrimination, *Journal of Personality and Social Psychology*, Vol. 70, pp. 28–40.

Fowles, D. C. (1980). The three-arousal model: implications of Gray's two-factor learning theory for heart rate, electrodermal activity, and psychopathy. *Psychophysiology*, 17, 87±104.

Fridja, N. (1993). Moods, emotios episodes and emotions. in *Handbook of emotions, eds. M. Lewis and I. Haviland, Guildford*: New York, pp. 381-403.

Gaillard, A. W. K., and Kramer, A. F. (2001). Theoretical and methodological issues in psychophysiological research. In *R. W. Backs and W. Boucsein (Eds.), Engineering psychophysiology* (pp. 31–58).Mahwah, NJ: Erlbaum.

Gao, C., Lu, D., and She, Q. (1990). The effects of VDT data entry work on operators, *Ergonomics*, Vol. 33, pp. 917-924.

Gardner, M. P. and Hill, R. P (1990). Consumers' mood states and the decision-making process, *Marketing Letters*, vol. 1, pp. 229-238.

Gendolla, G. H. E. (2000), On the Impact of Mood on Behavior: An Integrative Theory and a Review, *Review of General Psychology*, Vol. 4, No. 2, pp. 378-408.

Gowans, W. E., Dehueck, A., Voss, S., Silaj, A., Abbey, S.E. and Reynolds, W. J. (2001). Effect of randomized, controlled trial of exercise on mood and physical functions in individual with fibromyalgia, *Arthritis Care and Research*, vol. 45, No. 6, pp. 519-529.

Gray, J. A. (1975). Elements of a two-process theory of learning. New York: *Academic Press*.

Green, T. R. G. and Petre, M., (1996), Usability analysis of visual programming environments: A cognitive dimension Framework, *Journal of Visual Languages and Computing*, Vol. 7, No. 2, pp. 131-174

Gueheneuc, Y-G. (2005), a Theory of Program Comprehension, In. Joining Vision Science and Program Comprehension, *Technical report 1267*, University of Montreal.

Gundel, A., Drescher, J., MaaB, H., Samel, A., and Vejvdova, M. (1995). Sleepiness of civil air-line pilots during consecutive flights of extended duration. *Biological Psychology*, Vol. 40, pp. 131-141.

Haider, E., Luczak, H., and Rohmert, W. (1982). Ergonomics investigations of workplaces in a police command-control centre equipped with TV displays, *Applied Ergonomics*, Vol. 13,pp. 163-170.

Haigh, S. and Megarity, J. (2000). Studying E-Journal User Behaviour Using Log Files. The Experience of Super Journal, *Library and Information Science Research*, Vol. 22, No. 3, pp. 311-338.

Harper, R., Rodden, T., Rogers, Y. and Sellen, A. (2007). Being Human: Human Computer Interaction in 2020, 'HCI 2020', *Microsoft Research meeting* El Bulli Hacienda Hotel, Sveille, Spain.

Herbon, A., Peter, C., Markert, L., Meer, E.V.D. and Voskamp, J. (2005). Emotion studies in HCI - a new approach, *Proceedings of International Conference on Human-Computer Interaction*, Las Vegas.

Ingleton, C. (1999). Emotion in learning: a neglected dynamic, *HERDSA Annual International Conference*, Melbourne, pp. 12-15.

Irons, D. M. (1982) Cognitive correlates of programming tasks in novice programmers, *Proceedings of the conference on Human factors in computing systems*, pp.219-222

Isen, A. M. and Nehemia, G. (1987) The Influence of Positive Affect on Acceptable Level of Risk: The Person with a Large Canoe Has a Large Worry, *Organizational Behaviour and Human Decision Processes*, Vol. 39, pp. 145–54.

Izard, C. E. (2007), Basic Emotions, Natural Kinds, Emotion Schemas, and a New Paradigm, *Perspectives on psychological science*, Vol. 2, Issue. 3, pp. 260-280.

Izard, E.C., Kagan, J. and Zajonc, B.R. (1984). Introduction in: Emotions cognition and Behavior, *ed. Izard, E.C.,* 1st Edition, Cambridge University Press.

Jamison, K. R. (2003). Foreword, in Mood Disorders: A Handbook of Science and Practice (1st ed), Eds. M. Power.

Jamison, K.R. (2004), Foreward in Mood Disorders: A Handbook of Science and Practice, *ed. M. Power, 1st edn, John Wiley and Sons Limited*, West Sussex, England.

Jarvis, M. J. (1992), Does Caffeine intake enhance absolute levels of cognitive performance, *Psychopharmacology*, Vol. 110, No. 1-2, pp. 45-52.

Jeanick, B., Maya, C. and Arcelin, R. (2002). Effects of Acute Physical Exercise Characteristics on Cognitive Performance, *Sports Medicine*, Vol. 32, No. 9, pp. 555-566.

Johnson, N., Galata, A. and Hogg, D. (1998). The acquisition and use of interaction behavior models, *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 866–871.

Jones, C.B. (2003). The early search for tractable ways of reasoning about programs, *Annals of the History of Computing*, IEEE, vol. 25, no. 2, pp. 26-49.

Kaufmann, G. and Vosburg, S.K. (1997). Paradoxial Mood Effects on Creative Problem Solving, *Cognition and Emotion*, vol. 11, no. 2, pp. 151-170.

Kaufmann, G. (2003). Expanding the mood-Creativity Equation, *Creativity Research Journal*, vol. 15, No. 2 and 3, pp. 131-135.

Kennedy, A. and Raistrick, C. (2000), An integrated environment for simulation and modeling, *IEE Seminar on tools for Simulation and Modelling IEEE*, pp. 21.

Khan, I. A., Brinkman, W. P., Fine, N. and Hierons, R. M. (2008) Measuring Personality from Keyboard and Mouse Use, In: *European Conference in Cognitive Ergonomics*, Madeira, Portugal

Khan, I.A., Brinkman, W.P., and Hierons, R.M. (2007) Moods and Programmers Performance, *19th Psychology of programming workshop*, Joensuu, Finland, P.3-16

Kim, D. and Cho, I. (1997), FADIS: an integrated development environment for automatic design and implementation of FLC, *Annual Meeting of the North American Fuzzy Information Processing Society*, (1997). NAFIPS '97.IEEE, , pp. 33-39.

Kirchsteiger, G., Rigotti, L. and Rustichini, A. (2006). Your morals might be your moods, *Journal of Economic Behaviour and Organization*, vol. 59, pp. 155-172.

Klein, J. (1999) Computer Response to User Frustration, Thesis, *MIT Media Lab Technical Report* , No 480

Kleinginna, P.R. Jr. and Kleinginna, A.M. (1981). A categorized list of emotion definitions, with suggestions for a consensual definition, *Motivation and Emotion*, Vol. 5, No. 4.

Ko, A. J. and Myers, B. A., (2003). Development and Evaluation of a Model of Programming Error", *IEEE Symposia on Human-Centric Computing Languages and Environments*, Auckland, NewZeland, pp. 7-14.

Koedinger, K.R., Aleven,V., Heffernan, N., McLaren, B. and Hockenbery, M. (2004). Opening the Door to Non-Programmers: Authoring Intelligent Tutor Behaviour by Demonstration, in: *Lesture Notes in Computer Science*, Vol. 3220, pp. 162-174.

Kofman, N. (2007). SourceKibitzer EyeQ released. Available: http://www.theserverside.com/news/thread.tss?thread_id=47797. Last accessed 15 December 2008.

Krantz, J.H. and Dalal, R. (2000). Validity of Web-Based Psychological Research, In: *Birnbaum, M.H. (eds.): Psychological Experiments on the Internet*. Academic Press, USA pp.35-60.

Kukreja, U., Stevenson, W. E., and Ritter, F. E. (2005) RUI—Recording User Input from interfaces under Windows, *Behavior Research Methods*, vol. 38, No. 4, pp. 656-659.

Lafore, R. 1998. "Data structures and Algorithms in Java", Published *By 'Waite Group Press*".

Lane, A.M., Whyte, G.P., Terry, P.C. and Nevill, A.M. (2005), Mood, Self-set Goals and Examination Performance: The Moderating Effect of Depressed Mood, *Personality and individual differences*, vol. 39, no. 1, pp. 143-153.

Lang, P. J. (1995). The Emotion Probe: Studies of Motivation and Attention. *American Psychologist*. 50(5), 372-385.

Lang, P.J. (1980). Behavioral treatment and bio-behavioral assessment: computer applications. In: *Sidowski, J. B., Johnson, J. H. and Williams, T. A. (Eds). Technology in mental health care delivery systems*. Norwood, NJ: Ablex.

Larson, G. E., Alderton, D. L., Neideffer, M., and Underhill, E. (1997). Further evidence on dimensionality and correlates of the Cognitive Failures Questionnaire, *British Journal of Psychology*, 88, 29-38.

Lewis, P.A. and Critchley, H.D. (2003). Mood dependent memory, *Trends in Cognitive Sciences*, vol. 7, no. 10, pp. 431-433.

Lewis, T. L., and Smith, W. J. (2008). Creating High Performance Engineering Teams: the Impact of Problem Solving Style Dominance on Group Conflict and Performance, *Journal of Computing Science in Colleges*, Vol. 24, No. 2, pp. 121-129.

Lissetti, C.L. and Nasoz, F. (2002). MAUI: a Multimodal affective user interface, *Proceedings of the tenth ACM international conference on Multimedia*, Juan-les-Pins, France, pp. 161-170.

Lister, R., Adams, E., Fitzgerald, S., Fone, W., Hamer, J., Lindholm, M., McCartney, R., Moström, J. E., Sanders, K., Seppala, O., Simon, B. and Thomas, L. (2004). A Multi-National Study of Reading and Tracing Skills in Novice Programmers, *ACM SIGCSE Bulletin*, 36, p. 119-150.

Lowther, J. and Lane, A. (2006). Relationship between mood, Cohesion and Satisfaction with Performance among Soccer Players, Athletic Insight: *Online Journal of Sport Psychology*, [Online], vol. 4, no. 3, pp. 18.

Lui, K. M. and Chan, K. C. C. (2004) A Cognitive Model for Solo Programming and Pair Programming, *Proceedings of the third IEEE International Conference on Cognitive Informatics* (ICCI'04).

Macfarlane, I.A. and Reilly, I. (1995), Requirements traceability in an integrated development environment, *Proceedings of the Second IEEE International Symposium on Requirements Engineering IEEE*, , pp. 116-123.

MacWhinney, B., Keenan, J. M. and Reinke, P. (1982). The role of Arousal In memory for conversation', *Cognition and Emotion*, Vol. 10, No. 4, pp. 308-317.

Mahr, W., Carlsson, R., Fredriksson, J., Maul, O. and Fjeld, M. (2006). Tabletop Interaction, *Research Alert in Proceedings of ACM NordiCHI*, pp. 499-500

Martin, L. L. (2001). Mood as input: A configural view of mood effects. in Theories of Mood and Cognition, in: *eds. L. L. Martine and G. L. Clore*, Mahwah, NJ.: Lawrence Erlbaum., pp. 135-158.

Maxwell, S.E. and Delaney, H. D. (2004). Designing Experiments and analyzing data: A model comparison prospective, *Lawrence Erlbaum Associates*, Publishers Mahwah, New Jersey.

Mayer, J. D., Allen J. P. and Beauregard, K. (1995). Mood Induction for Four Specific Moods: A Procedure Employing Guided Imagery Vignettes With Music, *Journal of Mental Imagery*, Vol. 19 (1and2). pp. 133-150.

Mayrhauser, A.V. (1995). Program comprehension during software maintenance and evolution, *IEEE Computer*, Vol. 28, Issue. 8, pp. 44-55.

McMorris, T. and Graydon, J. (1997). The effect of exercise on cognitive performance in soccer specific tests, *Journal of Sports Sciences*, Vol. 15, No. 5, pp. 459-468.

Menon, C., and Tashakkor, E. (2005). Wireless Mood Detector, courses.ece.uiuc.edu/ece445/projects/fall2005/project2_design_review.doc, Date accessed 19/12/08

Meunier, L. E. 1996. Human Factors in a Computer Assisted Foreign Language environment: The Effects of Gender, Personality, and Keyboard Control, *CALICO Journal*, Vol. 3, No. 2and3, pp. 47-72

Mikkelsen, S., Vilstrup, S., Lassen, C.F., Kryger, A. I., Thomsen, J. F. and Andersen, J. H. 2007. Validity of questionnaire self-reports on computer, mouse and keyboard usage during a four-week period, *Occupational and Envionmental Medicine*, Vol. 64, pp. 541-547

Moby. (1995). Hymn. On Everything is wrong [CD]. New York: Elektra

Morris, J. D. (1995). SAM: The self-assessment manikin, an efficient cross-cultural measurement of emotional response (observations), *Journal of Advertising Research*, vol. 35, No. 6, pp. 63-68.

Mozart, W.A. (1985). Sonata for two pianos in D major, K448 [Recorded by Perahia, M. and Lupu, R.]. *On music for piano, four hands [CD]*. London, Sony Classical.

Mujica-Parodi, L., Greenberg, T. and Kilpatrick, J. (2004), A Multi-Modal Study of Cognitive Processing under Negative Emotional Arousal, in: *26th Annual Meeting of Cognitive Science Society*.

Nasoz, F., Alvarez, K., Lisetti, C.L. and Finkelstein, N. (2003). Emotion recognition from Physiological Signals for presence Technologies, *International Journal of Cognition*, Technology, and Work - Special Issue on Presence, vol. 6, no. 1.

Neighbors, C., Larimer, M.E and Lewis, M. A. (2004). Targeting Misperceptions of Descriptive Drinking Norms: Efficacy of a Computer-Delivered Personalized Normative

Feedback Intervention, *Journal of Consulting and Clinical Psychology*, Vol. 72, No. 3, pp.434–447

Norman, D.A. (2004). Emotional design: why we love (or hate) everyday things. New York: Basic Books.

O'Hara, K. P., and Payne, S. J., (1999). Planning and the user interface: The effects of lockout time and error recovery cost, *International Journal of Human-Computer Studies*, Vol. 50, No. 1, pp. 41.59.

Pane, J. F. and Myers, B. A. (2000). The Influence of the Psychology of Programming on Language Design: Project Status Report', In: *12th Workshop of the Psychology of Programming Interest Group*, Cozenza, Italy, pp. 193-208.

Parberry, I. and Gasarch, W. (2002). Problems on Algorithms, *2nd Ed. Published by Prentice Hall inc*.

Parkinson, B., Totterdell, P., Briner, R. B. and Reynolds, S. (1996). Changing Moods: The psychology of mood and mood regulation, Harlow: Longman.

Peres, S.C., Tamborello II, F. P., Fleetwood, M.D., Chung, P. and Paige-Smith, D.L. (2004). Keyboard shortcut usage: The roles of social factors and computer experience, *Human Factors and Ergonomics Society 48th Annual Meeting*, pp. 803.

Picard, R. (2000). Affective Computing. United States. The MIT Press.

Picard, R.W. (1997). Recognizing and Expressing Affect, 1st edn, The MIT Press Cambridge, Massachusetts London, London.

Plutchik, R. (1980). Emotion: A psychoevolutionary synthesis. New York: Harper and Row

Poels, K. and Dewitte, S. (2006), How to Capture the Heart? Reviewing 20 Years of Emotion Measurement in Advertising, *Journal of Advertising Research*, vol. 46, no. 1, pp. 18-37.

Pressman, R. S. 2005. Software Engineering: A practitioner's approach, 6th Ed. McGraw Hills Inc.

Reason, J.T. (1984). Lapses of attention. In R. Parasuraman and R. Davies (Eds.), Varities of attention. New York: Academic Press.

Reeves, B., and Nass, C. (1996). The media equation: How people treat computers, television, and new media like real people and places, Stanford, CA: CSLI Publications.

Reinecke, L. and Trepte, S. , 2007-05-23 "All Work, No Play? The Effects of Mood Management Processes on Subsequent Cognitive Performance" Paper presented at the annual meeting of the International Communication Association, TBA, San Francisco, CA Online <APPLICATION/PDF>. 2009-02-04 from http://www.allacademic.com/meta/p170096_index.html

Reips, U.-D. (2000). The Web experiment method: Advantages, disadvantages, and solutions, In M. H. Birnbaum (Ed.), *Psychological experiments on the Internet*, San Diego, CA: Academic Press, pp. 89–117.

Remington, N.A. and Fabrigar, L.R. (1995). Reexamining the circumplex model of affect, *Journal of Personality and Social Psychology*, vol. 79, no. 2, pp. 286-300.

Robelin, M. and Rogers, P.J. (1998). 'Mood and psychomotor performance effects of the first, but not of subsequent, cup-of-coffee equivalent doses of caffeine consumed after overnight caffeine abstinence', *Behavioral Pharmacology*, Vol. 9, No. 7, pp. 611-618.

Roberts, T.S. (2006). The use of Multiple Choice Tests for Formative and Summative Assessment, In: *8th Australasian Computing Education Conference (ACE2006),* Hobart, Tasmania, Australia, Vol. 52.

Rocco, E. and Igou, B. (2001). The critical role of high-availability infrastructure support services, *Gartner Research,* January 2001

Rooijen, L.V. and Vlaander G.P.J. (2006). Dramatic Induction of depressive mood, *Journal of Clinical Psychology*, Vol. 40, No. 6, pp. 1318-1322.

Ross, J. M. and Zhang, H. (1997). Structured Programmers Learning Object-Oriented Programming: Cognitive Considerations, *SIGCHI Bulletin*, Vol. 29, No. 4.

Russ, S.W. and Kaugars, A.S. (2001). Emotion in children's play and creative problem solving, *Creativity Research Journal*, vol. 13, no. 2, pp. 211-219.

Russell, J.A. (1980). A Circumplex Model of Affect, *Journal of Personality and Social Psychology*, Vol. 39, No. 6, pp. 1161-1178.

Rusting, C.L. (1998). Personality, mood, and cognitive processing of emotional information: Three conceptual frameworks, *Psychological Bulletin*, vol. 124, no. 2, pp. 165-196.

Sakurazawa, s., Yoshida,N., Munekata, N., Omi,A., Takeshima, H., Koto, H., Gentsu,K., Kimura, K., Kawamura,K., Miyamoto, M., Arima,R., Mori,T., Sekiya,T., Furukawa,T., Hashimoto, T., Numata, H., Akita, J., Tsukahara, Y. and Matsubara, H. (2003). A Computer gaming using galvanic skin response, *ACM International Conference Proceeding Series*, Vol. 38, pp. 1-3.

Sanchez, J. A., Kirschning, I., Palacio, J.C. and Ostrovskaya, Y. (2005). Toward mood-oriented interfaces for synchronous interaction. *CLIHC'05*, October 23-26, Cuernavaca, Mexico.

Schacter, D. L. (1999). The seven sins of memory: Insights from psychology and cognitive neuroscience. American Psychology, 54, 182-203.

Schleifer, L.M., and Ley, R. (1994). End tidal PCO as an index of psychophysiological activity during VDT data-entry work and relaxation, *Ergonomics*, vol. 37, pp. 245-2

Schmitt, R. (1969). Martin Heidegger on being human: An introduction to "Sein and Zeit", Random House, New York.

Schneiderman, B. and Mayer, J.E. (1979). Syntactic/semantic interaction in programmer behaviour: A model and experimental results, *International Journal of Computer and Information Science*, Vol. 8, No. 3, pp. 219-238.

Schneidewind and Hoffman. (1979). An Experiment in Software Error Data Collection and Analysis. *IEEE TSE*, SE-5:3 (May 1979), 276-286.

Schwarz, N. and Bless, H. (1991). Happy and Mindless, but Sad and Smart? The Impact of affective states on analytic reasoning, *Emotions and Social Judgments*, eds. Forgas, J. P, Published by Routledge.

Shaochun, X. and Rajlich, V. (2004). Cognitive process during program debugging, In: *Proceedings of the Third IEEE International Conference on Cognitive Informatics*, pp. 176-182.

Shepard, R. J. (1997). Curricular Physical Activity and Academic Performance', *Pediatric Exercise Science*, Vol. 9, No. 2.

Sheskin, D.J. (1996). Parametric and Nonparametric Statistical Procedures, CRC Press, 1997.

Shneiderman, B (1980). Software Psychology: Human Factors in Computer and Information Systems, *Little Brown and Co*., Boston, Massachusetts

Shneiderman, B. (1978). Improving the Human Factors Aspect of Database Interactions, In: *ACM Transactions on Database Systems*, Vol. 3, No. 4, pp. 417-439.

Siemer, M. and Reisenzein, R. (1998). Effects of Mood on Evaluative Judgments: Influence of Reduced Processing Capacity and Mood Salience, *Cognition and Emotion*, Vol. 12, No. 6, pp. 783–805.

Silvestrini, N. and Gendolla, G. H. (2007). Mood effects on autonomic activity in Mood regulation, *Psychophysiology*, Vol. 4, pp. 650-659.

Silvia, P. J. and Abele, A. E. (2002), Can positive affect induce self-focused attention? Methodological and measurement issues, *Cognition and Emotion*, Vol. 16, No. 6, pp. 845-853.

Smith, A.P., Brockman, P., Flynn, R., Maben, A., and Thomas, M. (1993). Investigation of the Effects of Coffee on Alertness and Performance during the Day and Night, *Biological Psychology/ Pharmacopsychology*, Vol. 7, No. 4, pp. 217-223.

Smith, H. J., and Keil, M. (2003). The reluctance to report bad news on troubled software projects: a theoretical model. *Information Systems Journal*, Vol. 13, No. 1, pp. 69-95.

Steptoe, A. and Cox, S. (1988). Acute effects of aerobic exercise on mood, *Health Psychology*, Vol. 7, No. 4, pp. 329-340.

Swindells, C., MacLean, K.E., Booth, K.S. and Meitner, M. (2006). A Case-Study of Affect Measurement Tools for Physical User Interface Design, In: *Proceedings of Graphics Interface (GI) Canadian Human-Computer Communications Society*, Canada.

Takada, Y., Matsumoto, K. and Torii, K. (1994). A programmer performance measure based on programmers state transitions in testing and debugging process, In: *16th International Conference on Software Engineering*, Vol. 16, No. 16-21, pp. 123-132.

Thayer, R. E., Newman, J. R. and McClain, T. M. (1994). Self Regulation of Mood: Strategies for Changing a Bad Mood, Raising Energy, and Reducing Tension, *Journal of Personality and social psychology*, vol. 67, pp. 910-925.

Thompson, W.F., Schellenberg, E.G. and Husain, G. (2001). Arousal, Mood and the Mozart effect, *Psychological Science*, Vol. 12, No. 3.

Tice D. M., Bratslavsky, E. and Baumeister, R. F. (2001). Emotional Distress Regulation Take Precedence Over Impulse Control: If You Feel Bad, Do It!, *Journal of Personality and Social Psychology*, Vol. 80, No. 1, pp. 53-67.

Tukey, J. (1977). Exploratory Data Analysis, Addison-Wesley Press.

Wade, H., Clarann, W., Shirley, C. (2006). 'Influence of Computer Intervention on the Psychological Status of Chronically Ill Rural Women: Preliminary Results', *Nursing Research*, Vol. 55, No. 1, pp. 34-42.

Watters, P. A., Martin, F., and Schreter, Z. (1997). Caffeine and Cognitive Performance: The Nonlinear Yerkes-Dodson Law, *Human Psychopharmacolology*, Vol. 12, pp. 249-257.

Weiss, R. P. (2000). Emotion and Learning - implications of new neurological research for training techniques, *Training and Development*. ,Vol. 54, no. 10, pp. 46-50.

Westermann, R., Spies, K., Stahl, G., and Hesse, F. W. (1996). Relative effectiveness and validity of mood induction procedures: A meta analysis, *European Journal of Social Psychology*, Vol. 26, pp. 557-580.

Wingrove, J. and Bond, A.J. (1996). Impulsivity: A State as Well as Trait Variable. Does Mood Awareness Explains Low Correlations Between Trait and Behavioural Measures of Impulsivity?, *Personality and Individual Differences*, Vol. 22, No. 3, pp. 333-339.

Wynekoop, J.L., and Walz, D.B. (1998). Revisiting the perennial question: are IS people diferent?, In: *Database for advances in information sytems (SIGMIS),* vol. 29, issue. 2, pp. 62-72.

Xu, S. (2005). A Cognitive Model for Program Comprehension, In: *Proceedings of the 2005 third ACIS International Conference on Software Engineering Research, Management and Applications*, (SERA'05).

Yerkes, R. M. and Dodson, J. D. (1908). The relation of strength of stimulus to rapidity of habit-formation. *Journal of Comparative Neurology and Psychology*, Vol. 18, pp. 459-482.

Yildiz, B. E. (2007). How to Handle Time Problems in Computer Programming Culture? A Case Study on Women Software Developers in Turkey, In: *4th European Gender and ICT Symposium,*March 8-10, Univeristy of Helsinki.

Zarinpoush, F., Cooper, M. and Moylan, S. (2000). The effects of happiness and sadness on moral reasoning, *Journal of Moral Education*, vol. 29, no. 4, pp. 397-412.

Zhai, J. and Barreto, A. (2006). Stress Detection in Computer Users through Non-Invasive Monitoring of Physiological Signals, *Biomedical Science Instrumentation*, vol. 42, pp. 495-500.

Zheng, W. (2008). The Effects of Message Arousal and valence on Implicit and Explicit Memory of Pass, In: *Annual Meeting of the International Communication Association*, Dresden, Germany, http://www.allacademic.com/meta/p91350_index.html

Zimmermann, P., Guttormsen, S., Danuser, B. and Gomez, P. (2003). Affective Computing - A Rationale for Measuring Mood with Mouse and Keyboard. In: Internatational Journal of Occupational Safety and Ergonomics. Vol. 9, 539-551.

Zimmermann, P., Guttormsen, S., Danuser, B. and Krueger, H. (2002). Automatic Recognition of moods in HCI by means of mouse and keyboard, In: *Proceedings of the 6th International Scientific Conference on Work With Display Units*, WWDU (2002), eds. H. Luczak, A.E. Çakir and G. Çakir, World Wide Work, , pp. 344.

Appendix I: Debugging Questions

Note. This appendix contains the debug question used in the experiments described in chapter 3. Each debug question is followed by code statements and then their answers. The answer in bold is the correct one. Difficulty level of the questions is indicated after the answers.

Q.1. which of the following C/C++ statements contain variables whose values are replaced?

```
a) int b, c, d, e, f, I, j, k, p;
b) cin >> b >> c >> d >> e >> f;
c) p = i + j + k + 7;
d) cout << "variables whose values are destroyed";
e) cout << "a = 5";
```

Answers

1. Statements (b) and (c).
2. Statements (c) and (d)
3. Statements (d) and (a)
4. Statements (c) and (a)

Difficulty Type = **Easy**

Q.2. which variable value is not changed?

```
int a,  b=3, c=2, d;
a = b+c;
d = a + c;
a = a – (b+c);
```

Answers

1. Variable a
2. Variable b
3. Variable c
4. All variables changed

Difficulty Type = **Easy**

Q.3. what is the order of evaluation of the operators in each of the following C/C++ statement? Also state the value of x.

```
x = 2 % 2 + 2 * 2 - 2 / 2;
```

1. *,%,/,-,+,=; x=4
2. %,*,/,+,-,=; x=3
3. +,-,%,*,/,=; x=0
4. No one is correct

Difficulty Type = **Medium**

Q.4. what is the order of evaluation of the operators in each of the following C++ statement? Also state the value of x.

```
x = ( 3 * 9 * ( 3 + ( 9 * 3 / ( 3 ) ) ) );
```

1. *,/,*,*,+,=; x=24
2. No one is correct
3. +,*,*,/,*,=; x=32
4. ***,/,+,*,*,=; x=324**

Difficulty Type = **Medium**

Q.5. Identify the error(s) from the following statements:

```
if (age >= 65);
    cout << "Age is greater than or equal to 65" << endl;
else
    cout << "Age is less than 65” << endl;
```

1. No error
2. else not properly used
3. **Semicolon after if**
4. Both 2 and 3

Difficulty Type = **Easy**

Q.6. what is error (logical, syntax) in the following?

```
int x = 1, total;
while ( x <= 10 ) {
    total += x;
    ++x;
}
```

Answers

1. x pre incremented
2. No error
3. while never executes
4. **total not initialized**

Difficulty Type = **Medium**

Q.7. what is error (logical, syntax) in the following?

```
Int  main()
{
    float sum;
    int n=1, count=0;
    while (n > 0)
   {
        scanf("%d", andn);
        sum = sum + n;
         count++;
  }
   return(0);
}
```

Answers

1. n is not initialized
2. **sum is not initialized**
3. count++ will never execute

Difficulty Type = **Easy**

Q.8. what is the error (logical, syntax) in the following?

```
double square( double number )
{
   double number;
   return number * number;
}
```

Answers

1. **variable number is declared twice**
2. No error
3. return type no matching

Difficulty Type = **Easy**

Q.9. what is the error (logical, syntax) in the following?

```
main()
{
      cout << recursive_call(5);
}

int recursive_call(int n)
{
    return recursive_call(n-1);
}
```

Answers

1. No base case
2. Infinite recursion
3. No error
4. **both 1 and 2**

Difficulty Type = **Medium**

Q.10. what is the error (logical, syntax) in the following?

Assume that:

```
int a[ 3 ];
cout << a[ 1 ] << " " << a[ 2 ] << " " << a[ 3 ] << endl;
```

Answers

1. No error

2. Syntax error
3. **A[3] is not valid location**

Difficulty Type = **Easy**

Q.11. what will be the output of the following code?

```
int x = 3;
cout << x++ << " ";
x--;
--x;
cout<<x << " ";
cout<< --x<<" ";
x+=2;
cout<< --x;
```

Answers

1. 3 2 1 2
2. 4 2 1 2
3. 4 3 2 3
4. None

Difficulty Type = **Medium**

Q.12. what is the output of the following code segment?

```
int i,j;
for (i=2;i<4;++i)
  for (j=1;j<3;++j)
   cout << i << " " << j << " ";
```

1. 2 1 3 1
2. No output
3. **2 1 2 2 3 1 3 2**
4. Error in code

Difficulty Type = **Medium**

Appendix II: Algorithms

Note. This appendix contains the algorithms used in the experiment described in Chapter 6. Each algorithm is followed by requirement from participant and difficulty level.

Algorithm:

Consider the following algorithm

```
1.   for i := 1 to N
2.        for j := 1 to i
3.             for k := 1 to j
4                   Change line
5              End for k
6         End for j
7    End for i
```

Requirement:

What will be the output from the algorithm above. Suppose N = 3

Difficulty Level: **Easy**

Algorithm:

Consider the following algorithm

```
1.   Sum: = 0
2.   for i := 1 to N
3.        Sum: = Sum + i
4.   End for loop
```

Requirement:

Identify all the variables and create a trace table from the algorithm above. Suppose N = 7

Difficulty Level: **Easy**

Algorithm:

Consider the following two matrices of 2x2 orders, initialized as follows

```
Metrics M1     Metrics M2
1    2         10   20
3    4         40   50
```

Consider following algorithm

```
1.   for i = 1 to Number of rows
2         for j = 1 to Number of cols
3                M3 (i , j) := M1 (i , j) + M2 (i , j)
4         End for j loop
```

```
5    End for i loop
```
Requirement:
Identify all the variables and create a trace table from the algorithm above.

Difficulty Level: **Easy**

Algorithm:
Consider the algorithm below.

```
1.   i := 0, j := 0, v := 1, N = 4
2.   for i:= 1 to N
3         for j := 1 to N – i
4              Output " "
5         End for loop j
6         for k := j to j + v
7              Output "*"
8         End for loop k
9         v := v + 2
10        Change Line
11   End for loop i
```

Requirement:
Identify all the variables and create a trace table from the algorithm above. Suppose N = 7.

Difficulty Level: **Medium**

Algorithm:
Consider the algorithm below.

```
1.   a := 1, b := 1, c:= 0, N := 7
2.   if (N < 2)
3.        Output "Enter 2 or greater number"
4.   else
5.        for i := 3 to N
7.             c = a + b
8.             a = b
9.             b = c
10             output a + " " + b + " " + c
11.       End for loop
12.  End else
```

Requirement:
Identify all the variables and create a trace table from the algorithm above.

Difficulty Level: **Medium**

Algorithm:
Consider the following two matrices M1 and M2 of order 2x2

```
Metrics M1     Metrics M2
1    2         10   20
4    5         40   50

Consider following algorithm
1.   I := 0, j:= 0, k:= 0
2.   for i=1 to Number of rows
3.        for j: = 1 to Number of cols
4.             M3 (I,j) := 0
5.             for k := 1 to number of rows
6.                  M3(i.j) := M3(i,j) + M1(I,k) *
M2(k,j)
7.             End for k
8.        End for j
9.   End for i
```

Requirement:
Identify all the variables and create a trace table from the algorithm above. Consider the values of the metrics as stated above.

Difficulty Level: **Medium**

Algorithm:
Consider the algorithm below.

```
1.   N:= 21, stringVar:=""
2.   n := N
3.   if (n > 0)
4.        while (n > 0)
5.             stringVar := stringVar + (n mod 2)
6.             n := n/2
7.        End while
8.        stringVar := reverse(stringVar)
9.   Output stringVar
```

Requirement:
Identify all the varibales and create a trace table from the algorithm above.

Difficulty Level: **Difficult**

Algorithm:
Consider the algorithm below.

```
1    function CallItself(array, arraySize)
2    if (arraySize > 1)
3         for i:= 1 to arraySize - 2
4              if array(i) > array(i+1)
5                   swap (array(i), array(i+1)
```

```
6              End if
7            End for loop
8            CallItself(array, arraySize – 1)
9    End if
10   End Function
```

Requirement:
Identify all the variables and create a trace table from the algorithm above. Consider that the following array is passed in the function.
1, -1, 2, 3, 4, 5, 6, 9

Difficulty Level: **Difficult**

Algorithm:

```
1    Function ThisFunction(n)
2    if (n = 0 or n = 1)
3.        return 1
4.   else
5.        return n * ThisFunction(n-1)
6.   End If
7.   End Function
```

Requirement:
Identify all the variables and create a trace table from the algorithm above. Consider n = 7
Difficulty Level: **Difficult**

Algorithm:
Consider the following piece of code

```
1.   Max: = 0;
2.   for i: = 1 to length of Array
3.   If (Array (i) < Max)
4.        Max: = Array (i)
5.   End for loop
```

Requirement:
Suppose Array has following values in it.
12, 3, 4, 53, 34, 3, -1, -9, 4, 4
Identify all the variables and create a trace table from the algorithm above.

Difficulty Level: **Easy**

Algorithm:
Consider the following piece of code

```
1.   Search: = Some Number
2.   for i: = 1 to length of Array
3.        If (Array (i) = Search)
4.             Output Search Found at Position i
```

```
5.              Exit
6.         End If
7.    End for loop
8.    Output "Number Not found"
```

Requirement:
Suppose Array has following values in it.
12, 3, 4, 53, 34, 3, -1, -9, 4, 4
Identify all the variables and create a trace table from the algorithm above.

Difficulty Level: **Easy**

Algorithm:
Consider the following piece of code

```
1.    Base: = Some Number
2.    Exponent: = Some Number
3.    Temp: = Nothing
4.    for i: = 1 to Exponent
5.         Temp = Temp * Base
6.         Output Temp
7.    End for loop
8.    Output Temp
```

Requirement:
Suppose Exponent has a value of 6 and Base has value of 5
Identify all the variables and create a trace table from the algorithm above.

Difficulty Level: **Easy**

Algorithm:
Consider the following algorithm

```
1.    i := 0, j:= 0
2.    for i:= 1 to length of array
3.         for j:= 1 to length of array – i
4.              if array(j) > array(j+1) than
5.                   swap( array(j), array(j+1))
6.              end if
7.         end for j
8.    end for i
```

Requirement:
Consider the following array
1, 2, 4, 0, 4, 9, 1, 2
Identify all the varibales and create a trace table from the algorithm above.

Difficulty Level: **Medium**

Algorithm:
Consider the following algorithm.

```
1.   Temp:= 0
2.   for i:= 1 to N
3.        Temp: = Temp * i
4.        Output Temp
```

Requirement:
Identify all the variables and create a trace table from the algorithm above. Assume N = 13

Difficulty Level: **Medium**

Algorithm:
Consider the following algorithm.

```
1.   for i := 1 to N
1.        x:= 8, y:= 9
2.        x:= x + y
3.        y:= x – y
4.        x:= x  - y
5.   End for i
```

Requirement:
Identify all the variables and create a trace table from the algorithm above. Assume N = 7.

Difficulty Level: **Medium**

Algorithm:
Consider the algorithm below.

```
1    Function RecursiveFibonacci(N)
2         if (n = 0) or (n=1)
3              return n
4         else
5              return RecursiceFibonacci(N-1) +
RecursivcFibonacci(N-2)
6         End If
7    End Function
```

Requirement:
Identify all the varibales and create a trace table from the algorithm above. Assume N = 8.

Difficulty Level: **Difficult**

Algorithm:

```
1. s1:= 0, s2 := 0
2.   Function RecursiveS (Array, N)
3.        if (n <= 1)
4.             return array(0)
5.        else
6.             s1 := RecursiveS(array, N - 1)
7.             s2 := s1 + array(N-1)
8.        End else
9.   return s2
10.  End function
```

Requirement:
Trace all the variables in the above algorithm execution. Consider array passed in as the following array
1, 2, -1, 2, 4, 5, 36, 17. Here n = length of array

Difficulty Level: **Difficult**

Algorithm:
Consider the following array
1, 2, -1, 2, 4, 5, 36,17

```
1.   Function ReturnM(array, N)
2.        M1 : = 0
3.        If (N = 1)
4.             Return array(0)
5.        Else
6.             M1:=  ReturnM (array, N-1)
7.             If (array(N-1) > M1)
8.                  Return array(N-1)
9.             Else
10.                 Return M1
11.            End Else
12.       End else
13.  End Function
```

Requirement:
Identify all the variables and create a trace table from the algorithm above. Consider the following array was passed in the array
1, 2, -1, 2, 4, 5, 36,17 Here N = length of array

Difficulty Level: **Difficult**

Algorithm:
Consider the following algorithms

```
1.   Function factorInRange( k, n)
2.   if (k >= n)
3.         return false
4.   else if ((n mod k) = 0)
5.         return true
```

```
6.   else
7.         return factorInRange(k+1, n)
8.   End Fuction
9.
10.  Function Check(n)
11.     return ((n > 1) and <>factorInRange(2, n)
12.  End Check
```

Requirement:
Suppose n =23, n = 21, n = 17, n = 14, n = 10. What will be the Output of the function Check if the n will be input in the algorithm?
Difficulty Level: **Difficult**

Algorithm:
7. Consider the following algorithm

```
1.   Function Add(a,b)
2.        if (a = 0)
3.             return b
4.        else
5              return Add(a-1,b+1)
6    End
```

Requirement:
What will be the output of the algorithm above if a = 9 and b = 7. Also create a trace table

Difficulty Level: **Difficult**

Algorithm:
Consider the following algorithms

```
1.   n := 6
2.   procedure add (item : items)
3.        if rear=n
4.             Output "Array full"
5.        else
6.             rear :=rear+1
7.             q[rear]:=item
8.        end else
9.   end{of add}

10.  procedure delete ()
11.       if front = rear
12.            Output "Array empty"
13.       else
14.            front := front+1
15.            item := q[front];
16.       end else
17.  end {of delete}
```

Requirement:
What will be the value of array after each of the following operations?
add(T), add(o), add(u), delete(), delete(), add(j), add(k), add(g), add(f), delete(), delete(), delete(), add( ),add(i), add(f),add(t), add(i).
Consider rear, front and n as global variable

Difficulty Level: **Medium**

Algorithm:
Consider the algorithms below

```
1.   Procedure add(item)
2.        if top = n
3.             Output " Array full"
4.        else
5.             top := top+1;
6.             array(top) := item;
7.        End else
8.   end {of add}

9.   Procedure delete()
10.       if top = 0
11.            Output " Array Empty"
12.       else
13.            item := Array(top)
14.            top := top - 1
15.       End else
16.       return item
17.  end {of delete}
```

Requirement:
What will be the value of array after each of the following operations?
add(T), add(o), add(u), delete(), delete(), add(j), add(k), add(g), add(f), delete(), delete(), delete(), add( ),add(i), add(f),add(t), add(i).
Consider top as global variable. Consider n = 7

Difficulty Level: **Medium**

Algorithm:
Consider the following algorithm

```
1    let := 0, dig := 0, other := 0
2    chr := ' '
3    while chr <> 'Esc Key'
4         if ((chr >= 'A' and chr <= 'Z') or (chr >= 'a'
and chr <= 'z'))
```

```
5               let := let + 1
6          else if (chr >= '0' and chr <= '9')
7               dig := dig + 1
8          else
9               other := other + 1
10   End while
```

Requirement:
Dry run the algorithm above and create a trace table.

Difficulty Level: **Easy**

Algorithm:
Consider the following algorithm

```
1.   n := 10, temp := 1
2.   for i := 1 to n
3.        for j := 1 to i
4.             Output j + '*'
5.             temp := temp * j
6.        End for j
7.        Output " = " + temp
8.        Change line
9.        temp := 1
10.  End for i
```

Requirement:
What will be the output of the algorithm above? Trace table is also required.

Difficulty Level: **Easy**

Appendix III: Consent form used in the experiments

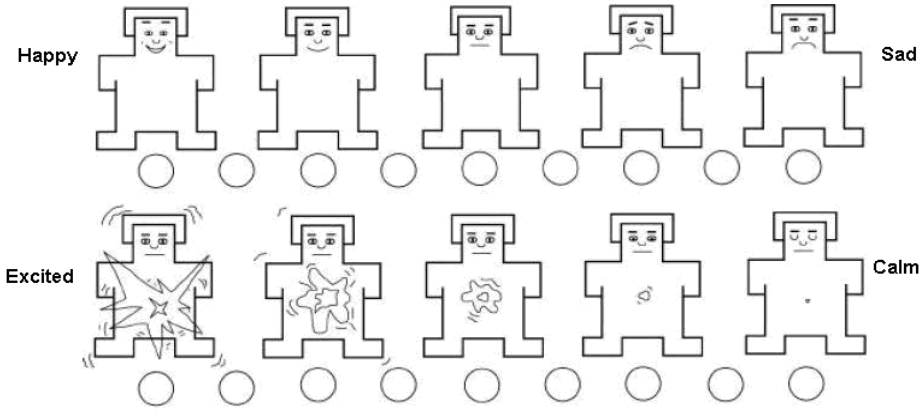
A.III.1. Form for self reporting moods for mood validation study

Before we start please answer the following questions

i. **What is your age in years? _____**
ii. **What is your gender? (Male/Female). Tick right answer**
iii. **Have you any programming experience? (Yes/No) _____**
iv. **If any programming experience rate your level from 1 to 5. 1 being a novice to 5 being an expert. _____**

Sample

Movie 1



Movie 2

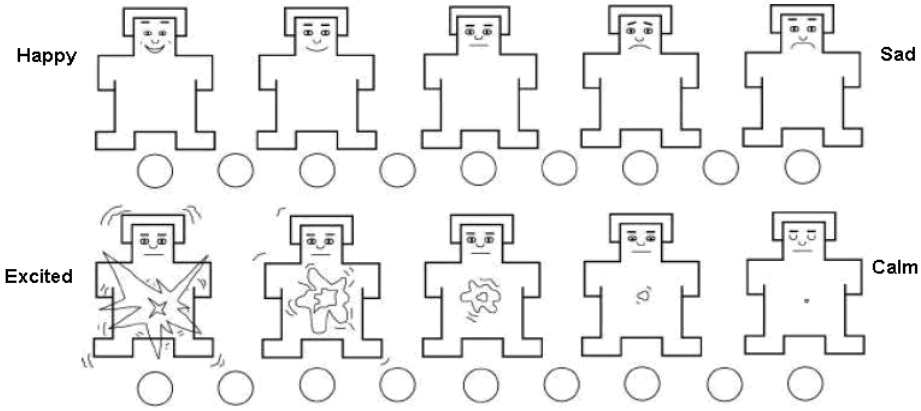
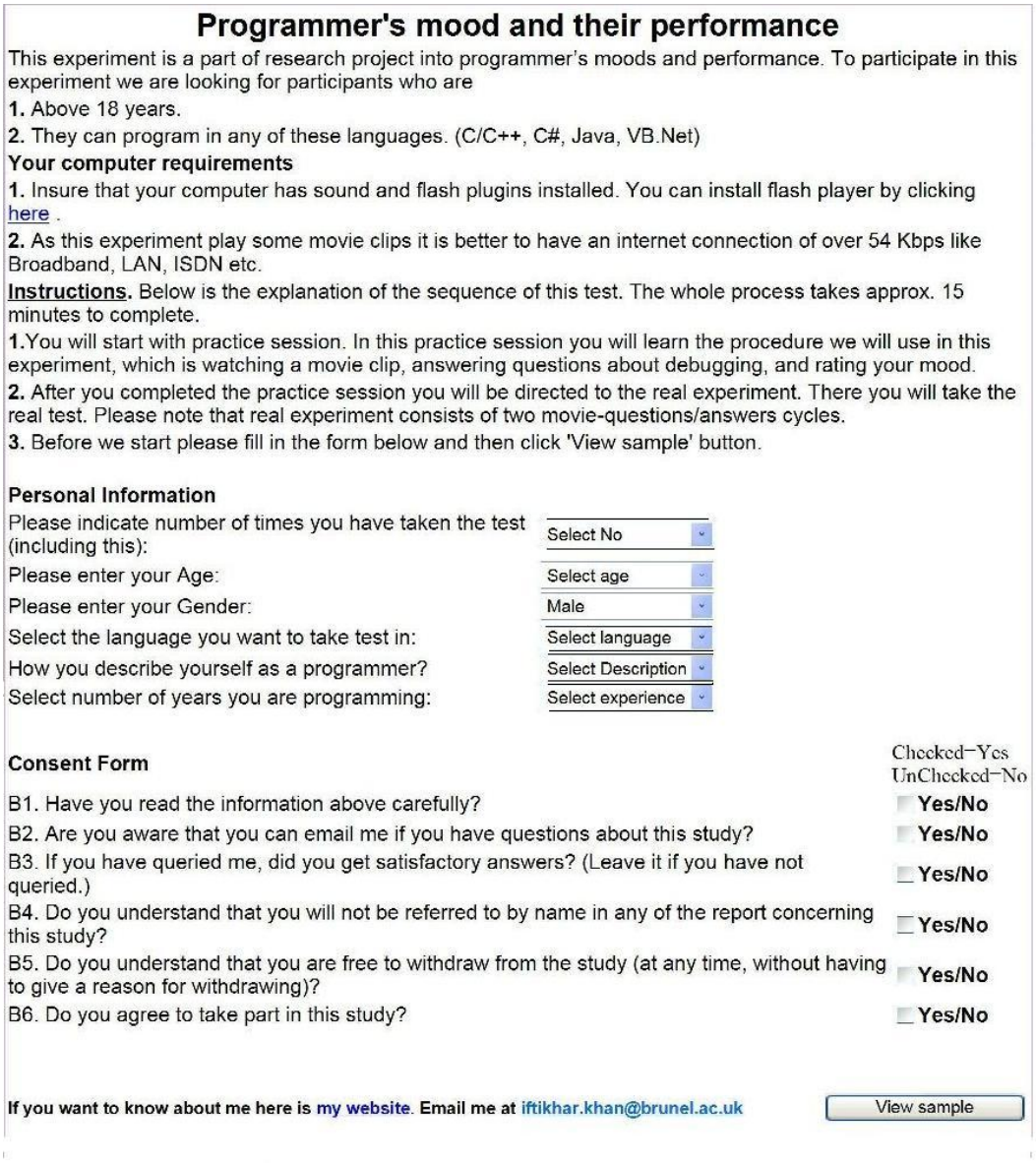


Figure AIII.1: *Self Assessment Manikin for self reporting of moods. Row one is used to report valence (pleasantness) ranging from happy figure to sad figure, whereas row 2 is used to report arousal ranging from excited figure representing high arousal to sleep\calm figure representing low arousal.*



Programmer's mood and their performance

This experiment is a part of research project into programmer's moods and performance. To participate in this experiment we are looking for participants who are

1. Above 18 years.

2. They can program in any of these languages. (C/C++, C#, Java, VB.Net)

Your computer requirements

1. Insure that your computer has sound and flash plugins installed. You can install flash player by clicking here .

2. As this experiment play some movie clips it is better to have an internet connection of over 54 Kbps like Broadband, LAN, ISDN etc.

Instructions. Below is the explanation of the sequence of this test. The whole process takes approx. 15 minutes to complete.

1.You will start with practice session. In this practice session you will learn the procedure we will use in this experiment, which is watching a movie clip, answering questions about debugging, and rating your mood.

2. After you completed the practice session you will be directed to the real experiment. There you will take the real test. Please note that real experiment consists of two movie-questions/answers cycles.

3. Before we start please fill in the form below and then click 'View sample' button.

Personal Information

Please indicate number of times you have taken the test (including this): Select No

Please enter your Age: Select age

Please enter your Gender: Male

Select the language you want to take test in: Select language

How you describe yourself as a programmer? Select Description

Select number of years you are programming: Select experience

Consent Form

Checked=Yes
UnChecked=No

B1. Have you read the information above carefully? **Yes/No**

B2. Are you aware that you can email me if you have questions about this study? **Yes/No**

B3. If you have queried me, did you get satisfactory answers? (Leave it if you have not queried.) **Yes/No**

B4. Do you understand that you will not be referred to by name in any of the report concerning this study? **Yes/No**

B5. Do you understand that you are free to withdraw from the study (at any time, without having to give a reason for withdrawing)? **Yes/No**

B6. Do you agree to take part in this study? **Yes/No**

If you want to know about me here is my website. Email me at iftikhar.khan@brunel.ac.uk View sample

Figure AIII.2: *Information and Consent form for Experiment 1A. (Web Study: H_1:- Moods affect programmers' coding\debugging performance)*

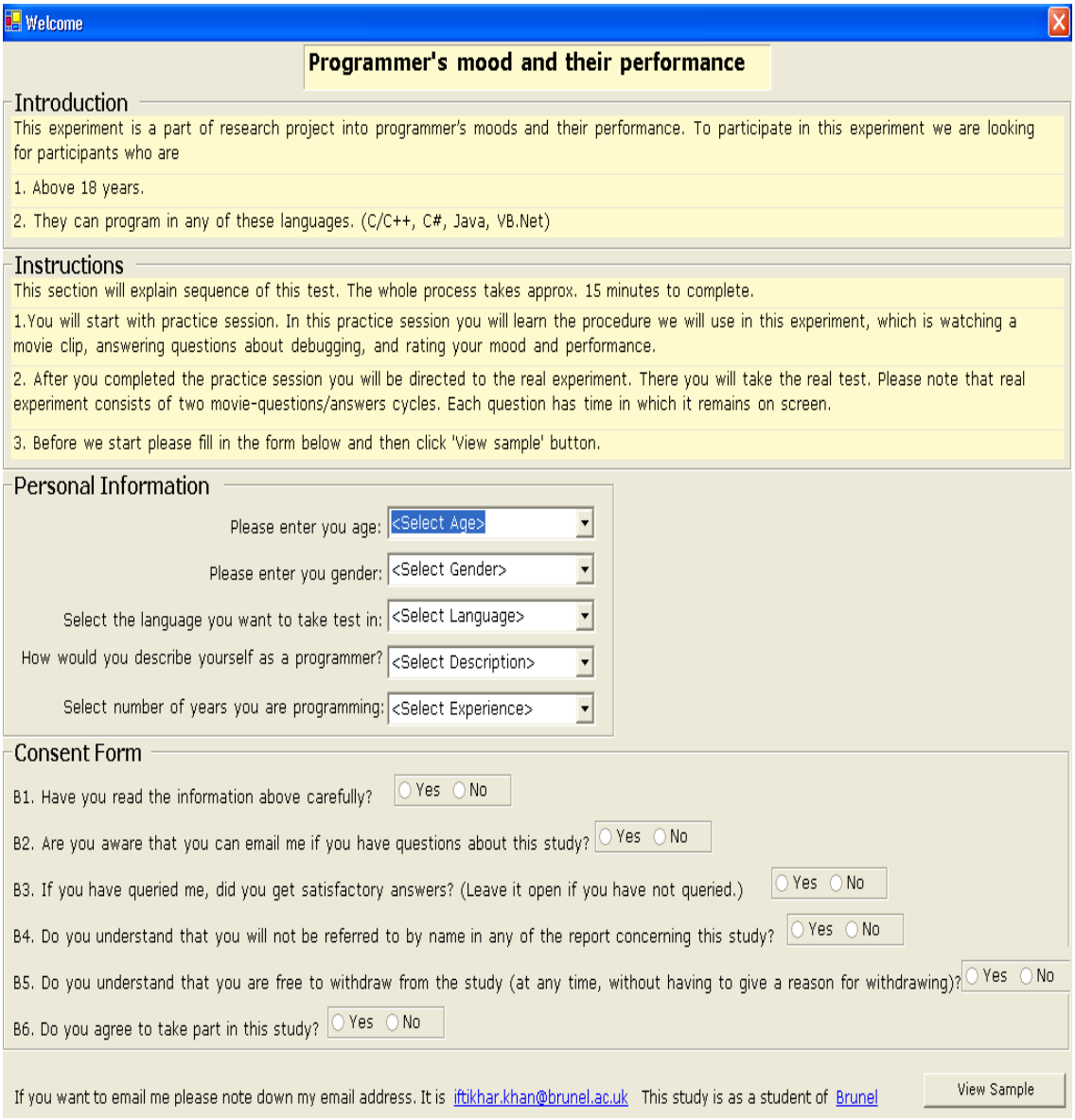


Figure AIII.3. *Information and consent form for experiment 1B. (Lab Study: H1:- Moods affect programmers' coding\debugging performance)*

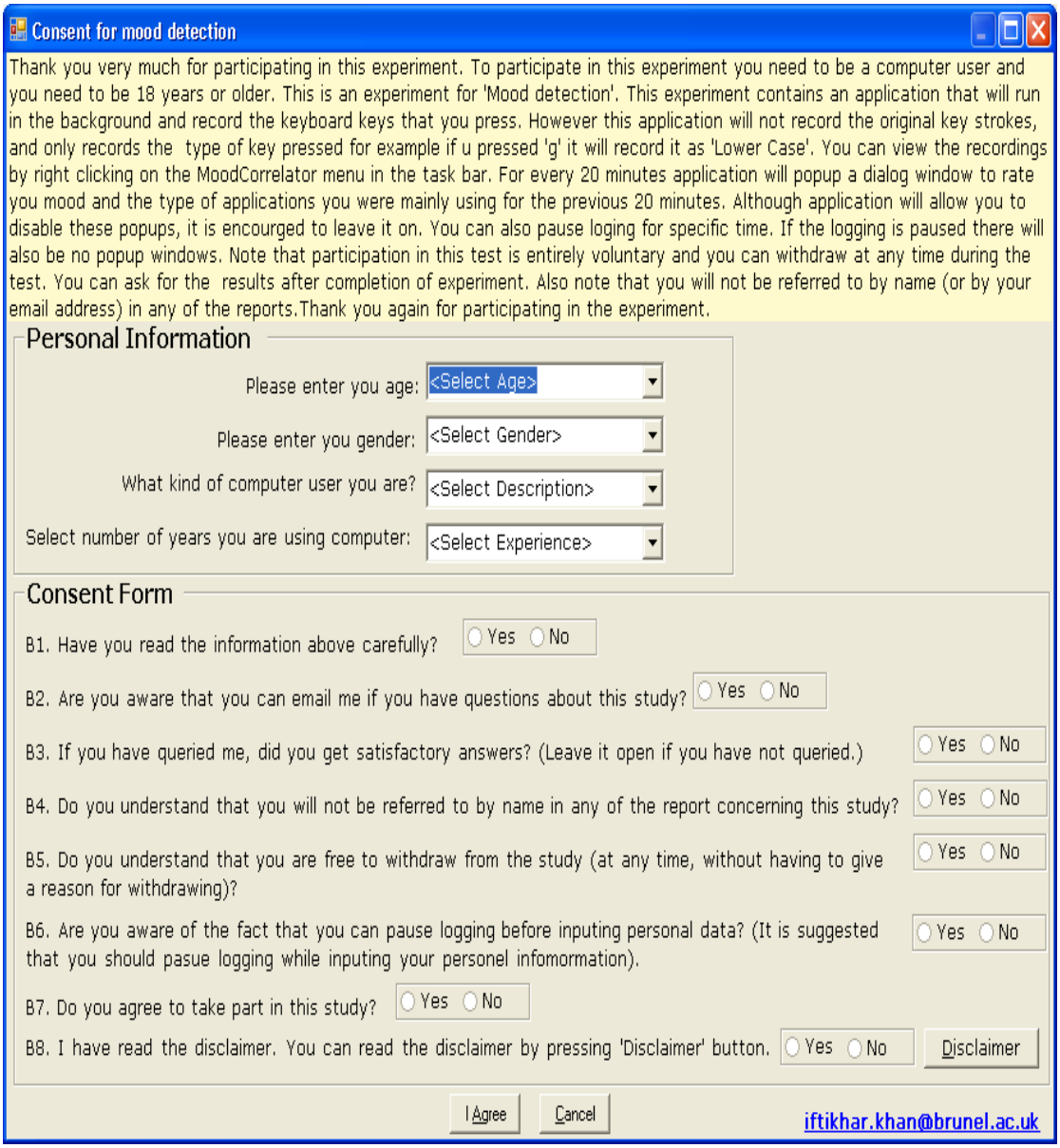


Figure AIII.4: *Information and consent form for experiment 2 (H2:- Programmers' mood can be measured from interaction behaviour with keyboard and mouse)*

Experiment 2B: Consent and Instructions Forms

Instructions:

This experiment contains an **application** that runs in background and will record the keyboard keys that you will press. The application will popup a **mood rating dialog** about 6 times in an hour on which you have to rate your mood. This experiment involves attaching a **GSR (Galvanic Skin Resistance)** meter with the fingers while you will do some programming in **Alice (Teaching language for children).** We will give you a story written on the paper which you will try to simulate in Alice. Please try to write the basic ideas and what you are intending to do in the editor provided with application. You can open **editor** by double clicking on the key icon in the task bar.

To make it easy for you we have divided the story into scenes. You have to just create the scenes if you want however there is no restriction. The basic purpose is to simulate the story. Don't worry if you are not able to exactly simulate things. All we require is your struggle to simulate the story. The application will also play some background music nominated by you and from us. Please try to feel the **mood of the music** while listening and working on task(s).

Please fill in the following information before you start experiment.

What is you age? _____ Years

What is your gender? _____

What kind of computer user you are? Tick one.

1) Novice 2) Medium User 3) Expert User 4) Programmer

For how many years you are using computer? _____ Years

Mood detection from the use of keyboard and mouse

Thank you very much for being a part of this experiment. To participate in this experiment you need to be a computer user and you need to be 18 years or older. This is an experiment for 'Mood detection from the use of keyboard and mouse'. This experiment contains an **application** that runs in background and will record the keyboard keys that you will press. The application will popup a **mood rating dialog** about 6 times in an hour on which you have to rate your mood. This experiment involves attaching a **GSR (Galvanic Skin Resistance)** meter with the fingers while you will do some programming in **Alice (Teaching language for children)**. You should have a training session for using Alice. This experiment will take about an hour. Note that participation in this experiment is completely voluntary and you can withdraw at any time during the experiment. You can ask for the results after the experiment. Also not that you will not be referred to by name or by any of your identification in any of the reports.

Please also fill the following consent form. Thanks again for participation.

1. Have you read the information above carefully? Tick one.

   1. Yes **2. No**

2. Are you aware that you can email me if you have any questions about this experiment? Tick one**.** **1. Yes** **2. No**

3. Do you understand that you will not be referred to by name in any of the report concerning this study? Tick one. **1. Yes** **2. No**

4. Do you understand that you are free to withdraw from the study (at any time, without having to give a reason for withdrawing)? Tick one.

   1. Yes **2. No**

5. Do you agree to take part in this study? Tick one. **1. Yes** **2. No**

Signature _____ **Date _____**

Task 1

Story - The Hare and Tortoise (Aesop Stories, 2007)

There once was a speedy hare who bragged about how fast he could run. Tired of hearing him boast, slow and steady, the tortoise, challenged him to a race. All the animals in the forest gathered to watch.

Hare ran down the road for a while and then and paused to rest. He looked back at Slow and Steady and cried out, "How do you expect to win this race when you are walking along at your slow, slow pace?"

Hare stretched himself out alongside the road and fell asleep, thinking, "There is plenty of time to relax."

Slow and Steady walked and walked. He never, ever stopped until he came to the finish line.

The animals who were watching cheered so loudly for Tortoise, they woke up Hare.

Hare stretched and yawned and began to run again, but it was too late. Tortoise was over the line.

After that, Hare always reminded himself, "Don't brag about your lightning pace, for Slow and Steady won the race!"

Scene 1 (For Programming in Alice)
Hare bragging himself that he can run fast.
Tortoise saying, “I am tired of your boasting. Run a race with me”.

Scene 2
Hare and tortoise on a starting line and some animals gathered around them to watch the race and saying hurrah.

Scene 3
Race started. Hare ran fast. Tortoise started walking slowly

Scene 4
Rabbit thinking that he has a lot of time. Why not to get a sleep

Scene 5
Rabbit sleeping while tortoise passing from nearby

Scene 6
Tortoise reached the finish line and rabbit cruising toward finish line. Tortoise saying “Slow and steady wins the race”

Task 2

Story - The cow and the Frog (Aesop, no date)

A young frog set out on his first adventure. As he came out of the pond he saw a large cow grazing in a field. Having never before seen such a creature, he hopped excitedly to his father, the bullfrog, and said, "I have just seen the biggest frog in the world!

"Humph!" said the bullfrog "Was he as big as me?" and he puffed himself up.

"Oh, much bigger than that!" said the little frog.

"Was he THIS big," said the bullfrog, puffing himself up even larger.

"Much, much bigger than you!" said the little frog.

"Ridiculous!" said the bullfrog, who fancied himself much more important than he was. "He couldn't be bigger than me! I'm the oldest frog in the pond. I was here first! Was he bigger than THIS?"

He puffed and puffed himself up so much...he burst!

Scene 1 (For Programming in Alice)
- A bullfrog and a young frog going along near a grazing field with a cow in the field Young frog saying “Wow! I never saw such a big frog” when he saw a cow.

Scene 2
- Hoping toward his father
- Stop hopping and saying to his father “I have just seen the biggest frog in the world!”
- Bullfrog: Puffed him up and said “Humph! Was he as big as me?”
- Young frog: "Oh, much bigger than that!"
- Bullfrog: Puffed him even more and said "Was he THIS big"
- Young frog: "Much, much bigger than you!"
- Bullfrog: Saying "Ridiculous!", "He couldn't be bigger than me! I'm the oldest frog in the pond. I was here first! Puffed him more and said “Was he bigger than THIS?" Bullfrog puffed and puffed himself

Scene 3
- Young frog and busted Bullfrog

Note: It is helpful to draw the scenes first and then start working. It will make the work much easier. There are no restrictions of following the same scenes. You can reduce or increase scenes. Important thing is the scenes should describe the whole story.

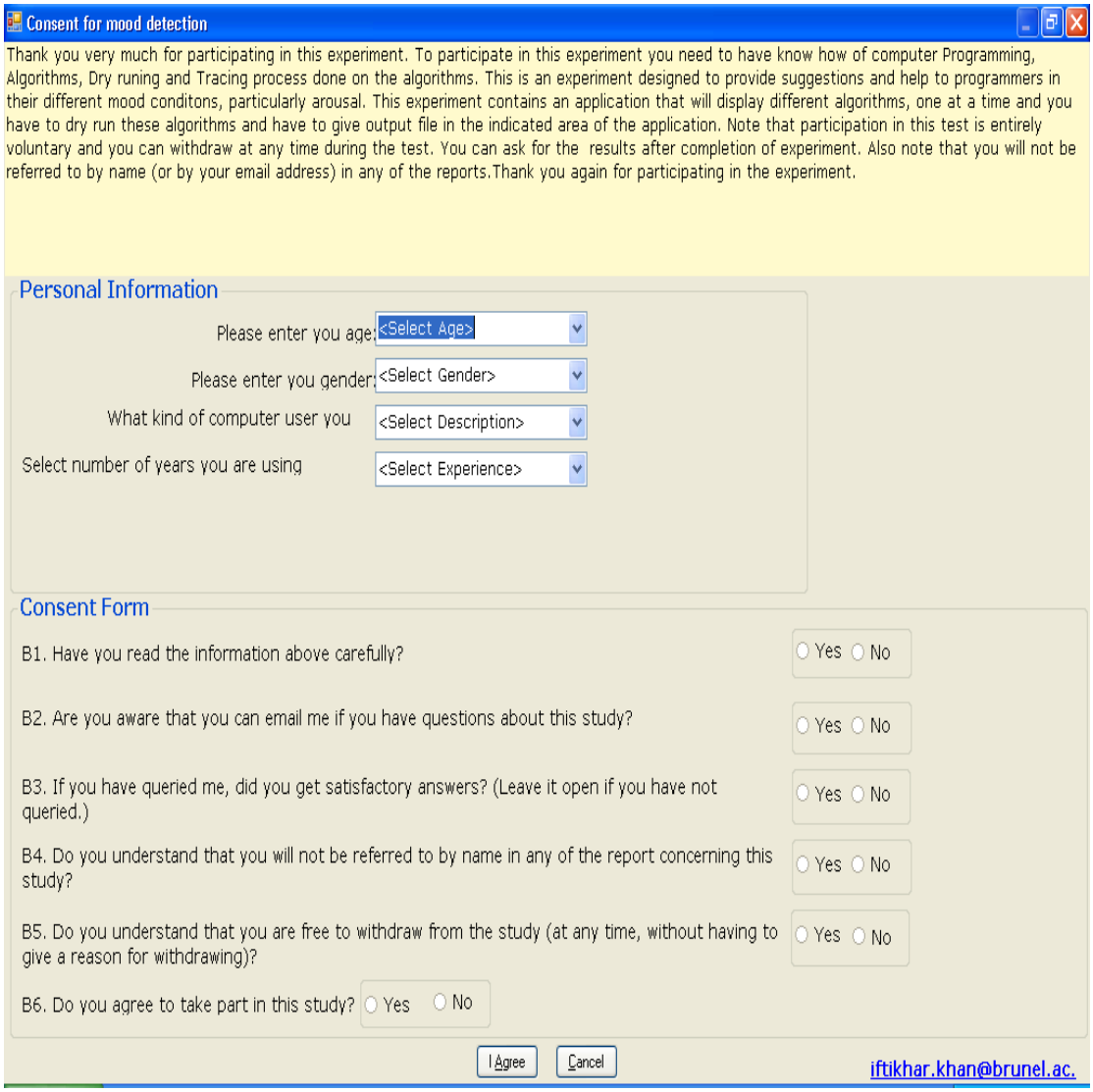


Figure AIII.6: *Information and consent form for experiment 3 ($H_{3:}$- An intervention while programmers' working can improve programmers' performance)*