Normative Manipulation as a way of Improving the Performance of Software Engineering Groups: Three Experiments

Alvin Teh

A thesis submitted for the degree of Doctor of Philosophy at The Australian National University

September 2012
© Alvin Teh

Typeset in Computer Modern by TeX and \LaTeX 2ε.
Except where otherwise indicated, this thesis is my own original work.

Alvin Teh
14 September 2012
To my family.
Acknowledgements

First, to my wonderful wife, who is endlessly patient and all around awesome and to my family, who have helped me in more ways than I can reasonably expect, I thank you. What I owe can possibly never be repaid.

Secondly, to Clive Boughton, who took on an eccentric upstart that he had never heard of and filled the role of quite possibly one of the best supervisors a PhD student could have hoped for - I am grateful. Being able to do what I wanted to do in research and not just continuing someone else’s research was both an eye popping and eye opening experience that I am well aware that not many others enjoy. For being everything short of a Father figure while I did my PhD away from home in matters both personal and work related, I am beyond grateful.

Third, to Dirk van Rooy and Elisa Baniassad, who were endlessly patient and helpful - I thank you. Your comments, experience and skill have given this research a huge leg up from its original form.

I would also like to thank Hwan-Jin Yoon of the Australian National University Statistical Consulting Unit on his advice on all matters statistics, all done for free, and all the participants for all my case studies, without which this thesis would be empty indeed.

Last but not least, I would like to thank the usual suspects of the CSIT break room haunt - you know who you are - may you always shoot first and laugh loudest.
Abstract

As the size of Software development projects increase, so too do the number of people working on these projects. This increase of software groups have brought about a new focus on the sociological issues associated with Software Development. There is a growing body of work that seeks to understand how Software Engineers work together effectively as a group, and also to identify factors that enable increased productivity consistently. Social Psychology looks at the interactions between individuals in groups (group dynamics) and may provide an applicable means to address this increased need for enhanced group effectiveness in Software Engineering.

The thesis of this research is that it is possible to apply Social Psychology research (in particular Normative Manipulation) to Software Engineering groups. It is possible to use Normative Manipulation effectively to increase the performance of Software Engineering groups in some types of tasks, and finally, this technique is adoptable by practising Software Engineering groups as it is non-intrusive.

Normative Manipulation is a technique in which particular behaviours are made to be favoured by group members. This behaviour is then actively practised by all group members. These particular behaviours may in turn increase the effectiveness of groups on particular tasks - for instance, a group favouring the behaviour of objectivity would then be more inclined to assess provided information its logical merits and may then in turn be more likely to uncover other related, but less obvious information. Since the success of elicitation and specification of Software Requirements is related to how complete the produced specification is, then it follows that such a group could possibly have increased performance in Software Elicitation and Specification tasks.

We demonstrate the validity of the thesis claims by performing three experiments. The first experiment attempts to replicate the results of a Social Psychology experiment on a sample of participants drawn from a Software Engineering population. The second experiment attempts to show that it is possible to affect the effectiveness of Requirements Elicitation by Software Engineering groups by instilling different norms. The third experiment applies Normative Manipulation on a practising Software Group to identify if the technique can be applied transparently as part of a normal Requirements Elicitation task.
Contents

Acknowledgements vii

Abstract ix

1 Introduction 1

1.1 Creating effective software teams 2
  1.1.1 Reductionist approaches introduce some complex related findings 3
  1.1.2 The social psychology approach to group efficacy 4
    1.1.2.1 Task Characteristics 4
    1.1.2.2 Individual Characteristics 6
    1.1.2.3 Group Characteristics 6
    1.1.2.4 Combining tasking, individual and group characteristics 6

1.2 Organisation of the Dissertation 7

1.3 Candidate Profile 8

2 Literature Review 9

2.1 Group Norms - Overview 9
  2.1.1 Groups: Types, Definition and Norms 9
  2.1.2 Group Dynamics: An Interactionist Approach 10
  2.1.3 Emergent Norms: Formation and Manipulation 11
    2.1.3.1 Group Influence 11
    2.1.3.2 Group Identity 11
    2.1.3.3 Events and Precedence 12

2.1.4 Norm Manipulation 12
  2.1.4.1 Norm Manipulation by Precedence 12
  2.1.4.2 Norm Manipulation by Influence 12

2.2 Norms in Software Literature 13

2.2.1 Personality & Communication Research 13
  2.2.1.1 Personalities and Group Performance 13
  2.2.1.2 Communication and Group Performance 15

2.2.2 Pair Programming 15

2.2.3 Software Development Norms in Different Companies 15
## Contents

2.2.4 Distributed Software Development .......................... 16
   2.2.4.1 Rapid Response & Interrupt Friendliness Norm .......... 17
   2.2.4.2 Availability of Information Norm ....................... 17
   2.2.4.3 Richness of Information Transference Norm ............. 17
   2.2.4.4 Embrace and Use of Appropriate Technology as a Tool
          Norm .......................................................... 18
   2.2.4.5 Role Clarity Norm ........................................... 18
   2.2.4.6 Cooperative Behaviour Norm ............................. 18
   2.2.4.7 Recommendations for Success, with a View of Norms ... 19
   2.2.4.8 The Relavence of Norms .................................. 19

2.2.5 Intercultural Research ........................................ 19

2.3 Norms in Social Psychology Literature ....................... 20
   2.3.1 Production Blocking ........................................ 20
   2.3.2 Social Loafing .............................................. 20
   2.3.3 Evaluation Apprehension ................................... 21
   2.3.4 Groupthink .................................................. 21

### 3 Study 1: Applicability

3.1 The Assembly Bonus Effect .................................... 23
   3.1.1 Letters-to-Numbers problem ............................... 24
      3.1.1.1 Task characteristics of the Letters-to-Numbers problem 26
      3.1.1.2 Analogy of Letters-to-Numbers problem to programming 26

3.2 Hypotheses and effects ....................................... 27

3.3 Participant Characteristics ................................... 27
   3.3.1 Raven’s Progressive Metrics ............................. 28

3.4 Group Characteristics ......................................... 28
   3.4.1 The Group Environment Questionnaire .................. 28

3.5 Goals .......................................................... 30

3.6 Method ........................................................ 31
   3.6.1 Participants ............................................... 31
   3.6.2 Materials ................................................... 31
      3.6.2.1 The modified Group Environment Questionnaire .......... 32
      3.6.2.2 The Letters-to-Numbers problem: A computer imple-
              mentation ................................................ 32

3.7 Procedure ...................................................... 34

3.8 Results ........................................................ 38
   3.8.1 Individual Characteristics .............................. 38
   3.8.2 Group Characteristics ................................... 40
3.9 The Letters-to-Numbers Performance: Triads VS Individuals ......... 40
3.10 Emergent Group Behaviour ........................................... 42
  3.10.1 Stepwise Regression: First Attempt .......................... 44
  3.10.2 Stepwise Regression: Second Attempt ....................... 44
  3.10.3 Stepwise Regression: Overall .................................. 45
3.11 Insights from Analysis .............................................. 46
  3.11.1 Performance in attempt 1 is related to measured communication 46
  3.11.2 Performance in attempt 2 is related to high visibility and directed communication ............... 47
  3.11.3 Overall performance in the letters-to-numbers problem is driven by high visibility and highly directed communication ...................... 48
3.12 Limitations and Threats to Validity .................................. 48
3.13 Discussion .............................................................. 49
  3.13.1 Known Issues with the Assembly Bonus Effect ................. 49
  3.13.2 Application of Results ........................................... 50
    3.13.2.1 Investigating other tasks ................................... 51
3.14 Related Work ......................................................... 51
3.15 Conclusion ............................................................ 53

4 Study 2: Effectiveness ................................................. 55
  4.1 Norms and Information Sharing background ....................... 55
    4.1.1 Critical and Consensus Norm ................................ 56
  4.2 Requirements Elicitation ............................................ 56
    4.2.1 Combining Norms and Requirements Elicitation ............... 57
  4.3 Hypothesis and effects .............................................. 57
  4.4 Task Characteristics ................................................. 58
    4.4.1 LIBSYS library system ......................................... 58
      4.4.1.1 Task characteristics of the requirements elicitation scenario ........................................ 60
  4.5 Participant Characteristics ......................................... 60
  4.6 Group Characteristics ................................................. 60
    4.6.1 The Critical/Consensus manipulation check .................... 60
      4.6.1.1 Critical/Consensus Priming Activity .................. 61
  4.7 Method .............................................................. 62
  4.8 Participants ........................................................ 62
  4.9 Materials ............................................................ 62
    4.9.1 The modified Group Environment Questionnaire ............... 63
  4.10 Procedure ........................................................... 63
4.11 Results ................................................................. 67
  4.11.0.1 Bootstrap Resampling Confidence Intervals .......... 67
  4.11.1 Individual Characteristics ............................... 69
  4.11.2 Group Characteristics .................................... 69
  4.11.3 Task Results .................................................. 70
    4.11.3.1 Resampling Results - Shared Requirements ......... 73
    4.11.3.2 Resampling Results - Unshared Requirements ...... 73
    4.11.3.3 Resampling Results - Total Requirements .......... 75
    4.11.3.4 Resampling Results - Wrong Requirements .......... 75

4.12 Insights from Analysis .................................... 76
  4.12.1 Critical and Consensus Norm Manipulation Do Not Affect Significantly the Cohesiveness of Software Groups .............. 76
  4.12.2 Normative Manipulation Check Metrics .................. 77
  4.12.3 Cohesiveness, GEQ and manipulation ..................... 78
  4.12.4 Critical and Consensus Manipulation: Manipulation Check Issues 78
  4.12.5 Triads primed with critical norms shared more common information 79
  4.12.6 Triads primed with critical norms shared more unique information 79
  4.12.7 Critical and Consensus primed triads did not have noticeably different error rates 79

4.13 Limitations and Threats to Validity ....................... 80
4.14 Discussion ..................................................... 81
4.15 Related Work .................................................. 82
4.16 Conclusion ................................................................ 82

5 Study 3: Adoptability ............................................. 85
5.1 Norm Manipulation Calibration ................................ 86
  5.1.1 Need for Closure ............................................. 88
  5.1.2 Individualistic/Collectivistic Measure ................. 88
5.2 Hypothesis and effects .......................................... 88
5.3 Task Characteristics ............................................. 89
5.4 Participant Characteristics ..................................... 89
5.5 Group Characteristics ............................................ 89
5.6 Method: Group Profile .......................................... 89
  5.6.1 Group Profile: Participants ................................ 89
  5.6.2 Group Profile: Materials ...................................... 90
    5.6.2.1 Modifying the Need for Closure Scale for the Experiment 90
    5.6.2.2 Modifying the Individualistic/Collectivistic Measure for the Experiment ........................................... 91
5.16.1.4 Response differences between Maintainers and Implementers - A Discussion .......................... 118

5.17 Related Work ........................................... 120

5.18 Conclusion ................................................. 121

6 Synthesised Analysis ........................................ 123

6.1 Applicability .............................................. 123

6.2 Effectiveness .............................................. 123

6.3 Adoptability .............................................. 124

6.4 Summary .................................................. 124

6.4.1 Proof of Generalizability of Social Psychology research .......................... 124

6.4.2 Applicability of Normative Manipulation to Improve Task Performance .......... 125

6.4.3 Normative Manipulation is Non-Intrusive ................................................ 125

7 Conclusion .................................................. 127

7.1 Contributions ............................................. 128

7.2 Future Work ............................................... 129

7.2.1 Harnessing the Assembly Bonus Effect .................................................. 129

7.2.2 Normative Manipulation for other task types ........................................... 129

7.2.3 Normative Identification and Manipulation Framework .......................... 129

7.2.4 Further adoption of Social Psychology Literature ..................................... 130

7.3 Human Ethics Approvals .................................... 130

A Original Group Environment Questionnaire .................. 131

B Raven's Advanced Progressive Metrics & Letters-to-Numbers Experimenter script .... 135

C Critical and Consensus Normative Manipulation Script ......................... 141

D Experiment 3: Introductory Survey Example ............................................. 145

E Experiment 3: Task, Responsibility and Interactions Responses .................. 161

F Experiment 3: Task assignment, Decisions and Approval ........................... 167

G Experiment 3: Priming and Individual Brainstorming Exercise .................. 169

H Experiment 3: Group Session Script ................................................ 177
<table>
<thead>
<tr>
<th>Contents</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>I    Experiment 3: Sample Exit Questionnaire</td>
<td>181</td>
</tr>
<tr>
<td>Bibliography</td>
<td>189</td>
</tr>
</tbody>
</table>
Introduction

“The major problems of our work are not so much technological as sociological in nature” - T. DeMarco & T. Lister [1]

There is a growing body of work that focuses on sociological issues as an important factor in software development. Barry Boehm’s work in Software Engineering Economics[2], assigns a significantly higher cost factor to Personnel/Team capability than any other factor identified across a sizeable sample of industry projects. Scacchi [3] points out that “Software Engineering and Project Management are performed within a dense web of technological, economic and social arrangements... [and] social analysis emerges as one of the most revealing approaches to software engineering and project management.” Agile software development, a method for software development that is gaining much traction in the development community, has been promoted as “for the people” [4]. Agile software development also notes that “the most important implication to managers working in the agile manner is that it places more emphasis on people factors in the project: amicability, talent, skill and communication” [4]. Further, Freudenberg & Sharp’s [5] Top 10 Burning Research Questions for Practitioners lists “Sociological studies - what were the personalities in successful/failed agile teams” as an item that is of great interest in the agile development community.

The growing emphasis on sociological issues demonstrate that how people work together is an important aspect of productivity in software development. As projects grow larger, coordination and cooperation issues will become even more common.

Several attempts have been made within the field of Software Engineering to address these underlying sociological issues, most of which stem from a traditional reductionistic approach 1 (see Section 1.1.1). These techniques consist of modelling group efficacy as a function of a small number of distinct group properties, but present several issues.

Social Psychology uses an alternative approach to examining group efficacy which

---

1A reductionistic approach reduces the forces that affect a model to a limited number for study, in order to obtain a relationship between cause and effect. Some models have attempted to relate group efficacy to simply group motivation, for example.
addresses the issues faced by traditional reductionistic approaches (see Section 1.1.2). Group efficacy is viewed as result of the non-decomposable, interacting components of task, individual and group characteristics.

The thesis of this research is threefold: Firstly, Software Engineering group behaviour can be predicted by Social Psychology theory (applicability). Secondly, the performance of Software Engineers in a particular Software Engineering task can be improved by normative manipulation (effectiveness), and finally, the process of normative manipulation can be performed in an unobtrusive manner (adoptability).

To establish the applicability of Social Psychology literature on Software Engineering group norms, a Social Psychology experiment was repeated on Software Engineers, and results corroborative to the original study and Social Psychology literature were obtained. The conducted experiment compares the performance of groups against individuals in a specific, defined task, and show that for that particular task, groups perform better than individuals. Further, we were able to show that groups that adopted behaviour that was consistent with Social Psychology theory for productivity gains did perform better than groups did not.

To examine the effectiveness of normative manipulation, a study was conducted in which a set of Software Engineering groups were manipulated in accordance to Social Psychology literature. Depending on the type of manipulation applied, the performance of the Software Engineering groups in a Software Engineering type task was affected. Normative manipulation is used to affect the behaviour of group members by making them adopt and value particular behaviour amongst themselves over other types of behaviour. If the valued behaviour is conducive to the task being performed, this manipulation would then serve to increase group productivity.

Finally, to examine the adoptability of the normative manipulation technique, this technique was carried out in an industry group to measure its intrusiveness. Normative manipulation was performed on Software Engineers in an active group in order to increase the amount of information shared in a requirements elicitation task. This third experiment examines the intrusiveness of the normative manipulation technique when applied to an active group.

### 1.1 Creating effective software teams

This section discusses the problems encountered by the the current approaches to software engineering group issues, and compares them to the interactionist approach. It goes through each of the factors that underlie the interactionist approach to group study in turn, beginning with Task Characteristics, followed by Individual Characteristics and finally Group Characteristics.
Conflict can affect the performance of software teams significantly. Sawyer’s [6] survey of 40 software development teams found that “a combination of the team’s characteristics, team member characteristics and existing levels of intragroup conflict accounts for nearly one-half of the variance between the best and the worst performing teams”. Indeed, Lakhanpal’s [7] study on 31 software development groups found that between the factors of cohesiveness, capability and experience, cohesiveness had the strongest and most significant effect on group performance. A group that works well together, therefore, is a precursor to creating effective groups.

1.1.1 Reductionist approaches introduce some complex related findings

Effective groups cannot be simply defined by the power of a single trait. Rather, it is a comprehensive and interconnected set of traits that come together to produce a single productive group. We begin this example by taking Lakhanpal’s [7] observation that cohesiveness is the more significant predictor of group performance.

However, creating an effective software team is not all about simply creating a cohesive group. Stogdill [8] and Langfred [9] found that cohesiveness did not reliably predict group performance. Indeed, in Stogdill’s work, it was found that although a third of the highly cohesive groups performed better completing tasks, in another third, highly cohesive groups in fact performed worse. Strangor [10] explains this effect: cohesion causes a group to exhibit more strongly actions that are identified as being centrally related to the group’s identity. If the group’s identity is to work hard and be productive, then cohesion will increase performance. However, if the group’s identity is to goof of, then that will happen instead!

Amount of communication is also not a simple predictor of success. An agreeable environment for communication can also be detrimental to group productivity. In an extreme case of this, a group might affirm each other’s stance on a particular issue, to the point of excluding new information that would lead to the correct solution [11] (see Section 2.3.4 on Groupthink). In this case, within the group, there still is communication among members, but only stale information is considered. For instance, in the Challenger disaster, although there were dissenting opinions on the safety of the launch (the fact that the weather was too cold for safe operation), an agreement was reached to continue with the launch [10].

Given the complexities of cohesion, ability and communication, creating effective software teams, then, is not simply limited to maximising the variables of “amicability, talent, skill and communication” [4]. Bell’s [12] 2007 meta-analysis on the variables that affected team performance resulted in a more comprehensive list, including, “minimum
agreeableness and team mean conscientiousness, openness to experience, collectivism and preference for teamwork". These variables actually can be viewed as co-dependent or co-regulating: For instance, although only a minimum amount of agreeableness between the team is a predictor of group efficacy, the group still has to have a strong sense of collectivism and preference for teamwork. Similarly, when performing tasks, the team must be conscientious in general, but still be open to attempting new ways of solving problems. Faced with the complexities inherent in this reductionistic approach (breaking down the variables that make a successful group), this work attempts to demonstrate a concept (adapted from social psychology research) that could take into account this complexity and yet yield a simple and useful way to create effective groups.

1.1.2 The social psychology approach to group efficacy

In contrast to the reductionist approach taken by software researchers, group efficacy is taken by social psychologists to consist of three broad interacting components: task characteristics, individual characteristics and group characteristics [10; 13]. These components affect each other and moderate the effectiveness of the group as a whole. The nature of a task can limit or enhance how multiple individuals contributing will affect the final output. In an army route march, for instance, the platoon can only move as fast as its slowest member. If the fastest member was replaced with another, even faster member, the performance of the platoon would not change significantly.

However, if the slowest member was replaced with a faster member, the change to the group would be more significant. This is especially so if the second slowest member is also significantly faster than the member being replaced. In this example, increasing particular individual characteristics of the group can increase the efficacy of the group, but only if the task characteristics are suitable.

1.1.2.1 Task Characteristics

Tasks have characteristics that affect how efficiently they can be completed. These characteristics are not exclusive, although by definition some will exclude each other. For example, a task cannot be unitary and divisible at the same time, but it can be unitary and judgemental at the same time (e.g. jury deliberation). Table 1.1 lists various task types and examples.

The characteristics of the task may dictate different group modifications for increased productivity. A task characterized as disjunctive, for instance, would show greater improvement if the strongest member of the group is replaced by an even stronger member, regardless of the ability of the other members. Similarly, for an additive task, raising only the strongest or weakest member ability may not improve
Table 1.1: Task categories and examples. Reproduced from [10]

<table>
<thead>
<tr>
<th>Task type</th>
<th>Description</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unitary Task</td>
<td>A unitary task cannot be further divided up among individuals and must be attempted in its entirety</td>
<td>A capella performance</td>
</tr>
<tr>
<td>Divisible Task</td>
<td>A divisible task can be further partitioned up into smaller distinct tasks and attempted piecemeal among these individuals</td>
<td>Organising a party</td>
</tr>
<tr>
<td>Judgemental Task</td>
<td>A judgemental task involves the ability of the group to make judgements that cannot be evaluated objectively because there are no clear criteria to judge them against</td>
<td>Jury Trial</td>
</tr>
<tr>
<td>Intellective Task</td>
<td>An intellective task involves the ability of the group to make a decision or judgement which can subsequently be evaluated objectively for correctness</td>
<td>Mathematical proof</td>
</tr>
<tr>
<td>Additive Task</td>
<td>An additive task is where the outcome is the direct sum of output of the individual group members</td>
<td>Relay race</td>
</tr>
<tr>
<td>Disjunctive Task</td>
<td>A disjunctive task is where the output is dependant on the output of the strongest member of the group</td>
<td>Quiz night</td>
</tr>
<tr>
<td>Conjunctive Task</td>
<td>A conjunctive task is where the outcome is dependant on the output of the weakest member of the group</td>
<td>Army march</td>
</tr>
<tr>
<td>Eureka Task</td>
<td>A eureka task is a task in which the solution is non-obvious, but once the ideal solution has been found, the task would be viewed as easy to do</td>
<td>Discovering $D = M/V$</td>
</tr>
</tbody>
</table>
the performance of the group as much as improving the ability of every member in the group slightly.

1.1.2.2 Individual Characteristics

Individual characteristics in the context of group efficacy are variables that drive the actions of an individual, such as personality (or personal bias), skill/ability and experience. Software Engineering has long recognised the importance of the ability and experience capabilities of group members. Boehm’s Software Engineering Economics [2] calls for a capability measure which heavily affects the probability of success of a project. This measure takes into account the skill and experience a member would contribute to the completion of a particular task.

In addition to simply having the ability to complete a task, however, the personality, or biases of an individual towards certain behaviour and beliefs can affect how well a group works together. These interactions and their emergent properties are known as group characteristics.

1.1.2.3 Group Characteristics

When individuals work together, emergent characteristics will form, and in turn, can be identified. For instance, how well a group gets along is a measure of the group’s cohesiveness. Another example is how well a group responds to information that is contrary to group belief. A critical group may debate and change its position in the light of new information, whereas a highly conformist group may outright reject all contrary information in order to preserve its position on a particular view. In the case of decision making tasks, this emergent behaviour of rejecting contrary information will impede the effectiveness of the group’s attempt at obtaining the correct solution [10].

1.1.2.4 Combining tasking, individual and group characteristics

The requirements for individual characteristics such as skill and experience in groups is understood enough that they can be used for predicting the success of a particular task [2]. Some work into tasking in software development has been made which has focused mainly on the technical or organisational requirements associated with these tasks. Chang & Christensen (PM-Net) and Lin & Yeh (Object Oriented Project Management) for instance, developed scheduling models that take into account information hierarchies in organisations [14; 15]. The Program Evaluation and Review Technique (PERT), Critical Path Method (CPM) and DesignNet on the other hand, take into account purely task ordering and skill requirement [16; 17; 18]. Even so, social psychology
literature does provide the means to classify tasks and also how they might impact the performance in a group (see Section 1.1.2.1).

Group characteristics (or group norms) concerning Software Engineering and how they affect group productivity however, is still relatively unknown. Social psychology research shows that the manipulation of group norms can affect the result of group performance in tasks. It may be possible that this research can be applicable to software engineering - given particular groups, it may be possible to manipulate the norms of a group in order to increase group effectiveness in particular types of tasks.

1.2 Organisation of the Dissertation

This dissertation is organised as follows: Chapter 2 covers a review of the literature of the field. This is then followed by a series of three experiments. Chapter 3 covers the first experiment that looks at the applicability of Social Psychology group norm theory/practice in Software Engineering groups. Chapter 4 covers the second experiment, that examines the effectiveness of manipulating norms in Software Engineering groups. Finally, Chapter 5 covers the adaptability of the normative manipulation technique in an industrial context.

The first two experiments are laid out identically. First, the chapters present the background of any measures or theory used in the experiments. Then, any hypotheses made for the experiments are presented. Keeping with the integrative approach (of treating task, individual and group characteristics equally), the task used for the experiment, individual and group characteristics that are to be measured are then presented. A method section then follows that show the participant demographic, the materials used for the experiment, and the procedure of the experiment. The results are then presented, followed by any insights gained from the analysis of results. Limitations and threats to validity are presented, followed by a discussion of the results. Every chapter contains a related work section that presents similar work undertaken by others. The chapters then finish with a conclusion that present the relevance of any findings to the Software Engineering field.

The third experiment maintains the same broad structure as the first two experiments. However, due to the nature of the third experiment as being comprised of two sequential parts, within the experiment are presented two separate method and results sections.

Each experiment includes a related work section that covers in detail the comparison of work that is similar to that undertaken in this thesis.

A synthesised analysis is presented in Chapter 6 that covers all three experiment and relate how each experiment contributes to the areas applicability, effectiveness and
adaptable. It also lays out a summary of the contributions of this research.

Finally, a conclusion is presented in Chapter 7, followed by an appendix of the experimental material.

1.3 Candidate Profile

The candidate is a full time PhD student at the Australian National University. The first two experiments described in this thesis were conducted by the candidate, although some assistance was obtained for distribution of material by third parties. All interactions described were heavily scripted and all assistants understand the need to conform strongly to the script to minimise bias and enhance repeatability.

For the last experiment, the candidate conducted the experiment with a full time PhD student at University of Victoria. Similarly, all interactions described were heavily scripted to minimise bias and enhance repeatability.
This section provides the reader with background for the area of research of this thesis. A related work section is provided in each experiment chapter that covers a more specific comparison to the work of others.

2.1 Group Norms - Overview

In this section we will explain the concept of group norms. We will then cover in detail how they are formed and how they can be manipulated. We conclude this section by examining software literature with an eye on normative constructs and recommendations.

2.1.1 Groups: Types, Definition and Norms

Groups can be classified into various types, shown in Table 2.1. With regard to this work, the types of groups that are of the most interest are reference groups and working groups. Reference groups are important because these groups shape the identity of individuals and guide the behaviours, beliefs and attitudes of people that identify with them. People that strongly identify with reference groups would be more likely to behave as expected of the prototypical member of the reference group [19]. Working groups are the focus of this thesis, specifically Software Engineering groups (e.g. development team) come together in order to accomplish some task or achieve some specific goal.

Groups consist of roles. The persons fulfilling each role can be interchanged, and each role describes to the particular place and function of the individual. Norms prescribe the behavioural and attitude expectations that are to be fulfilled by people in these roles [20]. People that fill roles have an expected conduct and set of behaviours (or normative expectations). For instance, software developers are expected to spend at least some of their time developing code, and not, as an example, surfing (This of course, can be untrue if the software developers in question are developing surfing
<table>
<thead>
<tr>
<th>Group</th>
<th>Definition</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Reference Group</td>
<td>A group of individuals that we look up to and identify with because we admire and want to be like those who belong to it</td>
<td>Friends, work colleagues, sports players</td>
</tr>
<tr>
<td>Working Group</td>
<td>A group consisting between 3 and 12 individuals who are actively attempting to meet a specific goal</td>
<td>A Jury, study group</td>
</tr>
<tr>
<td>Social category</td>
<td>A large and relatively permanent social group, such as people who share a gender, a religion, a nationality, or a physical disability</td>
<td>Men, women, Europeans, Americans, the elderly</td>
</tr>
<tr>
<td>Culture</td>
<td>A large social group made up of individuals who are normally in geographic proximity with each other and who share a common set of beliefs and values, such as language, religion and family practices</td>
<td>Western, Eastern</td>
</tr>
<tr>
<td>Crowd</td>
<td>A large number of individuals who come together in a common place for a common purpose</td>
<td>People shopping in a store, a riot or a mob</td>
</tr>
</tbody>
</table>

software!).

2.1.2 Group Dynamics: An Interactionist Approach

Group dynamics are based on the idea that group behaviour is the result of the reciprocal interaction between individual and group factors. This is an interactionist approach, which attempts to preserve the fidelity of both the components of a system under study (in this case, groups) and the interactions between them, without simplifying to a set of particular factors. The contrasting reductionistic models of the individual-level approach (where groups are studied exclusively from the point of view of the individual) or the group-level approach (where groups are studied as an entity and the individual make up largely ignored) is largely felt by social scientists to be incomplete [10] and will not be covered in this thesis.

A social (or group) norm is defined as a way of thinking, feeling, or behaving that is perceived by group members as appropriate [10]. Norms do not indicate the rightness or wrongness of the behaviour, merely the acceptability of the behaviour by fellow group members. For instance, it is accepted for Americans to consume hamburgers (which typically contain beef), whereas devout Hindus will never do so. In fact, even if a norm
is not based on any objective fact, it may still have an important validity for members and the group members will believe and behave within its boundaries because it seems appropriate to do so [21].

2.1.3 Emergent Norms: Formation and Manipulation

Group norms can be established spontaneously, either implicitly or explicitly. This subsection covers some ways norms are established, and consequently, how they can be manipulated.

2.1.3.1 Group Influence

In situations where there is a lack of information about an appropriate response, people tend to influence each other, either implicitly or explicitly, until they hold a similar attitude or position. An example of implicit influence is found in an experiment conducted by Sherif [22], which utilized the autokinetic effect. The autokinetic is an illusionary effect that occurs when a single stationary point of light appears to be moving in a dark room. In Sherif’s study, when participants were first quizzed alone, the responses on how much the light moved tended to vary. However, the responses on how much the light moved tended to conform when the participants were quizzed together, and these responses tended to persist even after the groups were split up.

Group influence can also explicitly affect the response of an individual within the group regardless of appropriateness. For instance, in an experiment conducted by Asch [23], when confederates to the experimenter gave deliberately wrong responses, sometimes the participants changed their responses to suit.

2.1.3.2 Group Identity

Individuals that identify strongly with a group tend to give responses that conform with the referenced group. In an experiment conducted by Abrams et. al. [24], if participants were told that they were radically different from a group giving a response, they tended not to give responses that matched up with the reference group, whereas if they were identified as being a part of the reference group, their responses tended to conform.

When groups are being formed, the salience of group identity will affect how strongly a person will adopt attitudes (or norms) conforming to that of the group. Even when a group is formed and the norms established, the level of salience of the group identity serves to show how likely the group member will act within the norms of the group. A nationalistic group, for instance, would be far more likely to perform actions that are associated with their national identity when the salience of said identity is increased (For example, Australians are more likely to hold a barbecue on Australia Day!)
2.1.3.3 Events and Precedence

Group norms can also be established via first exposure [19]. This happens when an event occurs a first time in a particular way, setting expectations on the format and content of future similar events. For instance, the order in which individuals report progress in meetings, or even the seating arrangement, can be established and upheld in future meetings.

Carry over behaviours can contribute to the norms of a group. Science teachers may establish a set pattern of lab work, lectures and tutorial through a week for their classes as was exposed to them when they were students themselves. They might adopt the norms of the appropriate level of duty and behaviour expected of students in class, and also of their own behaviours towards students.

Powerful group members (such as leaders) can also explicitly establish norms. For instance, a leader may explicitly set a rule on the number and length of the tea/coffee breaks allowed to group members throughout the work day, or even the mode the of address between members.

Critical events may also establish norms within groups. For instance, a group previously happy to post bill clients in the past may implement a norm of down payment before work is attempted for a new client after encountering issues with one particular client.

2.1.4 Norm Manipulation

Given the way norms are established, it follows then, that the manipulation of group norms, such as the implementation of a norm that is counter to what exists, can be accomplished largely either by precedence or by influence.

2.1.4.1 Norm Manipulation by Precedence

Norms can be manipulated by introducing a task that sets the precedence of operation by the group for subsequent tasks. For example, Postmes [25] successfully manipulated groups to exhibit a critical thinking norm by introducing a debating task to a group prior to a second task. He showed that groups would continue to operate in the critical thinking mode even for the second task.

2.1.4.2 Norm Manipulation by Influence

Norms can also be manipulated via influence, either externally or internally. Internally, if group members begin to behave in a certain manner, other group members, if their sense of attachment to the group is raised, may begin to adopt this behaviour
themselves. Group members may also be influenced internally by powerful group members that make explicit statements. In Sherif's [22] experiment, group members were more likely to agree how much the point of light had moved if there was more internal agreement.

Externally, norms can be manipulated by raising the salience of group identity so that group members are then more likely to exhibit the behaviour expected of the prototypical member of the group. In Asch’s [23] study, a group that had been manipulated to feel closer (i.e. raising of the salience of group identity) was more likely to have individuals that changed responses to match the responses of planted confederates, who were in fact wrong.

2.2 Norms in Software Literature

In this section we review Software Engineering literature with an eye to group norms. We identify and collate normative recommendations and observations present in various works.

2.2.1 Personality & Communication Research

Research in Personality & Communication largely differ from the Normative approach fundamentally by utilizing the reductionist approach to group performance as opposed to the interactionist approach. Reductionist approaches break down a system into components, which are then studied in isolation by assuming some consistency within the items not being studied, whereas the interactionist approach attempts to take into account the effects (or emergent properties) that may arise from the system as a whole.

In this section we discuss two typical ways of breaking down the components of group performance - personalities and group performance [26; 27; 28; 29; 30; 31; 32], and communication [33; 34; 35].

2.2.1.1 Personalities and Group Performance

A large amount of research has been conducted into the relationship between personalities and effective teams [26; 27; 28; 29; 30; 31]. Some researchers [36; 31] argue that personality can be an excellent predictor in job performance. Gorla & Lam [32] go further in their work and identify some desirable personalities and mixes that may lead to better software teams.

In spite of the common belief that personalities may effectively predict performance, the results within the software field seem to be inconclusive. McDonald & Edwards [26] argue that the inconclusiveness of results may be due to the inappropriate application
of personality factors. More importantly, some results simply do not agree - Darcy & Ma's [37], Carpentz's [28] and Karn et. al's [27] studies all failed to find strong links between personality and performance.

Within the field of psychology itself, the link between personality and performance has also been debated [38]. Wicker [39] found that personal attitudes did not necessarily affect directly the behaviours of individuals, and the lack of strong empirical relations between personality and behaviour has even led some researchers to question the validity of the argument that personality can predict behaviour. The theory of planned behaviour [40] was developed to explain the inconsistency of behaviour based on personality. Ajzen [40] notes that personality and other individual factors only serve to predict what action an individual will probably take - the actual behaviour itself is determined by a combination of interleaved factors such as the normative expectations of both themselves and other important referent individuals or groups, in addition to their personal bias (personality/attitude) and raw ability to perform accordingly to their intentions. The ability of referent people to affect the actions of others is highly relevant in a group context, where group norms impose expectations on behaviour of individuals. Therefore, it follows that personality is simply one aspect of predicting the actions of individuals, and consequently only one aspect in predicting how effectively they would work together in a team. In order to study how individuals work together, it is important to also study how norms affect behavior.

In spite of the contest of the link between actions and personality, we can still obtain some normative recommendations within personality study literature. Karn et. al. [27] notes that in their study “...teams without disruption but who lacked debate and suffered from inadequate communication encountered serious problems, failing to meet project milestones...”, and conclude that groups that experienced no disruption experienced negative consequences such as “limiting discussion to only a few alternatives; or initial solutions never being reconsidered; or alternatives either not being proposed or being ruled out by the majority of the team, resulting in further lack of debate on future issues.”. This fits in with the results of establishing a conformist norm: Postmes et. al. [25] manipulated groups by exposing them to either a congenial task or a debating(disruptive) task to either exhibit conformist or critical norms respectively. They showed that the conformist groups on the whole exhibited much the same behaviour as those observed by Karn et. al. [27]. Gorla & Lam [32] recommend that “in terms of the social-interaction dimension, the preferred personality for programmers on small teams is extroverted...”. Extroversion can be taken to refer to out-going people that are comfortable with expressing opinions with each other - therefore leading to the conclusion that a recommendation for the norm of a high level of communication is desired in smaller development teams.
2.2.1.2 Communication and Group Performance

Communication has long been identified as a strong moderator of group performance [33; 34]. Communication in software groups includes coordination, and also knowledge management [35]. Knowledge Management is a relatively new field that includes the acquiring of technological knowledge, the access to domain knowledge, sharing of knowledge, policies and practices and also the identification of the sources of knowledge.

High levels of communication are often stated as desirable for productivity. As a group grows larger, however, the number of communication pathways increases exponentially [33]. This increase can be shown mathematically [41] and empirically [42] and results in the problem in that beyond a certain size, communication and coordination would dominate the workload of the individuals involved and no benefits can be obtained from the addition of further staff. In terms of software engineering, the works [33; 34; 33] can be taken to show that a norm of effective, relevant communication should be strongly instilled in software groups in order to drive productivity and effectiveness.

2.2.2 Pair Programming

Pair programming is defined as an activity in which "...two programmers jointly produce one artefact (design, algorithm, code)...." [43]. Studies on pair programming have shown that pairs may perform more effectively [44; 45]. Follow on studies have been conducted to find compatible pairs [46; 47; 29] in order to harness the efficacy of pairs more easily.

Pair programming, however, focuses on groups of two (dyads). Although the recommendations of the literature can be applied to other dyad work pairs, it may not be appropriate to adopt the same recommendations within larger groups. Hackman & Vidmar [48] for instance, find that it is impossible to extrapolate effects from larger groups down to groups that are smaller than triads, or from dyads up to larger groups. They conclude that dyads are a special type of group due to the ability of members to respond immediately to their counterparts. This is in contrast to larger groups which tend to increase competition on individual members for communication, teamwork and coordination. In the context of this thesis then, the normative recommendations for pairs will not be covered in detail.

2.2.3 Software Development Norms in Different Companies

Perlow [49] studied the work norms for three companies located in India, China and Hungary respectively. The differences in time commitment and operating hours were observed. For instance, the Chinese engineers were observed to work almost strictly within office hours, and overtime was only requested if determined to be absolutely
necessary. In contrast, the Indian engineers were observed to spend long hours at work, even on Saturdays. One engineer notes that there is great pressure to spend all Saturday at the office, even if there isn’t any work to do. The Hungarian engineers kept flexible hours that changed to meet the demands of work. These norms were observed to reflect other aspects of work such as vacation time, and lunch time arrangements.

Perlow also studied co-ordination norms within these different companies. These norms seemed to continue to be reflected through how the engineers worked together. For example, it was observed that the Chinese engineers were highly formal and work was tightly cantered around the engineers and their immediate superiors. Individual engineers had little knowledge of what went on outside of their own work area. The Indian engineers were observed to exhibit high interdependency with peers, with team members highly valuing each other’s input. The Hungarian engineers were observed to be very flexible with work, and team members were observed to be able to substitute easily for each other when the need arose.

Perlow’s work showed that even within a single domain of Software Engineering, vastly different norms may exist and affect multiple areas, from work commitment to group co-ordination.

### 2.2.4 Distributed Software Development

Distributed software development refers to the practice of having geographically distributed software teams that work concurrently and collaboratively on a single project. Olson & Olson [50] published a review of 10 years of investigations on the problems affecting distributed software development, and a collection of solutions for some of the problems highlighted. They began by observing effective co-located and distributed teams and identified the following norms that contributed to team success:

- Rapid response & interruption friendliness
- Availability of information
- Richness of information transference
- Embrace and use of appropriate technology as a tool
- Role clarity
- Cooperative behavior

Olson & Olsen then proceed to recommend some norms that were aimed at creating effective distributed software teams. These norms mainly attempt to bring the effective co-located norms into distributed software teams.
2.2.4.1 Rapid Response & Interrupt Friendliness Norm

Olson & Olson observed that in co-located teams, "if a team member wants to observe his manager's reaction to a problem, he [sic] can just glance quickly in her [sic] direction". Groups were observed to rapidly form and reform subgroups as different tasks were attempted. They also note in a survey of groups moving to an open-style environment that allowed dynamic interruptions that, although there is some initial resentment to interruptions, the group members ended up with the opinion that the advantages of being accessible far outweigh the disadvantages, which they felt could be worked around. These norms emphasize the importance of immediacy and accessibility of communication between group members.

2.2.4.2 Availability of Information Norm

Co-located teams studied by Olson & Olson were noted to have information widely and easily available to group members. An automotive design group studied was noted to "display all parts of a competitor’s car on the hall walls to serve as reference points". These norms emphasize the importance of the ease of information availability and serve as a convenient point of information sharing between group members.

2.2.4.3 Richness of Information Transference Norm

Group members working together were observed to use both explicit and implicit methods of communication. In addition to the standard way of communicating such as speaking and writing, group members were observed to derive information from body language & gestures, facial expressions, state of being and the environment. For instance, Olson and Olson observed that in one group, a group member referred to a particular idea explained by someone else by gesturing to the location in the room where the explanation had previously taken place. For example, Olson & Olson observed that a teleconference call could have been more amicable if the remote parties had somehow been able to observe that the teleconference partner's city had been in a snowstorm. This norm emphasizes the importance of the availability of peripheral information that may aid productivity and effectiveness. In the case of the snowstorm, the non-affected party could have instead spent the time performing some other task. In the case of body language and gestures, it allows people to establish efficiently ideas and concepts that are being discussed.
2.2.4.4 Embrace and Use of Appropriate Technology as a Tool Norm

Olson & Olson note that a counter-productive norm that exists is using technology for the sake of using it. They observed that “at the large automobile company, some of the remote participants used video conferencing, not because they personally believed it would help them communicate but because they wished to be seen using it by the higher level managers who invested in it”. This resulted in meetings being cut short by follow-on groups. Therefore it follows that technology should be used appropriately and be fit for purpose. For example, they note that the automobile company had begun to use email attachments to view pictures of cars at remote locations which was an effective way of showing defects and manufacturing problems to remote engineers.

2.2.4.5 Role Clarity Norm

It was noted by Olson & Olson that if group members were well versed in their area of responsibility or had a good level of understanding among themselves, they would function well regardless of being distributed or co-located. They observed that this clarity served well to negate the problems introduced by what they termed as tightly coupled work. Tightly coupled work was defined to be work that required close collaboration, consisting of “rapid back and forth in conversation or awareness and repair of ambiguity”. In essence, tightly coupled work requires group members to actively work together to continuously and actively eliminate ambiguity. Therefore, this norm requires role clarity the group members involved should be able to clearly pick out the relevant expertise and responsibilities involved and deconstruct tasks in order to contribute effectively towards the common goal.

2.2.4.6 Cooperative Behaviour Norm

Group members compensated according to how competitively they were against the performance of other group members were less likely to work well together, or adopt norms such as information sharing. In fact, in a competitive environment, it was reported that “consultants [in a particular company] even reported avoiding learning Lotus Notes (a collaboration tool)... There was no incentive to share one’s best ideas if they were going to be seen as common, no longer unique.” In contrast, Olson & Olson also observed that, in organizations or communities (such as space physics) where collaboration had a strong tradition, group members were far more ready and willing to adopt new technologies or techniques to continue collaboration. Therefore, a cooperative behaviour norm directly influences the willingness of group members to share information, and embrace technology that allows information to be shared.
2.2.4.7 Recommendations for Success, with a View of Norms

After Olson & Olson concluded their observations on productive and counter-productive norms adopted by groups, a list of recommendations were produced. In addition to attempting to implement the identified productive norms in distance teams, they also recommend that distributed groups should have rotational members that are on site with the remote group so that they can "know the local people and their roles, [and] translate various behaviours so that they will be less likely to be misinterpreted".

In the context of group norms, this rotation is equivalent to the insertion of members into the remote group and having them exposed to the remote group norms by influence. The rotated engineers maintain the salience of identity to his original sub-group, while simultaneously adopts norms to conform and perform within the norms of the remote sub-group. Therefore, a rotated engineer would respond appropriately within the norms of both subgroups as the salience of the sub-group is raised i.e. while dealing with the remote group, he or she exhibits site and group specific behaviour, whereas when reporting to his home group he returns to the original set of behaviours.

2.2.4.8 The Relavence of Norms

Olson & Olsons work showed that it was possible to characterize the various positive practices observed into particular group norms and to explain the recommendations for improvement in terms of normative theory.

2.2.5 Intercultural Research

Much work has been done to determine how best to get culturally different people to work effectively together [51; 52; 53; 54]. For instance, Diamant et. al. [51] studied the difference in effect on performance when American and Chinese students had to collaborate in either racially homogeneous or heterogeneous dyads using computer technologies as a medium. They found that the culturally diverse groups performed worse than homogeneous groups in their non-creative task. Diamant et. al. also point out that this was consistent with prior research by Massy et. al. [55], which showed that unless it was a highly creative task, culturally diverse groups tended to perform worse than culturally homogeneous groups.

The high success rate of diverse groups is consistent and predicted by social norm theory. Social norm research has shown that highly conformist groups would be less likely to share unique or hidden information [25], and therefore more likely to exhibit the phenomenon of groupthink [11]. The groupthink phenomenon leads a group to fixate on a particular solution and disregard any information that may lead to refinement or improvement. In sum, a more diverse group would be more likely to consider more
aspects of the problem, which would lead to the proposal of complete or creative solutions that would take into account all of the shared information.

2.3 Norms in Social Psychology Literature

Social Psychology has identified some group norms as being counter productive in group environments. These group dynamics restrict the efficiency of sharing information, or the levels of contributions by some group members. Some of these norms will be covered here, including

- Production Blocking
- Social Loafing
- Evaluation Apprehension
- Groupthink

2.3.1 Production Blocking

Production blocking is the result of the properties of the medium of communication when groups are together. Typically, in a group, only one member can use a particular medium for communication at a time. For instance, when someone is talking, the other members of the group have to listen. Members of a group in such a situation cannot make optimal use of their time because they either forget their ideas while waiting for their turn to speak, or they do not pay attention to other ideas as they think of their own, or even because they must spend some effort in order to be heard by other members of the group.

Production blocking can be counteracted by instigating approaches such as the nominal group technique. In this technique, individuals are kept apart for the initial ideas phase of a discussion and then finally brought together for discussion and to build on the ideas of fellow group members.

2.3.2 Social Loafing

Social loafing is a phenomenon in which group members self-regulate their own contributions because they perceive the group norm of productivity as lower than it should be. This may be because the group members do not wish to contribute disproportionately as other members coast along on their hard work. Social loafing has been reliably demonstrated in groups where individual contributions are hard to attribute.

Social loafing is shown to be reduced significantly if the visibility and accountability of group member participation is increased.
2.3.3 Evaluation Apprehension

Evaluation apprehension occurs when some group members are unwilling to state some of their own ideas because they are apprehensive about being negatively judged by other group members.

The problem of evaluation apprehension can be addressed by getting group members to believe that they are more expert than they think they are. It has been shown that this belief causes individuals to increase their contributions accordingly.

2.3.4 Groupthink

Groupthink is a phenomenon where a group adopts a strong mindset against external influences, rejecting any input that proves contrary to their views. In extreme cases, the in-group systematically excludes members that criticize their beliefs or assumptions. This results in an ‘information vacuum’ where new input contrary to that which is accepted is rejected.

Groupthink was identified by Janis [11] in a set of four historical situations that included firstly, the decision in 1941 by Admiral Kimmel to ignore the defence of Pearl Harbour in spite of warnings of an attack by the Japanese, secondly, the decision by President Truman to intensify the Korean war by crossing troops into North Korea, thirdly, the decision by President Kennedy to invade Cuba at the Bay of Pigs, and finally, the sequence of decisions by President Johnson to escalate the Vietnam war.

Groupthink has been found to be activated by directive leaders. Directive leaders tend to explicitly assign jobs to subordinates and explicitly define how the jobs should be completed. Flowers [56] and Richardson [57] found that groups with directive leaders reported more self-censorship and more mind-guarding, and indeed tended to exhibit more groupthink symptoms than other groups. Groupthink has also been found to be activated in highly cohesive groups with poor decision making processes [58; 59], or even simply groups that required unanimous rather than majority decision making [60].

Groupthink can be counteracted by installing a critical thinking norm, in which group members are encouraged to be less accepting of other’s views [61]. Groups that exhibit such norms will have symptoms such as, firstly, encouragement of divergent viewpoints, secondly, the open expression of ideas, thirdly, the awareness of limitations and threats to ideas and the group itself, and finally, the active discussion of collective doubts.
Study 1: Applicability

The first part of this thesis is that Social Psychology theory and practice can be applicable in Software Engineering. In this chapter, I describe an experiment that is conducted to show that, in spite of the lack of agreement as to the sort of person that is a typical Software Engineer (see Section 2.2.1.1), Software Engineering groups exhibit similar behaviour as predicted by general groups - therefore leading to the investigation of normative manipulation techniques in Software Engineering groups in order to improve group productivity.

Studies have been conducted on software engineers to profile how they would likely behave or even how they think [37; 31; 27; 26]. However, these studies have different conclusions and tend to show little common agreement (see Section 2.2.1.1). As a result, it can be difficult to apply results from other fields that require generalisability in terms of external validity. For instance, a study that identifies effects in a particular population may not be simply or easily adopted by the software engineering community as it would be difficult to establish the equivalence in the populations.

In this chapter we describe an experiment that was conducted to establish whether, even in the absence of a similar population profile, it is possible to reproduce the effects predicted in the Assembly Bonus Effect experiment within a Software Engineering group.

3.1 The Assembly Bonus Effect

The assembly bonus effect is a phenomenon in which effective interaction allows group members to combine their individual knowledge in a manner that achieves higher quality outcomes than would have been attributable to any individual members' efforts [62]. In contrast, most tasks have the opposite result due to factors such as the need to communicate [41].

Laughlin et. al. [63] identified that the letters-to-numbers problem was shown to exhibit the assembly bonus effect. The letters-to-numbers problem is a puzzle in which
numbers have been substituted for letters and participants attempt to uncover the hidden mapping in as few steps as possible. In Laughlin et. al.’s experiment, groups of social psychology students were shown to solve the problem more effectively than individuals.

### 3.1.1 Letters-to-Numbers problem

In the letters-to-numbers problem, a set of letters (A, B, C, D, E, F, G, H, I, J) are assigned randomly to the digits (0, 1, 2, 3, 4, 5, 6, 7, 8, 9). For instance, we can have the mapping shown in Table 3.1. It is important to note that the number system is not changed - only the digits have been substituted with numbers.

This mapping is predetermined for each problem but not revealed to the participants. The participants try to uncover this hidden mapping through a series of 10 trials. Each trial contains a proposed equation and a hypothesis, to which answers are provided. For instance, in a single attempt, a participant may propose an equation (comprising of subtraction or addition or a combination of subtraction and addition) which will produce a result that is revealed in terms of the mapping (see Table 3.2). After the response is received from the proposal of the equation, the participant may then propose a hypothesis (see Table 3.3), which will then either be revealed as true (the hypothesis is correct) or false (the hypothesis is incorrect). The participant may then try to fill in the entire mapping of the letters-to-numbers problem before proceeding to the next trial. The score of the participant is the number of trials taken to produce the (correct) entire mapping for the problem. If the participant cannot solve the mapping within 10 trials, they are assumed to require 11 trials. A complete example from a group of three participants is reproduced in Figure 3.1.

### Table 3.1: Example mapping of a letters-to-numbers problem

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
</tr>
</thead>
<tbody>
<tr>
<td>4</td>
<td>6</td>
<td>9</td>
<td>0</td>
<td>1</td>
<td>2</td>
<td>8</td>
<td>7</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

### Table 3.2: Example equation proposal and response

<table>
<thead>
<tr>
<th>Participant interaction</th>
<th>Proposal</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>A + B</td>
<td>ED</td>
<td></td>
</tr>
<tr>
<td>4 + 6</td>
<td>10</td>
<td></td>
</tr>
</tbody>
</table>

### Table 3.3: Example hypothesis proposal and response

<table>
<thead>
<tr>
<th>Participant interaction</th>
<th>Proposal</th>
<th>Response</th>
</tr>
</thead>
<tbody>
<tr>
<td>E = 1</td>
<td>True</td>
<td></td>
</tr>
</tbody>
</table>
Figure 3.1: Example complete problem (reproduced). Responses to participants are in **bold**. Participants finished the problem in 5 trials.

Student ID: ABE-4, ABE-30, ABE-31

<table>
<thead>
<tr>
<th>Trial</th>
<th>Equation</th>
<th>Feedback</th>
<th>Hypothesis</th>
<th>Feedback</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>A + B + C</td>
<td>ID</td>
<td>I = 1</td>
<td>T</td>
</tr>
<tr>
<td></td>
<td>A = B = C = D = E = F = G = H = I = J =</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>A + B + C</td>
<td>IC</td>
<td>C = 3</td>
<td>T</td>
</tr>
<tr>
<td></td>
<td>A = B = C = D = E = F = G = H = I = J =</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>B - A</td>
<td>-I</td>
<td>E = 0</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td>A = 9 B = 4 C = 3 D = 6 E = F = G = H = I = J = 5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>B + C</td>
<td>E</td>
<td>F = 0</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td>A = 9 B = 4 C = 3 D = 6 E = F = G = H = I = J = 5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>F - G</td>
<td>F</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>A = 9 B = 4 C = 3 D = 6 E = F = G = H = I = J = 5</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>A = B = C = D = E = F = G = H = I = J =</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>A = B = C = D = E = F = G = H = I = J =</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>A = B = C = D = E = F = G = H = I = J =</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>A = B = C = D = E = F = G = H = I = J =</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>A = B = C = D = E = F = G = H = I = J =</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
3.1.1.1 Task characteristics of the Letters-to-Numbers problem

**Unitary.** The letters-to-numbers problem as presented must be attempted in its whole. All participants must be present for the entirety of the process, and participate in each trial in order to understand the problem fully. As such, this task is non-divisible and therefore a unitary task.

**Intellective.** The letters-to-numbers problem is an intellective task as there is a correct solution that can be determined and tested against. The strategy for the task can also be evaluated objectively: a strategy that solves a coding in fewer trials is better than one that requires more. A strategy that relies less on luck in uncovering codings can be determined as better than one that does.

**Eureka.** The letters-to-numbers problem has an optimal strategy that can be used to solve the problem in the fewest number of steps. Once this strategy is revealed, all future iterations can be solved simply. Participants that have been subsequently exposed to the optimal strategy have felt that the problem becomes trivial in difficulty.

**Assembly Bonus.** Laughlin et. al. [63] showed in an experiment conducted with participants from a general Psychology university population, that groups that work together outperform nominal groups (defined as collections of individuals whose works are combined after working individually). They also showed that groups outperformed the best performing individuals by using fewer number of trials to arrive at a solution.

**Code-breaking, arithmetic and logical.** The letters-to-numbers problem can be observed to have some other properties. For instance it requires participants to have some understanding of arithmetic and the number system. It also requires participants to logically examine the responses provided to their equations and hypothesis in order to extract clues to the mapping of the numbers. Participants have to identify a sequence of steps that would provide clues as to the hidden code (i.e. mapping) of the problem. The task can be taken as analogous to a code-breaking style task, wherein the objective is to uncover the hidden mapping.

3.1.1.2 Analogy of Letters-to-Numbers problem to programming

Skill in mathematics has been shown to be significant in determining success in programming [64]. Although the same cannot be directly said about the letters-to-numbers problem, familiarity with the number system and arithmetic may aid the participants to propose more advanced techniques to solve the problem. Coding a program can be considered to comprise of a set of activities such as working memory factor, problem identification and problem sequencing elements [65]. Similarly, the Letters-to-Numbers problem requires participants to identify the root problem (a sequence of steps to solve for any hidden mapping reliably) and work out a strategy that uses a sequence of re-
sponses to equations and hypotheses to uncover the mapping consistently in the least number of trials.

3.2 Hypotheses and effects

This experiment was conducted to determine if we can replicate the assembly bonus effect in a group of software developers. Thus, we state the hypothesis for this experiment below:

Hypothesis 1: There is no evidence to suggest that, like the experiment conducted by Laughlin et. al., groups perform better than individuals in the letters-to-numbers problem.

Additionally, extra data is collected to determine if cohesiveness or raw ability would significantly affect the performance of software developers in the letters to numbers problem, leading to the following hypotheses:

Hypothesis 2a: The raw abilities of the constituent group members affect the performance of the groups in the letters-to-numbers problem; that is, the performance of the group is related to the overall abilities of group members.

Hypothesis 2b: The cohesiveness of a group does not affect how well it performs in the letters-to-numbers problem.

3.3 Participant Characteristics

Some information about the characteristics of individuals were collected for study, such as intellectual ability.

Differences between participants specific to particular skill sets was not collected. Since the participants were obtained from a computer science and engineering university environment, it was assumed that they had the basic arithmetic abilities required to attempt the Letters-to-Numbers problem. It was also assumed that the participants had no prior exposure to this specific problem as it had only been published and described in professional research journals. Although similar problems may have been attempted by participants, it is unlikely that they would have encountered the problem presented in the form of the letters-to-numbers problem. As such, it was decided to collect data on intellectual ability to see if this factor plays a role in the effectiveness of how the problem is solved.
3.3.1 Raven’s Progressive Metrics

Intelligence can be defined in terms of Spearman’s $g$. $g$ was proposed as an aggregate measure of how well an individual does in various areas of ability (such as spatial intelligence, reading etc). As such, IQ tests expressly serve to measure this value of $g$ and the Raven’s Progressive Metrics [66] was one of the measures that was developed expressively to measure it [67]. Later, the Raven’s Advanced Progressive Metrics were developed in order to provide greater discrimination among the upper levels of abilities of individuals than required for the standard version of the test. Although the Advanced Progressive Metrics test is a relatively short test that takes 60 minutes to complete, Bors and Stokes [68] developed an even shorter version of the Advanced Progressive Metrics test in 1998 that was shown to reliably reproduce the results of a full test in 20 minutes.

Bors and Stokes’ version of the Raven’s Advanced Progressive Metrics consists of 2 sets of questions. The first set is a practice set consisting of 10 questions in increasing order of difficulty. The experimenter then walks through the first question with the participants and reveals the correct answer (see Figure 3.2). Each question is multiple choice, and participants are to select the correct answer out of eight offered solutions. The participants were then given 10 minutes to answer the rest of the questions. After the elapsed time, the answers were revealed to the participants.

The second set of questions comprise a timed set of 12 assessments. Although the questions are independent, the Raven’s Advanced Progressive Metrics questions are designed to be successively more difficult. The scores obtained from the questions are used to discriminate the intellectual abilities of the participants. Participants attempt the second set of questions in 10 minutes after which time the correct answers are revealed and the scores are tallied.

3.4 Group Characteristics

A cohesiveness test was chosen as cohesion is a measure of how well a group ‘bonds’ and works together. The test selected was a questionnaire that was administered to the group after the activity in order to determine its cohesiveness and also any emergent properties that had developed over the duration of the task.

3.4.1 The Group Environment Questionnaire

Originally, the group environment questionnaire [70] (GEQ) was developed to measure the cohesiveness of sports teams. The questionnaire measures two main components of group cohesion - *Group Integration* and *Individual Attraction to Group*. The *Group*
**Figure 3.2:** Raven's Advanced Progressive Metrics example problem (set 1 question 1). Problem reproduced from [69]

Viewing from the bottom left of the image towards the right, we can conclude that the missing cutout pattern should contain three horizontal dotted lines. Doing the same from the top right downward, we conclude that the missing cutout should contain a single solid vertical line. Combining these observations, we conclude that option 8 is the correct solution for the missing cutout.
Integration dimension measures the perspective of cohesion in terms of how well the group works together; Individual Attraction to Group dimension measures how well (comfortable or integrated) the individuals feel to be part of the group.

Each dimension is further broken down into task and social aspects. The task aspects are directly related to the functioning of the group during operating conditions (i.e. while playing the sport) whereas the social aspects were related to how well and how often the group operated at other times. The GEQ consists of 18 questions in total, of which some were reverse coded. Reverse coded questions are where the question has been phrased opposite to that of its original intention. For example, in order to measure the happiness of a participant with the level to complete a puzzle, one might phrase the question as “I am happy with the group’s level of desire to complete the puzzle”. In this case, a strongly agree response would indicate the participant’s strong happiness with the group’s level of desire to complete the puzzle. In the case of a reverse coding, the question is reversed to read “I am unhappy with the group’s level of desire to complete the puzzle”. Therefore, for this statement, a strongly agree response should be translate to the participants extreme unhappiness with the group’s level of desire to complete the puzzle. The responses received are on a 9 point Likert scale. The score of each aspect within each dimension is summed together (taking into account any reverse coding).

Although the GEQ in its original form was designed for measuring the cohesion of sports teams, the authors developed a set of guidelines to modify the questionnaire for different settings [70]. The guidelines mainly relate to the wording of the questions and how they should be adapted to fit the task and group being studied. The guidelines state that researchers using the GEQ should consider if (reproduced from [70]):

1. The items in the GEQ could be used directly, with no modification.

2. The wording of the items that appear to be useful but contains language, terminology or situational reference not characteristic of the group(s) under focus should be changed.

3. Items not appropriate should be deleted.

The GEQ used in this experiment was modified accordingly. The original version of the GEQ is shown in AppendixA

3.5 Goals

The goals of this experiment are as follows:
Establishing a baseline. The first goal is to establish a baseline for the behaviour of Software Engineers in a task that was previously done with Psychologists. We seek to find out if the results would be the same as the study conducted by Laughlin et. al. [63], or if there are systematic differences that can be identified due to the different populations.

Emergent Properties. The second goal is to establish if groups of Software Engineers exhibit any emergent behaviour when attempting a task with the characteristics of the Letters-to-Numbers problem. This should allow us to predict how similar tasks in the future should be approached when done by groups.

3.6 Method

In this section we discuss our experimental set up including participants, demographic information, materials, and procedure.

3.6.1 Participants

The participants were 37 Masters students that were enrolled in a requirements elicitation course at the Australian National University. The majority of the participants were enrolled in the Masters of Computing program, with four participants enrolled in the Masters in Information Technology Studies program. The participants were majority males (34) with only three females. With the exception of three participants that have been enrolled for more than one year, and four participants that are part time, the rest of the participants (30) have been enrolled for one year or less and are full time students. These 37 participants were randomly assigned to work together in nine groups of three participants each, known from here on as triads (treatment), and the rest were assigned to work alone as individuals (control).

3.6.2 Materials

Each participant was given the following:

1. A consent form and cover sheet, explaining the exercise as an experiment.

2. Bors & Stoke's shortened Raven's Advanced Progressive Metrics [68].

3. Presentation handouts for the instructions of the letters-to-numbers problem.

In addition, each triad received a modified Group Environment Questionnaire.
3.6.2.1 The modified Group Environment Questionnaire

Based on the guidelines in Section 3.4.1 it was decided to modify the GEQ by removing questions relating to the social aspects of both the Individual Attraction to Group and the Group Integration dimensions of the questionnaire. This was done because the groups formed for this experiment were short lived and only existed in the context of the experiment. In addition, the following rules were used to modify the GEQ questions to suit the context of the experiment:

- Where the GEQ original text refers to "playing time" or any sport related activity, the text was changed to refer to the letters-to-numbers problem as "the puzzle".

- The original item that references communication about athlete's responsibilities during competition was changed to be a generic question on the freedom of communication experienced by the group members during the task.

As much as possible, the original GEQ was preserved. For instance, the reverse coded questions were kept. That is, if a specific question was reverse coded in the GEQ, then the modified version also was reverse coded. The ordering of questions was also kept as similar as possible, even with the removal of the social aspect questions. The final GEQ used is attached in Figure 3.3.

3.6.2.2 The Letters-to-Numbers problem: A computer implementation

Administering the letters-to-numbers problem as described in Laughlin et. al. [63] was deemed too labour intensive. As a result, a website was developed in order to automate the process. This website was implemented to account for all of the rules present in Laughlin et. al's Letters-to-Numbers problem description. The website generates a random coding of letters to numbers for each attempt at the problem, and responds to both the equation proposal and the hypothesis appropriately. Each response is saved by the website which also determines when the attempt should end (either due to the participants running out of trials, or the complete coding being determined).

The system presents the letters-to-numbers problem by trial. Upon logging into the website, the participants are presented with the input fields for the first trial (see Figure 3.4). The web page javascript executes a randomisation of the letters-to-numbers problem that is to be solved. This randomisation is stored in memory and cannot be obtained by looking at the web page source code. Each trial presents the participants with three interactive buttons, corresponding to the responses that can be obtained after inputs. The participant first inputs an equation and clicks the "calculate" button, which then produces an answer in the required form. The web page has been programmed to lock the associated input field to prevent subsequent changes. The same
Figure 3.3: Sample GEQ Response sheet for experiment

Group Activity Questionnaire

ID: 1426

This is a quick survey to see how much you liked this bit of the course pilot.

In the following questions help assess your feelings about your personal involvement with your group. On a scale of 1 through 9, 1 indicating the strongest agreement, and 9 indicating the strongest disagreement, answer each question by circling the appropriate response.

<table>
<thead>
<tr>
<th>I am unhappy about the level of involvement I had with my group members</th>
<th>1 (strongly agree) 2 3 4 5 6 7 8 9 (strongly disagree)</th>
</tr>
</thead>
<tbody>
<tr>
<td>I am unhappy with the group's level of desire to complete the puzzle</td>
<td>1 (strongly agree) 2 3 4 5 6 7 8 9 (strongly disagree)</td>
</tr>
<tr>
<td>My group members did not explain their technique for approaching the puzzle to me</td>
<td>1 (strongly agree) 2 3 4 5 6 7 8 9 (strongly disagree)</td>
</tr>
<tr>
<td>I like the working style of the group</td>
<td>1 (strongly agree) 2 3 4 5 6 7 8 9 (strongly disagree)</td>
</tr>
<tr>
<td>My group was united in trying to solve the puzzle</td>
<td>1 (strongly agree) 2 3 4 5 6 7 8 9 (strongly disagree)</td>
</tr>
<tr>
<td>We all took responsibility for the level of performance in solving the puzzle</td>
<td>1 (strongly agree) 2 3 4 5 6 7 8 9 (strongly disagree)</td>
</tr>
<tr>
<td>Our group had conflicting aspirations concerning our performance in solving the puzzle</td>
<td>1 (strongly agree) 2 3 4 5 6 7 8 9 (strongly disagree)</td>
</tr>
<tr>
<td>Our group communicated freely with each other</td>
<td>1 (strongly agree) 2 3 4 5 6 7 8 9 (strongly disagree)</td>
</tr>
</tbody>
</table>
is done for the hypothesis and solution. After the trial is completed, if the problem isn’t solved, a new trial appears for the participant, with the previous trial input and responses still remaining visible (see Figure 3.5). This arrangement continues until the participant has performed 10 trials, or solves the coding completely. All responses are logged for analysis.

Although the computer implementation of the letters-to-numbers problem can be said to introduce an additional layer of complexity, in this context it can be argued that this does not apply. The participants used in this experiment are computer majors and therefore should be very familiar and comfortable with using browsers and computers in general. Therefore, we consider it reasonable to assume that the computer implementation of the letters-to-numbers problem only serves to minimize the labour requirements of administering the problem and does not detract from the focus of the experiment.

3.7 Procedure

The experimental session ran for 1.5 hours as part of the first class of the semester for all participants. The session was highly scripted in order to control for repeatability, and the script was pre-examined for any possible bias (for experimental script, see Appendix B). The session began with an introduction of the experimenter and an explanation of the structure of the experiment. The participants first complete, individually, Raven's Advanced Progressive Matrices 1998 Set I, followed by Bor's and Stoke's Raven's Short Test. The participants were then divided into two groups, one of which formed triads (nine triads total) and the other consisting of 10 individuals. The triads and individuals then attempt the computer implementation letters-to-numbers problem in their respective assigned configurations.

The letters-to-numbers problem was attempted in a computer lab that was equipped with Linux machines connected to the web. Each individual was seated in front of a machine (control group), whereas the triads were seated in groups of three to a machine(experimental group). The lab was set up such that there was a dividing wall between the two groups of participants, but a common presentation area to ensure that they were all able to receive the same instructions for the letters-to-numbers problem together (see Figure 3.6).

The presentation on the letters-to-numbers problem closely follows the procedure used for Laughlin et. al.'s own experiment [63]. In order to control for any learning effects, each triad and individual was told to attempt the problem twice. Upon completion of both letters-to-numbers problems, the triads were told to complete the GEQ, and thanked for their participation.
### Figure 3.4: Letters-to-Numbers problem implementation - first trial.

<table>
<thead>
<tr>
<th>Trial 1</th>
<th>Equation</th>
<th>Hypothesis</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Solution?</th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
<th>G</th>
<th>H</th>
<th>I</th>
<th>J</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- [Test]
<table>
<thead>
<tr>
<th>Trial 1</th>
<th>Equation</th>
<th>Hypothesis</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>$a + b = d$</td>
<td>$d = 9$ is false</td>
</tr>
<tr>
<td>Solution?</td>
<td>$a = 3$</td>
<td>$b = 4$</td>
</tr>
<tr>
<td>Trial 2</td>
<td>Calculate</td>
<td>Test</td>
</tr>
<tr>
<td>Solution?</td>
<td>A</td>
<td>B</td>
</tr>
</tbody>
</table>
Figure 3.6: Computer Lab seating arrangement for Letter-to-Numbers experiment.
The experimental session was run as part of the class material and some basic ideas on tasking covered in Table 1.1 was presented. In addition, the identities of the participants were obfuscated and not revealed until the analysis of results was complete.

Obfuscation of the participant identities in this experiment was achieved by simply not collecting participant data. All handouts given to the participants had the “ID” field pre-filled, and the handouts were shuffled before distribution.

### 3.8 Results

Some participants were excluded from the final analysis. Four individuals were removed from analysis as they turned up late to class and did not participate in the initial briefing, nor the earlier components of the experiment. Due to an oversight in the presentation, the time limit was not stated to all the participants involved. This is directly in contention to Laughlin et al.’s study, therefore we deemed that it was prudent to remove the results from two triads and three individuals for non-completion. This left a total of 24 participants in six triads (treatment) and six individuals (control).

#### 3.8.1 Individual Characteristics

The participants’ Raven’s test scores ranged from 4 to 10. The highest and most common score was 10, with 29% of the participants obtaining this score. This is closely followed by scores of 8 and 9, each of which were obtained by 21% of the participants. The score breakdown is shown in Figure 3.7.

The scores had a median value of 8.5, with the majority of the participants obtaining a score of between 7 and 10 (See Figure 3.8). The descriptive statistics are shown in Table 3.4.
Figure 3.7: Bors and Stoke's shortened Raven's Advanced Progressive Metrics score & number of participants with a particular score

Figure 3.8: Boxplot of shortened Raven's Advanced Progressive Metrics Score
The descriptive statistics and the boxplot both indicate that the data is slightly skewed towards the higher end of the results. The kurtosis indicates that the sample peaks in a similar way to a standard normal distribution. The skewness does not fall outside the normal accepted range for determining if a distribution is approximately normal (skewness falls within the values of $-2 \times 0.45216$ and $2 \times 0.45216$).

Six triads were formed from randomly chosen participants. The average of the shortened Raven’s Advanced Progressive Metics is used as an aggregate measure of the group ability. This average was compared against the remaining individuals using an ANOVA test. Although the ANOVA test only applies for a population with a normal distribution, we make this assumption in order to get an idea of how different the two groups were in terms of ability. The F value of the ANOVA test was 0.04, with the p value at 0.845. Therefore, if the ANOVA test holds for the two groups, we can say that there is insufficient evidence to show that the distribution of ability was significantly different between the triads and the individuals (see Figure 3.9 for plotted scores).

### 3.8.2 Group Characteristics

Cronbach’s Alpha [71] values were used to calculate the internal consistency of the items within the GEQ responses. Since the GEQ that was administered consisted of two dimensions, Individual Attraction to Group and Group Integration, Cronbach’s Alpha values were obtained to test for reliability of these two dimensions. Each dimension corresponded to 4 constituent questions.

It was found that the values associated with the variables were low (less than 0.7, see Table 3.6). Due to these weak values, it was decided to use the individual questions for regression analysis instead of the aggregate variables themselves. Although the weak values meant that a direct statement about the cohesiveness of the group could not be made, the constituent questions may provide information about how the members interacted in the triads.

The constituent questions have been assigned the labels shown in Table 3.5. The reverse codings for the constituent questions have been removed for analysis (where applicable).

### 3.9 The Letters-to-Numbers Performance: Triads VS Individuals

The performance of triads and individuals in the letters-to-numbers task was compared and the results are presented in Table 3.7. Individuals on average needed more trials to solve the letters to numbers problem. The overall (sum of attempt 1 and attempt
Figure 3.9: Shortened Raven's Advanced Progressive Metrics individuals and triads comparison
Table 3.5: GEQ constituent questions and labels used for analysis

<table>
<thead>
<tr>
<th>Identifier</th>
<th>Question</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATT1</td>
<td>I am happy about the level of involvement I had with my group members</td>
</tr>
<tr>
<td>ATT2</td>
<td>I am happy with the group’s level of desire to complete the puzzle</td>
</tr>
<tr>
<td>ATT3</td>
<td>My group members explained their technique for approaching the puzzle to me</td>
</tr>
<tr>
<td>ATT4</td>
<td>I like the working style of the group</td>
</tr>
<tr>
<td>INT1</td>
<td>My group was united in solving the puzzle</td>
</tr>
<tr>
<td>INT2</td>
<td>We all took responsibility for the level of performance in solving the puzzle</td>
</tr>
<tr>
<td>INT3</td>
<td>Our group had united aspirations concerning our performance in solving the puzzle</td>
</tr>
<tr>
<td>INT4</td>
<td>Our group communication freely with each other</td>
</tr>
</tbody>
</table>

Table 3.6: Cronbach’s Alpha reliability test values for GEQ dimensions

<table>
<thead>
<tr>
<th>Items Tested</th>
<th>Cronbach’s Alpha</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATT1, ATT2, ATT3, ATT4</td>
<td>0.326</td>
</tr>
<tr>
<td>INT1, INT2, INT3, INT4</td>
<td>0.635</td>
</tr>
</tbody>
</table>

2) was also lower for triads than individuals.

Assuming that the underlying distributions of the results of the letters-to-numbers problem for both the individuals and the triads were normal, an ANOVA test was performed to see if the differences between the results were significant. For both the first, second and overall attempts, the score for the triads is significantly different to the individuals. The ANOVA test that was run used the shortened Raven’s Advanced Progressive Metrics score as a covariate, to see if there was any relationship between the letters-to-numbers scores and the shortened Raven’s Advanced Progressive Metrics score (results are shown in Figure 3.10). There was insignificant evidence to show that Raven’s Advanced Progressive Metrics score and the Number of Attempts were related as covariates (see Table 3.8).

### 3.10 Emergent Group Behaviour

Due to the weak Cronbach’s Alpha values obtained on the GEQ (see Section 3.8.2), it was decided to use the individual questions in the GEQ in stepwise regression. The

Table 3.7: Letters-to-Numbers problem performance compared between triads and individuals (lower is better)

<table>
<thead>
<tr>
<th></th>
<th>First Attempt</th>
<th>Second Attempt</th>
<th>Overall</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>μ</td>
<td>σ</td>
<td>μ</td>
</tr>
<tr>
<td>Triads</td>
<td>5.67</td>
<td>2.34</td>
<td>5</td>
</tr>
<tr>
<td>Individuals</td>
<td>9.17</td>
<td>1.47</td>
<td>8.5</td>
</tr>
</tbody>
</table>
Figure 3.10: Letters-to-Numbers results arranged by shortened Raven’s APM Score.
Table 3.8: Letters-to-Numbers problem performance ANOVA analysis between triads and individuals with shortened Raven’s Advanced Progressive Metrics as a co-variate

<table>
<thead>
<tr>
<th></th>
<th>F value</th>
<th>p value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Triads against individuals (first attempt)</td>
<td>35.82</td>
<td>&lt; 0.001</td>
</tr>
<tr>
<td>shortened Raven’s test score (co-variate, first attempt)</td>
<td>1.10</td>
<td>0.309</td>
</tr>
<tr>
<td>Triads against individuals (second attempt)</td>
<td>15.51</td>
<td>&lt; 0.001</td>
</tr>
<tr>
<td>shortened Raven’s test score (co-variate, second attempt)</td>
<td>2.24</td>
<td>0.154</td>
</tr>
<tr>
<td>Triads against individuals (overall)</td>
<td>45.60</td>
<td>&lt; 0.001</td>
</tr>
<tr>
<td>shortened Raven’s test score (co-variate, overall)</td>
<td>3.55</td>
<td>0.078</td>
</tr>
</tbody>
</table>

Table 3.9: t statistics for significant factors in stepwise regression for the first attempt at the letters-to-numbers problem

<table>
<thead>
<tr>
<th>Terms</th>
<th>Wald Statistic</th>
<th>d.f.</th>
<th>t Stat</th>
<th>p Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATT1</td>
<td>5.09</td>
<td>1</td>
<td>5.09</td>
<td>0.044</td>
</tr>
<tr>
<td>ATT3</td>
<td>15.39</td>
<td>1</td>
<td>15.39</td>
<td>0.002</td>
</tr>
<tr>
<td>INT1</td>
<td>8.44</td>
<td>1</td>
<td>8.44</td>
<td>0.013</td>
</tr>
<tr>
<td>INT2</td>
<td>18.79</td>
<td>1</td>
<td>18.79</td>
<td>&lt; 0.001</td>
</tr>
<tr>
<td>INT4</td>
<td>9.27</td>
<td>1</td>
<td>9.27</td>
<td>0.010</td>
</tr>
</tbody>
</table>

questions in the administered version of the GEQ relate to particular activities and sentiments of particular group members towards other members of the group. With this we may be able to identify any emergent behaviours or attitudes that members had developed over the course of attempting the letters-to-numbers problem. The GEQ questions are labelled ATT1 - ATT4 and INT1 - INT4 with relation to the dimensions that the questions were supposed to originally affect.

3.10.1 Stepwise Regression: First Attempt

A stepwise procedure for deriving regression equations was performed in order to identify factors that could explain the performance of the triads in the first attempt of the letters-to-numbers problem. The results are presented in Table 3.9. The coefficients associated with the significant factors are produced in Table 3.10. The regression of the terms were significant (F(5, 12) = 7.40, p = 0.002, R² = 0.76).

Written as an equation,

\[ ATT1 = 1.97 + 1.09(ATT1) - 0.85(ATT3) + 0.96(INT1) - 1.56(INT2) + 0.69(INT4) \]

3.10.2 Stepwise Regression: Second Attempt

A stepwise procedure for deriving regression equations was performed in order to identify factors that could explain the performance of the triads in the second attempt of
Table 3.10: Regression Coefficients for the first attempt at the letters to numbers problem

<table>
<thead>
<tr>
<th></th>
<th>Coefficients</th>
<th>t Stat</th>
<th>p Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>1.97</td>
<td>0.55</td>
<td>0.59</td>
</tr>
<tr>
<td>ATT1</td>
<td>1.09</td>
<td>2.26</td>
<td>0.04</td>
</tr>
<tr>
<td>ATT3</td>
<td>-0.85</td>
<td>-3.92</td>
<td>0.002</td>
</tr>
<tr>
<td>INT1</td>
<td>0.96</td>
<td>2.91</td>
<td>0.01</td>
</tr>
<tr>
<td>INT2</td>
<td>-1.56</td>
<td>-4.33</td>
<td>&lt; 0.01</td>
</tr>
<tr>
<td>INT4</td>
<td>0.69</td>
<td>3.05</td>
<td>0.01</td>
</tr>
</tbody>
</table>

Table 3.11: t statistics for significant factors in stepwise regression for the second attempt at the letters-to-numbers problem

<table>
<thead>
<tr>
<th>Terms</th>
<th>Wald Statistic</th>
<th>d.f.</th>
<th>t Stat</th>
<th>p Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATT3</td>
<td>8.336</td>
<td>1</td>
<td>8.34</td>
<td>0.011</td>
</tr>
<tr>
<td>INT2</td>
<td>5.50</td>
<td>1</td>
<td>5.50</td>
<td>0.033</td>
</tr>
</tbody>
</table>

The letters-to-numbers problem. The results are presented in Table 3.11. The coefficients associated with the significant factors are produced in Table 3.12. The regression of the terms were significant (F(5, 12) = 6.76, p = 0.008, R^2 = 0.47).

Written as an equation,

\[ \text{ATTEMPT}_2 = 12.07 - 0.558(\text{ATT3}) - 0.329(\text{INT2}) \]

3.10.3 Stepwise Regression: Overall

A stepwise procedure for deriving regression equations was performed in order to identify factors that could explain the overall performance of the triads of the letters-to-numbers problem. The results in Table 3.13 were obtained. The coefficients associated with the significant factors are produced in Table 3.14. The regression of the terms were significant (F(5, 12) = 6.41, p = 0.010, R^2 = 0.461).

Written as an equation,

\[ \text{ATTEMPT}_{TOT} = 24.868 - 1.135(\text{ATT3}) - 0.645(\text{INT2}) \]

Table 3.12: Regression Coefficients for the second attempt at the letters to numbers problem

<table>
<thead>
<tr>
<th></th>
<th>Coefficients</th>
<th>t Stat</th>
<th>p Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>12.07</td>
<td>6.13</td>
<td>&lt; 0.001</td>
</tr>
<tr>
<td>ATT3</td>
<td>-0.558</td>
<td>-2.89</td>
<td>0.011</td>
</tr>
<tr>
<td>INT2</td>
<td>-0.329</td>
<td>-2.35</td>
<td>0.033</td>
</tr>
</tbody>
</table>
Table 3.13: t statistics for significant factors in stepwise regression for the overall attempt at the letters-to-numbers problem

<table>
<thead>
<tr>
<th>Terms</th>
<th>Wald Statistic</th>
<th>d.f.</th>
<th>t Stat</th>
<th>p Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATT3</td>
<td>8.134</td>
<td>1</td>
<td>8.13</td>
<td>0.012</td>
</tr>
<tr>
<td>INT2</td>
<td>4.988</td>
<td>1</td>
<td>4.99</td>
<td>0.041</td>
</tr>
</tbody>
</table>

Table 3.14: Regression Coefficients for the overall attempt at the letters to numbers problem

<table>
<thead>
<tr>
<th></th>
<th>Coefficients</th>
<th>t Stat</th>
<th>p Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intercept</td>
<td>24.868</td>
<td>6.138</td>
<td>&lt; 0.001</td>
</tr>
<tr>
<td>ATT3</td>
<td>-1.135</td>
<td>-2.852</td>
<td>0.012</td>
</tr>
<tr>
<td>INT2</td>
<td>-0.645</td>
<td>-2.233</td>
<td>0.041</td>
</tr>
</tbody>
</table>

3.11 Insights from Analysis

Hypothesis 1 is rejected. Based on the results obtained in section 3.9, there is sufficient evidence to show that hypothesis 1 can be rejected - similar to the results obtained by Laughlin et. al. on psychology students, groups tended to outperform individuals in the letters-to-numbers problem.

Hypothesis 2a is rejected. Based on the data obtained via both ANOVA analysis with Raven’s Short Test score as a co-variate and stepwise regression, we can say that there is sufficient evidence to show that hypothesis 2a can be rejected - the raw ability of constituent group members has little affect on the performance of groups in the letters-to-numbers problem.

Unable to reject Hypothesis 2b. Due to the poor Cronbach’s Alpha values obtained (see Section 3.8.2), we were unable to use the GEQ as intended to study if group cohesiveness significantly affected the performance of groups in the letters-to-numbers problem. The individual items of the GEQ, however, were broken up and regression analysis was applied to see if any emergent behaviour could be identified. These behaviours are covered below.

3.11.1 Performance in attempt 1 is related to measured communication

Based on the regression coefficients in Table 3.10, the performance of the triads in the first attempt at the letters-to-numbers problem is significantly positively affected if participants rated lowly their happiness about the level of involvement with their group members (ATT1). It was also significantly positively affected if participants rated lowly their perceptions of group unity in solving the puzzle(INT1). Performance was significantly positively affected if the participants perceived communications were
somewhat measured (INT4). In addition, it was significantly positively affected if the participants rated highly their perceptions of other triad members explaining the puzzle solving techniques (ATT3) and the perceptions of the triad level of responsibility for solving the puzzle (INT2).

We can surmise from the effects of the variables on the regression equation derived that in the first attempt of the letters-to-numbers problem that successful groups tended to exhibit a lower amount of perceived unity in solving the puzzle and a lower amount of free communication. Successful group members tended to be unhappy about the level of involvement with group members. This is consistent with the steps associated with production blocking (see Section 2.3.1) as a more divided group with lower amounts of communication would have more time to think about their own solutions for the puzzle at the start, similar to the solution proposed by the nominal group technique. A high perception in taking responsibility for the level of performance in solving the puzzle is consistent with social psychology research recommendations to reduce social loafing (see Section 2.3.2). Finally, evaluation apprehension (see Section 2.3.3) is addressed by the high level of desirability that group members explain their puzzle techniques, which can be viewed as a high desire for relevant communication on the subject.

3.11.2 Performance in attempt 2 is related to high visibility and directed communication

Based on the regression coefficients in Table 3.12, the significant predictors for the performance of triads in the letters-to-numbers problem are explaining the technique of the puzzle and responsibility for the puzzle. The performance of the triads is significantly positively affected by high perception that group members explained the puzzle techniques to other members (ATT3) and that group members took greater responsibility for solving the puzzle (INT2).

We can surmise that in the second attempt of the letters-to-numbers problem, successful groups tended to value more highly the explanation of the puzzle and a higher level of responsibility in solving the puzzle. This is consistent with norms that protect from process loss in the letters-to-numbers problem. High visibility and high directed address social loafing issues (see Section 2.3.2) and production blocking issues (see Section 2.3.1)
3.11.3 Overall performance in the letters-to-numbers problem is driven by high visibility and highly directed communication

Similarly to the performance in the second attempt of the letters-to-numbers problem, the regression coefficients for the overall performance is shown to be significant for the explaining of the technique of the puzzle and the responsibility of the puzzle. The performance of the triads is positively related to high perception that the other members (ATT3) explained the puzzle techniques and also that the group members took greater responsibility (INT2).

Therefore, overall, successful groups tended to exhibit more highly the explanation of the puzzle and a higher level of responsibility in solving the puzzle. This is consistent with norms that protect from process loss in the letters-to-numbers problem. High visibility and high directed address social loafing issues (see Section 2.3.2) and production blocking issues (see Section 2.3.1)

3.12 Limitations and Threats to Validity

Convenience Sample VS True Population Sample. While it is possible that we can make a good case for our convenience sample of Software Engineering Masters students as being a likely sample of the general population, we have not proven that this is actually the case. If the participants in the experiment indeed are biased in some way when compared with the Software Engineering population, then the generalisability of our results will be limited to group sharing a similar profile to those that were studied; that is, the generalisability of the results are limited to student participants in a Masters course in a single university.

Small Sample Size. Another problem with the generalisability of the results is the small sample size; this is evident in the fact that the Cronbach's Alpha values did not match. Due to this, the aggregate values for cohesion were not used and the analysis was limited to the constituent items.

Specificity of Work. Although this experiment successfully applies the letters-to-numbers experiment to a group of software engineers, we cannot assume that this validates the entire body of literature of social psychology so that results and theories can be adopted wholesale into the software engineering domain. Unless some direct proof can be found that Software Engineers are indeed a population that conforms with the properties of the populations that were studied in the Social Psychology field, predicted phenomena should be retested before adoption in more experiments.
3.13 Discussion

The questions of which tasks are better assigned to groups than individuals, and what factors determine the performance of the group has received relatively little attention in software engineering literature. The decision to assign tasks, or parts of tasks, to groups instead of individuals is more often informed by pragmatical, local constraints, rather than insight into the nature of tasks, group dynamics or how these factors interact [72]. This, to a large extent, is due to the fact that knowledge generated by social psychological research that might potentially be useful to software engineering has not yet infiltrated the software engineering community.

When implementing the results of social psychological research in other fields, it is important to consider the context validity of that research. Strong empirical and scientific reasons must be given to the appropriateness of applying the findings from social psychology research in software engineering. In the experiment conducted in this chapter, it was found that the results corroborate those of Laughlin et. al, even though it can be argued that the participants in the experiment were drawn from different populations. This work should be viewed as a first step in adopting relevant ideas on this particular area into the field of software engineering.

It is worth nothing here that, based on visual inspection of the figure 3.10, it is possible that there may exist an inverse relationship between the shortened APM score and the total number of attempts required to solve the Letters-to-Numbers problems over two trials. However, an investigation into determining if this effect indeed exists is not within the scope of this thesis and will not be treated here.

3.13.1 Known Issues with the Assembly Bonus Effect

Within the Psychological Community, the Assembly Bonus Effect is contentious. First demonstrated in Collins & Guetzkow's work [73], this effect has both been replicated on multiple occasions [74; 63] and also challenged on others [75; 76; 77].

One of the largest arguments against the existence of the Assembly Bonus Effect is Pavitt [77], who attacks the Assembly Bonus Effect from the standpoint of reductism and from within the field of human communication - he argues that, in practice, groups tend to choose the solution offered by "the most confident and (presumably persuasive) member, who is often not the group's most task competent member. In addition, he shows that process loss is greater for greater numbers of group members, and with no exception, the groups under perform when compared with a totally lossless, ideal information interchange model. He, however, concedes that "in every study, groups outperform individuals" although "group cognition is limited by and cannot transcend individual cognition, and that communication does not allow us to function optimally
in group settings”.

Propp [78] responds by pointing out that “if one looks only at the end product and attempts to deduce whether interaction created a process gain or loss, it is possible that the effects may cancel each other”. She gives an example - “when studying [Transactive Memory Systems], it is possible that a group may be able to remember items of information that no individual recalled - a clear example of an assembly bonus effect. At the same time, however, the group may omit items of information that its individual members initially were able to remember - a clear example of reductionism. If one studies only the summative data, both of these effects are lost.” Therefore, although Pravitt’s arguments must be considered strongly, in order to “substantiate (or refute) the assembly bonus effect while taking a communication perspective, researchers must analyse process gains and performance at a more microlevel - in the interaction of the group”.

Pavitt’s arguments do not apply in our experiment - in our experiment we examine if a pool of individuals will be capable of producing a solution that is on average better than a pool of groups. Indeed, the task used is intellective and presumes that the group will recognise a better solution when presented, and the strength of the solution can be verified by the mechanism inherent in the task itself (i.e. by simply comparing the number of steps required). This removes some of the constraints imposed by Pavitt’s more detailed study (of group communication and the bonus effect of recalling information), and moves it back into the domain of group performance in a particular task type.

In conclusion, although Pavitt’s arguments do have merit in the case of the assembly bonus effect in communication, the experiment we conducted falls outside the much wider scope of Pavitt’s work. This experiment seeks to identify if the ABE can be demonstrated by Software development groups as shown by Laughlin in his original experiment, and if the related predictions and theories on this particular task can be applied to Software groups.

3.13.2 Application of Results

If a task can be identified as having similar characteristics as the letters-to-numbers problem, then we reason that it is better assigned to a group rather than a single high performing individual. Based on our results, it would be preferential for a task matching the characteristics of the letters-to-numbers problem to be assigned to a group of no outstanding ability than a single individual of high ability as this would give a much better chance of a good performance.

Assigning a group of lower performers to a task in preference to a single high
performer could have various benefits in limited resource situations. For example, in a scenario where teams that consist of a fixed membership, it may be more economical to reserve the single high performer for a task where they would be able to provide a larger relative contribution.

When deciding to assign a task to either a group or an individual, it is important to have an idea which characteristics play a significant role in determining the success of completing that task. Research into tasks or combinations of tasks show that each generic type of task comes with its own particular challenges that sometimes can be met best by an individual, whilst others by a group. For tasks that are best assigned to groups, it is also clear that a successful outcome is never just driven by task characteristics - rather, it is the emergent product of interaction between individual, group and task characteristics.

One shortcoming of other research into letters-to-numbers tasks is that no attempt has been made to examine which other characteristics, other than those of the task itself, determine the outcome. The letters-to-numbers task was chosen in this experiment because it seemed to capture some key processes of real-world tasks within groups without sacrificing a great deal of experimental control. We showed that for the letters-to-numbers task, it is the group characteristics that substantially affect performance.

3.13.2.1 Investigating other tasks

The method used here to investigate the properties of the letters-to-numbers task in a group environment can possibly be extended to other tasks. Investigating the emergent properties of other tasks could yield interesting results on how best to assign tasks in Software development groups.

The method used consists of 3 steps as follows: Firstly, the investigators must categorise the task picked for experiment in terms of its properties as shown in Section 3.1.1.1. Secondly, the measures of individual characteristics must be considered - in the case of this experiment, only the intellectual abilities were deemed sufficient (see Section 3.3); however, in the context of other tasks with particular skill sets, additional measures may be required such as experience, expertise or ability. Finally, group characteristics should be considered as described in Section 3.4.

3.14 Related Work

Our work in this experiment is related to work in 7 areas, namely the Assembly Bonus Effect, Geographic and Virtual Software Teams, Personality Compatibility, Paired Programming, Group Norms in Software Teams and Social Psychology.
Assembly Bonus Effect. The Assembly Bonus Effect literature is covered in Human Communications Research [77; 78; 75; 76] and Group Research [73; 63; 74]. In contrast to the identification of why and how the Assembly Bonus Effect occurs, our work is based on applying the Assembly Bonus Effect on relevant task types where it has been identified to exist in order to improve the performance of groups.

Geographic and Virtual Software Teams. Research in the area of geographic and virtual software teams [79; 80; 81; 82; 83] focuses on the coordination of geographically distributed software teams and issues such as differences in teamwork style, problems with information flow, time differences and cultural norm issues. The study of effective norms for productive norms across geographically distributed teams is different from the focus of this thesis, which is on implementing effective norms for successful task completion.

Personality Compatibility. The work in the area of personality compatibility [32; 26] focuses on the compatibility of individuals in teams. In contrast, this work focuses on the successful interactions and intra-team behaviors that should be changed based on the characteristics of task type. In addition, this work focuses on group norms, which will be established organically in groups regardless of member compatibility.

Paired Programming. Although paired programming places some emphasis on creating successful and effective duos, Hackman & Vidmar [48] showed that in the area of group effects, the effects of dyads do not fit in with the extrapolation of group effects. Social psychology regards the triad as the smallest possible group - due to this, our experiment focus is fundamentally different from the literature focused on pairs and the results for dyads are not applicable to the results related to groups.

Group Norms in Software Teams. Perlow's [49] work on differences in the social norms of various software engineering groups in different countries is different from our examination of organic norms and how particular norms can possibly be encouraged in groups in order to promote greater productivity. The work by Perlow, although significant, does not draw a relationship between normative benefits and how they may be harnessed, rather, it recognises that productive groups can adopt different norms but still be effective.

Social Psychology. Although the work here borrows heavily from the literature of social psychology, its reapplication in the software engineering domain represents new knowledge that is different from established general studies. For instance, the original letters-to-numbers task was performed on social psychology students in the psychology domain [63]. Further, studies that use IT groups to examine social effects have largely been limited to computer mediated communication [84].
3.15 Conclusion

This first experiment surrounding the letters-to-numbers problem shows that Software Engineering groups do exhibit the same assembly bonus effect predicted by Social Psychology experiments. It shows that in certain situations it may be possible that groups will outperform individuals. In addition, it uncovers some emergent properties that were adopted by the better groups, which can be applied to similar tasks. For example, groups that had performed well in the letters-to-numbers problem were not necessarily aware that they had performed well as groups. Although they felt like they were not communicating well, it meant that production blocking was lowered. Groups that exhibited preferences to behaviours that were consistent with efficient brainstorming, low social loafing and low evaluation apprehension tended to perform better. It was also found that ability was not a significant factor in affecting the group performance in the letters-to-numbers problem.

This finding has immediate relevance to software engineering; We established a baseline for tasks like the letters-to-numbers problem, and showed the conditions necessary to perform well in these like tasks. In the context of this thesis we show that the area of Social Psychology theory may be applicable to Software development groups, and that Software development groups may behave the same way as groups in other domains and thus exhibit predictable effects.
The first experiment in Chapter 3 established that it is possible for predicted social psychological effects to also be exhibited in software engineering groups. It also showed that many of the same group performance enhancing norms were exhibited by the high performing software groups. This part of the thesis looks at the effectiveness of manipulating norms in such a way that that groups adopt behaviour that is beneficial to the performance of a particular task. This part of the work attempts to answer the following questions:

- Is it possible to identify positive social norms that might be instilled for a particular type of task from social psychology literature, which may in turn benefit Software groups?

- Is it possible to instil artificial norms in a software group?

- Is it possible for artificially instilled norms to improve the performance of the software group?

4.1 Norms and Information Sharing background

Groups have been found to be relatively poor at sharing information. Wittenbaum & Stasser [85] found that group members tend not to pool critical information, which in turn resulted in poor quality decisions being made. Stasser & Titus [86] also showed that the act of not sharing information unique to particular group members tended to drastically affect the quality of group performance in particular tasks. In their study, they had groups attempt to make a decision in hiring a candidate for a particular job, with select information only made available to particular members. In the groups in their experiment that shared all information, the correct decision was made 83% of the time, whereas if information was unshared, the correct decision was only made 25% of the time. Stasser and Titus also found that group discussion did not improve decision
quality (the rate at which correct decisions are made) - in fact, discussion tended to cause the rates of making a correct decision to plummet to 20% and 17% respectively.

It is not simple to promote information sharing in groups: Stasser et. al. [87] found that even groups with explicit instructions to improve decision making (by using a structured discussion technique), did not in fact, improve the quality of group decisions. Although it was found that prompting members to recall as much as possible from the information given did increase the amount of information discussed, this increase was predominantly due to the increase in discussion of already shared information. Even groups that were told explicitly that information should be pooled did not seem to improve the quality of the decisions made [85].

Postmes et. al. [25] reasoned that the tendency of groups to focus on shared information is similar to the phenomena that is encountered by groups exhibiting the groupthink phenomenon (see Section 2.3.4 and also [11]). As a result of this similarity to group norms, they reason that it may be possible to increase the amount of effort spent in sharing information by manipulating group norms.

4.1.1 Critical and Consensus Norm

If a group values consensus, it follows that the group should value shared information more highly. This is because shared information can be validated socially (i.e. all members can agree on the information) whereas unshared information cannot [88]. The consensus of information may be seen as contributing to the perceived correctness of the decision, so that even if all the information is shared in a discussion, the common information will be valued more highly. Conversely, a highly critical group should promote "objective" standards and individual thought, which implies that group members should treat every piece of information as equally valid [25].

4.2 Requirements Elicitation

Effective requirements definition is acknowledged to have a large bearing on final product quality [89; 90]. The requirements phase of a development project is characterised by intense communication activities [91]. Due to the need for stakeholders to both bridge the differences between background, skill, knowledge and status in order to come to an agreement on a sometimes volatile set of requirements, requirements elicitation is an activity that is acknowledged as very difficult [92].

Various stakeholders drive the requirements specifications of a new/modified system. Techniques have been proposed to elicit requirements from stakeholders including the use of interviews [93], protocol analysis [94], knowledge acquisition grid [95] amongst others. These techniques are used to extract information by providing a structured way
in which questions can be posed or interactions can be studied and subsequently coded into useful specification.

From a psychological perspective, requirements elicitation can be seen as an information sampling task: in order to develop and appropriate piece of software, all parties involved (i.e. users and developers) need to bring together and share all relevant information. Social psychology has a large body of work studying such complex collective tasks [96; 97; 98; 99; 100; 101; 102; 103; 104], where the task can be anything ranging from selecting a suspect in a murder mystery [105], developing a diagnosis for a hypothetical medical case [106] or selecting an individual for student body president [100]. Using existing social psychological approaches for these studies allowed the analysis requirements elicitation in terms of underlying, critical social psychological processes related to memory and communication.

4.2.1 Combining Norms and Requirements Elicitation

Given that a critical norm has been previously shown to promote the sharing of information and subsequently a better decision making process within groups, it may be possible to improve requirements elicitation in software groups by instilling critical norms. A critical software group would be more likely to value objective standards in discussing information. Group members would also be more likely to spend more effort in actually sharing information and treat all information as valid for consideration. In particular it may improve the inclination of stakeholders to freely share information which may actually be important but otherwise might not have been shared.

4.3 Hypothesis and effects

This second experiment was conducted to determine if norms can be identified from literature that can then be used to improve the performance of software engineering groups. The Critical norm was identified as a means of promoting information sharing, which in turn could be used to drive requirements elicitation. Thus, we state the hypotheses for this experiment below:

**Hypothesis 1:** There is no evidence to suggest that artificially instilled norms can affect information sharing in software groups.

**Hypothesis 2:** There is no evidence to suggest that critical norms offer a positive benefit to information sharing in software groups.

Hypothesis 2 can be further broken down into three sub hypotheses:
Hypothesis 2a: There is no evidence to suggest that critical norms offer a positive benefit to the sharing of common information in software groups.

Hypothesis 2b: There is no evidence to suggest that critical norms offer a positive benefit to the sharing of unique information in software groups.

Hypothesis 2c: There is no evidence to suggest that different norms do not affect the reliability of the information shared in software groups. (i.e. different norms do indeed affect the reliability of information shared software groups)

4.4 Task Characteristics

An elicitation scenario was created to stimulate a group of stakeholders to share unique information amongst themselves in specifying a software system. The system presented to participants is the LIBSYS library lending system. The participants were told that they would each be given a set of requirements for the current LIBSYS system for study, and that collectively, they are to reproduce a combined system specification so that a new version of LIBSYS can be developed as a replacement.

4.4.1 LIBSYS library system

The LIBSYS specification consisted of a set of requirements that were common, and also unique requirements that could be divided into 6 different categories. Common requirements are requirements that are presented to all members to a group, whereas unique requirements are only presented to a particular member of the group, but not any other member. The unique categories were user classes, fees and fines structures, purchase suggestions, user and borrower rights, damaged item procedures, and finally, late returns procedures.

In order to account for any possible variations of the difficulty of the requirements (such as conceptual complexity) or the variations on the amount of requirements that had to be studied, it was decided to scramble the six classes of unique requirements into 4 different sets with different combinations of requirements. Each copy of requirement requirements specification contained a common portion and 2 categories of unique information, with three separate copies making up a set of requirements. All sets of requirements, when combined, would yield the same complete set of information. The 4 different sets of unique requirements were randomly distributed to all the triads involved in the experiment (see Figure 4.1).
**Figure 4.1:** Table of LIBSYS Requirements showing Unique and Shared Information showing what unique information was assigned to each participant in each set.

<table>
<thead>
<tr>
<th></th>
<th>Set 1</th>
<th>Set 2</th>
<th>Set 3</th>
<th>Set 4</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>User Classes</strong></td>
<td>P1</td>
<td>P1</td>
<td>P2</td>
<td>P2</td>
</tr>
<tr>
<td><strong>User and Borrower Rights</strong></td>
<td>P3</td>
<td>P3</td>
<td>P2</td>
<td>P3</td>
</tr>
<tr>
<td><strong>Late Procedures</strong></td>
<td>P3</td>
<td>P1</td>
<td>P3</td>
<td>P1</td>
</tr>
<tr>
<td><strong>Asset Damage</strong></td>
<td>P1</td>
<td>P2</td>
<td>P1</td>
<td>P3</td>
</tr>
<tr>
<td><strong>Fine/Payment Amounts</strong></td>
<td>P2</td>
<td>P2</td>
<td>P3</td>
<td>P2</td>
</tr>
<tr>
<td><strong>Purchase Suggestions</strong></td>
<td>P2</td>
<td>P3</td>
<td>P1</td>
<td>P1</td>
</tr>
</tbody>
</table>

Key:
- Participant 1: P1
- Participant 2: P2
- Participant 3: P3
4.4.1.1 Task characteristics of the requirements elicitation scenario

Unitary. Due to the way a task is defined, with unique requirements available only to particular members of a triad, it is impossible to complete the task by dividing it further. All participants must be present for the entirety of the task, and participate in the specification task in order to complete the task in its entirety. As such, this task is non-divisible, and therefore referred to as a unitary task.

Judgemental. The group has to consider the available information and specify it in a coherent, complete form. For the purposes of this requirements elicitation scenario, there is no way for the groups involved to verify their specifications for correctness, nor is there a clear set of metrics to evaluate their performance for the duration of the task.

Additive. The outcome of the requirements specification task is the direct result of the summation of all of the pertinent knowledge available to each member of the group. The common and unique requirements must be combined in order to complete the task.

4.5 Participant Characteristics

The participants in this experiment were drawn from the same pool as the for Applicability experiment (see Section 3.3). These Masters students were enrolled in a requirements elicitation course at the Australian National University. The majority of the participants were enrolled in the Masters of Computing program, with the remainder in the Masters of Information Technologies Studies program. For this experiment, there were a total of 24 participants, with one female participant, and the remainder males. The subjects of this student have been enrolled for one year or less and are full time students. Since the participants were drawn from a class that included requirements specification and elicitation, it was assumed that they would have some background as to how the activity is relevant and important to system development.

4.6 Group Characteristics

The same information about group cohesiveness was collected (see Section 3.4.1). Also, since the groups were being promoted to exhibit emergent critical/consensus norms, information on the strength of the critical/consensus norms was also collected.

4.6.1 The Critical/Consensus manipulation check

In order to check the success of the critical/consensus norm manipulation, the participants answered the questions presented in Table 4.1. The manipulation check also
Table 4.1: Critical/Consensus Manipulation check [25]

<table>
<thead>
<tr>
<th>Question</th>
<th>notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>People in this group are critical</td>
<td>reverse coded</td>
</tr>
<tr>
<td>People in this group generally adjust to each other</td>
<td></td>
</tr>
<tr>
<td>I feel connected to the people in this group</td>
<td>cohesion check</td>
</tr>
<tr>
<td>I like to see myself as a member of this group</td>
<td>cohesion check</td>
</tr>
<tr>
<td>I identify with the members of my group</td>
<td>reverse coded</td>
</tr>
<tr>
<td>I am like the others in this group</td>
<td></td>
</tr>
<tr>
<td>In this group you’d think critically</td>
<td>reverse coded</td>
</tr>
<tr>
<td>In this group you ought to act independently</td>
<td></td>
</tr>
<tr>
<td>In this group you are expected to make an independent contribution</td>
<td></td>
</tr>
<tr>
<td>In this group you ought to align yourself with the opinions of other members</td>
<td>reverse coded</td>
</tr>
<tr>
<td>In this group you should conform to others</td>
<td></td>
</tr>
</tbody>
</table>

includes questions that check for the overall cohesiveness of the group to ensure that the characteristic does not become a confounding variable. Responses for the manipulation check were obtained using a nine point Likhart scale, with some questions reverse coded. Manipulation check questionnaire items were reused from the work of Postmes et. al. [25].

4.6.1.1 Critical/Consensus Priming Activity

For the purposes of this experiment, the participants were divided into two groups and then formed into triads. The triads will then be manipulated to favour particular norms through priming activities. One group of triads were primed with the critical norm, whereas the other group of triads were primed with a Consensus norms using a set up activity. This priming activity was used to establish norms through both precedence and influence (see Section 2.1.4 for more detail).

Norm manipulation by precedence. By getting groups to operate on a task that requires critical thinking before the actual activity, it sets a precedence for the subsequent activity. Group members would be expected to establish these norms via first exposure and keep them for the follow up task.

Norm manipulation by influence. The groups would be influenced to adopt norms via external influence. Depending on what group the participants were in, they would be overtly told that in order to succeed they would need to be either critical or consensusual.
4.7 Method

In this section we discuss our experimental set up including participants, demographic information, materials and procedure.

**Instilling Artificial Norms.** The first goal of this experiment was to establish if it was possible to instil an artificial norm in software engineering groups. Postmes’s [25] experiment on information sharing did not use Software engineers; therefore it may be possible that Software engineering groups could react differently to the same priming techniques.

**Using Norms to Improve Group Performance.** The second goal of this experiment was to establish if the establishment of a theoretical norm that could promote information sharing would positively enhance the performance of Software Engineering groups in a requirements elicitation task.

4.8 Participants

Participants for this experiment were drawn from the same pool of 37 Masters students as the first experiment (see Section for more detail). These individuals were formed into eight triads, with four triads assigned to the control condition and the other four in treatment condition.

4.9 Materials

Each participant was given the following:

1. A consent form and cover sheet, explaining the exercise as an experiment.

2. A handout detailing the priming activity that they were to undergo. (one for each member of the triad)

3. A sheet with priming manipulation check questions

4. A handout relating to the requirements elicitation task

5. A group environment questionnaire

6. A copy of the requirements, specially printed in sets of three for each member of the triad (see Section 4.4.1)
4.9.1 The modified Group Environment Questionnaire

Based on the guidelines in Section 3.4.1 it was decided to modify the GEQ in a similar way to the applicability experiment. The social aspects of both the Individual Attraction to Group and the Group Integration dimensions of the questionnaire were removed as the groups formed in this experiment were also short lived and only exist in the context of this experiment. The following additional rules were used to modify the GEQ questions for this experiment:

- Where the GEQ original text refers to “playing time” or any sport related activity, the text was changed to refer to the requirements elicitation activity as “the activity”.

- The original item that references communication about athlete’s responsibilities during competition was changed to be a generic question on the freedom of communication experienced by the group members during the task.

As in the first experiment, as much of the original GEQ was preserved, such as the coding format, and ordering of questions. The final GEQ is attached in Figure 4.2.

4.10 Procedure

The experimental session ran for 90 minutes as part of the last class of the semester for the participants. Similar to the letters-to-numbers experiment, the session was highly scripted in order to control for repeatability. Since the participants were split into two groups, two separate scripts were developed and pre-examined for possible bias (for experimental scripts, see Appendix C.1 and C.2). The session began with a quick introduction of the session agenda, after which the class was split into two separate groups for the rest of the experiment. In each separate group, the participants were formed into triads, who then sat together in groups of three.

The presentation for the priming exercise was an adapted form of the activity that was used to prime groups with the appropriate norms in the study conducted by Postmes et. al. [25]. Due to ethical considerations, the priming exercises were adapted to fit the class material so that the students would still receive some learning value from the experience. The triads in the critical group were told to debate a statement on requirements, and the triads that were in the consensus group were told to create a poster on what they have learned in the course. The instruction slides are attached in Figure 4.3 and Figure 4.4. Upon completion of the priming exercise, the participants were told to fill out the critical/consensus manipulation check questionnaire.
### Group Activity Questionnaire

**ID:**

This is a quick survey to see how much you liked this revision activity.

The following questions help us to assess your feelings about your personal involvement with your group. On a scale of 1 (Strongly Agree) through to 9 (Strongly Disagree), circle the number that best represents your response.

<table>
<thead>
<tr>
<th>STRONGLY AGREE</th>
<th>STRONGLY DISAGREE</th>
</tr>
</thead>
<tbody>
<tr>
<td>I am unhappy about the level of involvement I had with my group members</td>
<td>1</td>
</tr>
<tr>
<td>I am unhappy with the group’s level of desire to complete the activity</td>
<td>1</td>
</tr>
<tr>
<td>My group members did not explain their technique for approaching the system to me</td>
<td>1</td>
</tr>
<tr>
<td>I like the working style of the group</td>
<td>1</td>
</tr>
<tr>
<td>My group was united in trying to specify the system</td>
<td>1</td>
</tr>
<tr>
<td>We all took responsibility for the level of performance in the system specification</td>
<td>1</td>
</tr>
<tr>
<td>Our group had conflicting aspirations concerning our performance in the activity</td>
<td>1</td>
</tr>
<tr>
<td>Our group communicated freely with each other</td>
<td>1</td>
</tr>
<tr>
<td>Our group had a defined leader</td>
<td>1</td>
</tr>
</tbody>
</table>

Mapping of responses to variables used in analysis shown in yellow speech bubbles
Course Review

1. Discuss the following

   "Requirements Specifications should always reflect design constraints"

2. 15 minute activity
3. Anonymous
4. Used as class discussion in COMP8100 in 2011
5. Warm up revision on what you have learned
6. Share your opinions, both for and against!
Course Review

1. Design an A4 Poster
   
   "What I have learned in COMP8100"

2. 15 minute activity
3. Anonymous
4. Used as introduction to COMP8100 in 2011
5. Warm up revision on what you have learned
6. Every member must contribute!
The requirements elicitation activity was begun after a five minute break from the priming activity. The participants were told to examine the requirements individually for 10 minutes without discussion. The critical group participants were told explicitly that to be successful in the requirements elicitation task, "the information contributed by all three members of the group needs to be evaluated critically" whereas the consensus groups were told explicitly that in order to be successful, "the group must reach consensus on the system specification". In order to focus the activity, the participants were all told to ignore issues such as "document templates" and "hardware" and concentrate on "specifying as much of the system as possible in the group" and "functional requirements" (see Figure 4.5). At the conclusion of the elicitation activity, the participants were told to complete the GEQ and were thanked for their participation.

Similarly to the experiment in Chapter 3, the identities of the participants were obfuscated and not revealed to the experimenters until the analysis of results was complete. Obfuscation in this experiment was achieved by pre-filling the "ID" field of the handouts. These handouts were then shuffled and randomly distributed within the appropriate experimental group (e.g. Consensus groups received handouts marked consensus, and critical group received handouts marked critical).

4.11 Results

A triad was excluded from the final analysis because they seemed to have ignored the instructions to recreate the specifications via discussion - instead they copied all the requirements verbatim from their specification sheets. Their product specification therefore was deemed invalid for the purposes of this experiment and excluded from analysis.

For the purpose of statistical analysis, it was decided to use the median instead of the mean in order to minimize the effect of extreme values on comparisons.

4.11.0.1 Bootstrap Resampling Confidence Intervals

Bootstrap resampling was introduced by Efron [107], which samples with replacement from a given set of observations multiple times. Suppose that $C$ is the observed data with $n$ observations. From this data, bootstrap resampling obtains a large number of observations $C_1, C_2, \ldots, C_n$ by randomly sampling with replacement $n$ number of times from $C$.

A number of factors make bootstrap resampling a suitable statistical tool for comparing the medians between the Critical and Consensus triads [108]. Firstly, the number of observations obtained from this experiment is very small. Secondly, it is difficult to justify the assumption that the underlying populations of the samples are from a
Do

- Concentrate on the **functional requirements**
- Specify as much of the system as possible as a group
- **Write legibly on the provided materials**

Don't

- Ask questions; the customer is **not available**
- Focus on document templates, hardware
- Use representations that cannot be understood by laymen
normal population. Thirdly, it is difficult to justify that the populations have equal variances.

Bootstrap resampling does not make any of the above assumptions - in fact, the resampled data came exactly from the observed data, which in turn means that it mirrors the distributions from the samples taken. Instead of relying on theoretical assumptions to derive sampling distributions for statistical estimators, the bootstrap method estimates distributions empirically, using information directly drawn from the sample of observations, which allows us to avoid some of the limitations of traditional significance testing [107]. The generated samples (from the resampling process) forms the approximate distribution of the observations and then can subsequently be used for analysis, such as calculating the 90% confidence interval of data.

Four main types of bootstrap resampling techniques exist, namely the Standard Bootstrap (SB), the Percentile Bootstrap (PB), the Biased-Corrected Percentile Bootstrap (BCPB) and the Bias-Corrected and Accelerated Bootstrap (BCa). Since the BCa confidence interval has been shown to be especially suitable for non-parametric data (i.e. data that has no identified underlying distribution) [108] and it simultaneously adjusts for Bias and Skewness (when compared to the remaining resampling techniques), it was chosen as the technique for analysis. The mathematical proof for the robustness of this technique will not be covered here and the reader is encouraged to consult the text by Efron & Tibshirani [109] for more information.

### 4.11.1 Individual Characteristics

Unlike the first experiment, it was assumed that the participants had approximately equivalent levels of competency in requirements elicitation as they were all enrolled in the same course (University enrolments require a basic level of competency to be reached before course admission). No individual characteristics were recorded.

### 4.11.2 Group Characteristics

A BCa resampling procedure was performed on each response, with 99,999 resamples, on the critical manipulation check between the critical and consensus groups to compare the responses between both triads.

Our manipulation did not affect the perceptions of the participant of their triads after the normative evaluation, with the exception of C1 and C11 (see Table 4.2). The point values and ranges of the median at 90% confidence interval are shown in Table 4.2 for these items.

**C1:** People in this group are critical. Members of the Consensus triads generally felt that the members of their triad were more critical (median=4) than members of
the Critical triads (median = 9.5).

C11: In this group you conform to others. Members of the Critical triads felt less required to conform to others in their triads (median = 12.5) than members of Consensus triads (median = 6).

The GEQ was used to determine the levels of cohesion in the triads. Firstly, Cronbach’s Alpha [71] values were used to calculate the internal consistency of the items within the GEQ responses. The results are presented in Table 4.3. In this experiment, the internal consistency of the results were sufficiently high to allow the use of the aggregate ATT and INT scores, taking into account any reverse coding as specified in [70]. The aggregate scores for both the ATT and INT scores were compared using a BCa sampling procedure on the median. The results are shown in Table 4.4. The BCa resampling procedure showed that at 90% confidence, there is no significant difference between the Critical triads and the Consensus Triads in terms of cohesion.

A BCa resampling procedure was also performed on individual components of the GEQ to examine whether there was any within scale differences between the critical and consensus triads. Generally, this test also failed to find any significant differences both the critical and consensus primed triads, with the exception of the question labelled ATT1. The results are presented in Table 4.5.

ATT1: I am unhappy about the level of involvement I had with my group members. Using the BCa bootstrapping procedure with 99,999 resamples, the median models for both the Critical and Consensus were tested. The point values and ranges of the median at 90% confidence interval are shown in Table 4.5. Members of the Critical triads felt less happy about the level of involvement in their triads than members of Consensus triads.

4.11.3 Task Results

The triads elicited between 10 and 21 requirements for this activity. Triads that underwent the critical manipulation elicited between 18 and 21 requirements and triads that underwent the consensus manipulation elicited between 10 and 18 requirements. Critical triads specified between 2 and 6 unshared requirements, while consensus triads specified between 0 and 5 unshared requirements. Both critical triads and consensus triads had similar error rates (specifications that were contradictory to the original LIBSYS specification). The experimental results are visualized in Figure 4.6 and summarized in Table 4.6 and Table 4.7.

Requirements elicited was divided into three categories - shared, unshared and wrong. Shared requirements were requirements that were reproduced by the triad with
Table 4.2: Critical/Consensus Manipulation [25] check median 90% significant results on a scale of 1 to 7, with 1 as strongly agree, 7 as strongly disagree

<table>
<thead>
<tr>
<th>Question Code</th>
<th>Question</th>
<th>Critical Group median a</th>
<th>Lower ←→ Upper 90% CI</th>
<th>Consensus Group median b</th>
<th>Lower ←→ Upper 90% CI</th>
</tr>
</thead>
<tbody>
<tr>
<td>C1</td>
<td>People in this group are critical</td>
<td>9.5</td>
<td>(6.0, 14.5)</td>
<td>4</td>
<td>(3.0, 4.0)</td>
</tr>
<tr>
<td>C2</td>
<td>People in this group generally adjust to each other</td>
<td>8.5</td>
<td>(4.0, 14.5)</td>
<td>5</td>
<td>(3.0, 5.0)</td>
</tr>
<tr>
<td>C3</td>
<td>I feel connected to the people in this group</td>
<td>7.5</td>
<td>(4.0, 14.5)</td>
<td>5</td>
<td>(3.0, 5.0)</td>
</tr>
<tr>
<td>C4</td>
<td>I like to see myself as a member of this group</td>
<td>7.5</td>
<td>(4.0, 14.0)</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>C5</td>
<td>I identify with the members of my group</td>
<td>6</td>
<td>(4.0, 13.5)</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>C6</td>
<td>I am like the others in this group</td>
<td>6</td>
<td>(4.0, 11.5)</td>
<td>8</td>
<td>(4.0, 8.0)</td>
</tr>
<tr>
<td>C7</td>
<td>In this group you’d think critically</td>
<td>6</td>
<td>(4.0, 6.0)</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>C8</td>
<td>In this group you ought to act independently</td>
<td>12.5</td>
<td>(11.0, 13.5)</td>
<td>16</td>
<td>(11.0, 16.0)</td>
</tr>
<tr>
<td>C9</td>
<td>In this group you are expected to make an independent contribution</td>
<td>12</td>
<td>(5.0, 15.0)</td>
<td>11</td>
<td>(10.0, 11.0)</td>
</tr>
<tr>
<td>C10</td>
<td>In this group you ought to align yourself with the opinions of other members</td>
<td>13.5</td>
<td>(10.0, 17.0)</td>
<td>10</td>
<td>(3.0, 10.0)</td>
</tr>
<tr>
<td>C11</td>
<td>In this group you conform to others</td>
<td>12.5</td>
<td>(10.0, 15.0)</td>
<td>6</td>
<td>(3.0, 6.0)</td>
</tr>
</tbody>
</table>

aShown as a summation of all of the responses in a triad - i.e. Response = member1Response + member2Response + member3Response

bShown as a summation of all of the responses in a triad - i.e. Response = member1Response + member2Response + member3Response

cVariance on results was so small that the interval is essentially a point value

dVariance on results was so small that the interval is essentially a point value

eNo CI can be calculated as all the responses were identical

Table 4.3: Cronbach’s Alpha reliability test values for GEQ dimensions

<table>
<thead>
<tr>
<th>Items Tested</th>
<th>Cronbach’s Alpha</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATT1, ATT2, ATT3, ATT4</td>
<td>0.9</td>
</tr>
<tr>
<td>INT1, INT2, INT3, INT4</td>
<td>0.894</td>
</tr>
</tbody>
</table>
### Table 4.4: GEQ dimensions median 90% significant level comparison

<table>
<thead>
<tr>
<th>GEQ Component</th>
<th>Critical Group Median</th>
<th>Lower $\leftarrow\rightarrow$ Consensus Group Median</th>
<th>Upper 90% CI</th>
<th>Lower $\leftarrow\rightarrow$ Upper 90% CI</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATT</td>
<td>81.5</td>
<td>(77.0, 86.5)</td>
<td>79</td>
<td>(74.0, 79.0)</td>
</tr>
<tr>
<td>INT</td>
<td>26</td>
<td>(14.0, 32.0)</td>
<td>29</td>
<td>(27.0, 29.0)</td>
</tr>
</tbody>
</table>

$^a$Shown as a summation of all of the responses in a triad - i.e. $Response = member1Response + member2Response + member3Response$

$^b$Shown as a summation of all of the responses in a triad - i.e. $Response = member1Response + member2Response + member3Response$

### Table 4.5: Critical/Consensus GEQ Statistical Summary

<table>
<thead>
<tr>
<th>Question Code</th>
<th>Critical Triad Median</th>
<th>Lower $\leftarrow\rightarrow$ Consensus Triad Median</th>
<th>Upper 90% CI</th>
<th>Lower $\leftarrow\rightarrow$ Upper 90% CI</th>
</tr>
</thead>
<tbody>
<tr>
<td>ATT1</td>
<td>25</td>
<td>(23.0, 26.0)</td>
<td>18</td>
<td>(15.0, 18.0)</td>
</tr>
<tr>
<td>ATT2</td>
<td>25.5</td>
<td>(23.0, 26.0)</td>
<td>24</td>
<td>(20.0, 24.0)</td>
</tr>
<tr>
<td>ATT3</td>
<td>25</td>
<td>(21.0, 25.0)</td>
<td>24</td>
<td>(20.0, 24.0)</td>
</tr>
<tr>
<td>ATT4</td>
<td>8.5</td>
<td>(5.0, 9.5)</td>
<td>13</td>
<td>(5.0, 13.0)</td>
</tr>
<tr>
<td>INT1</td>
<td>5.5</td>
<td>(3.0, 7.0)</td>
<td>5</td>
<td>(4.0, 5.0)</td>
</tr>
<tr>
<td>INT2</td>
<td>7.5</td>
<td>(4.0, 9.0)</td>
<td>9</td>
<td>(5.0, 9.0)</td>
</tr>
<tr>
<td>INT3</td>
<td>23.5</td>
<td>(17.0, 25.5)</td>
<td>18</td>
<td>(16.0, 18.0)</td>
</tr>
<tr>
<td>INT4</td>
<td>4</td>
<td>(3.0, 4.0)</td>
<td>5</td>
<td>(4.0, 5.0)</td>
</tr>
</tbody>
</table>

$^a$Shown as a summation of all of the responses in a triad - i.e. $Response = member1Response + member2Response + member3Response$

$^b$Shown as a summation of all of the responses in a triad - i.e. $Response = member1Response + member2Response + member3Response$
Table 4.6: Requirements Elicited, Critical vs Consensus triads

<table>
<thead>
<tr>
<th>Triad</th>
<th>Number of Shared Items Specified</th>
<th>Number of Unshared Items Specified</th>
<th>Number of Wrong Items Specified</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consensus 1</td>
<td>10</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>Consensus 2</td>
<td>11</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>Consensus 3</td>
<td>15</td>
<td>2</td>
<td>1</td>
</tr>
<tr>
<td>Critical 1</td>
<td>12</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>Critical 2</td>
<td>12</td>
<td>6</td>
<td>0</td>
</tr>
<tr>
<td>Critical 3</td>
<td>16</td>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>Critical 4</td>
<td>18</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

Table 4.7: Requirements Elicited Medians

<table>
<thead>
<tr>
<th></th>
<th>Consensus</th>
<th>Critical</th>
</tr>
</thead>
<tbody>
<tr>
<td>Median</td>
<td>10 16 17</td>
<td>17 18 19</td>
</tr>
<tr>
<td>n</td>
<td>3</td>
<td>4</td>
</tr>
</tbody>
</table>

information that was available to all members of the triad. Unshared requirements were requirements that were reproduced by the triad with information that was only available to a single member of the triad. Wrong requirements were specified requirements that were invalid. Invalid requirements included requirements that were contrary to the requirements specified in LIBSYS (e.g. “All damaged material is destroyed” - this is wrong because LIBSYS includes requirements on repair of books where possible) or extraneous to the requirements (e.g. “Superuser can fine late users” - this is extraneous because although superuser can manipulate the details of the users, they cannot fine the users directly).

4.11.3.1 Resampling Results - Shared Requirements

Using the BCa bootstrapping procedure with 99,999 resamples, the median models for both the Critical and Consensus shared requirements were tested. The results of these models appear in Table 4.8, and contain the point estimates and the 90% confidence intervals around each coefficient. As can be seen from Table 4.8, there is minimal overlap between the two results.

4.11.3.2 Resampling Results - Unshared Requirements

Using the BCa bootstrapping procedure with 99,999 resamples, the median models for both the Critical and Consensus unshared requirements were tested. The results of these models appear in Table 4.9, and contain the point estimates and the 90%
Figure 4.6:! Requirements Elicited, Critical vs Consensus triads
Table 4.8: Confidence Intervals of number of Shared Requirements from Critical and Consensus groups (ranges are exclusive)

<table>
<thead>
<tr>
<th></th>
<th>point estimate</th>
<th>Lower ←→ Upper 90% CI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Critical Requirements</td>
<td>14</td>
<td>12</td>
</tr>
<tr>
<td>Number of Consensus Requirements</td>
<td>11</td>
<td>10</td>
</tr>
</tbody>
</table>

Table 4.9: Confidence Intervals of number of Unshared Requirements from Critical and Consensus groups (ranges are exclusive)

<table>
<thead>
<tr>
<th></th>
<th>point estimate</th>
<th>Lower ←→ Upper 90% CI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Critical Requirements</td>
<td>4.5</td>
<td>2.0</td>
</tr>
<tr>
<td>Number of Consensus Requirements</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

confidence intervals around each coefficient. As can been seen from Table 4.9, there is minimal overlap between the two results.

4.11.3.3 Resampling Results - Total Requirements

Using the BCa bootstrapping procedure with 99,999 resamples, the median models for both the Critical and Consensus total requirements were tested. The total requirements were calculated with the formula totalReq = sharedReq + unsharedReq. The results of these models appear in Table 4.10, and contain the point estimates and the 90% confidence intervals around each coefficient. As can been seen from Table 4.10, there is minimal overlap between the two results.

4.11.3.4 Resampling Results - Wrong Requirements

Using the BCa bootstrapping procedure with 99,999 resamples, the median models for both the Critical and Consensus wrong requirements were tested. The results of these models appear in Table 4.11, and contain the point estimates and the 90% confidence intervals around each coefficient. As can been seen from Table 4.11, the two results overlap.

Table 4.10: Confidence Intervals of number of Total Requirements from Critical and Consensus groups (ranges are exclusive)

<table>
<thead>
<tr>
<th></th>
<th>point estimate</th>
<th>Lower ←→ Upper 90% CI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Critical Requirements</td>
<td>19</td>
<td>17</td>
</tr>
<tr>
<td>Number of Consensus Requirements</td>
<td>16</td>
<td>10</td>
</tr>
</tbody>
</table>
Table 4.11: Confidence Intervals of number of Wrong Requirements from Critical and Consensus groups

<table>
<thead>
<tr>
<th></th>
<th>point estimate</th>
<th>Lower ←→ Upper 90% CI</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Critical Requirements</td>
<td>0.5</td>
<td>0</td>
</tr>
<tr>
<td>Number of Consensus Requirements</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

4.12 Insights from Analysis

**Hypothesis 1 is rejected.** Based on the results shown in Section 4.11.3, we can say that there is sufficient evidence to show that artificially instilled norms can affect information sharing in software groups.

**Hypothesis 2a is rejected.** Based on the results shown in Section 4.8, since the number of shared requirements is significantly more for critically primed groups than consensus groups, we can say that critical norms do indeed offer a positive benefit to the sharing of common information in software groups.

**Hypothesis 2b is rejected.** Based on the results shown in Section 4.9, since the number of unshared requirements is significantly more for critically primed groups than consensus groups, we can say that critical norms do indeed offer a positive benefit to the sharing of unique information in software groups.

**Hypothesis 2c is rejected.** Based on the results shown in Section 4.11, since the number of wrong requirements is not significantly different between critical and consensus groups, we can say that different norms do not affect the reliability of information shared in software groups.

**Hypothesis 2 is rejected.** In sum, since Hypothesis 2a, 2b and 2c are all rejected, we can reject Hypothesis 2. Further, the combined shared and unshared requirements counts were significantly different between the Critical and Consensus Triads, with Critical Triads having produced more requirements (see Table 4.10). We can say that critical norms offer a positive benefit to information sharing in software groups.

4.12.1 Critical and Consensus Norm Manipulation Do Not Affect Significantly the Cohesiveness of Software Groups

Based on the results obtained from Table 4.5, there is insignificant difference between the cohesiveness metrics on the GEQ responses from the critically and consensus primed triads. This indicates that the manipulation that was used did not affect the goodwill or otherwise the within group camaraderie of software groups. Triads that were critically
primed, for instance, did not experience greater within group animosity due to the more
critical environment that was fostered on them. Likewise, triads that were consensus
primed did not experience greater within group likeability due to the norms that were
adopted.

We can surmise therefore, that this particular normative manipulation, particularly
the introduction of critical or consensus norms, did not affect the cohesiveness of groups.
Cohesion did not play a role in the performance of the triads under study, and therefore
was not a confounding factor of triad performance in the task.

Within the GEQ, one item showed significantly different responses between the
Critical and Consensus primed triads (ATT1).

**ATT1: Critical triad members were less satisfied with the level of invol-vement in their triad when compared to Consensus triads.** This dissatisfaction may be an indicator of the Critical norm that was imposed - Critical triads accept and tolerate differences in perception, opinion and knowledge. The differences in points of view between the triad members would result in members not being able to fully obtain acceptance from other triad members, resulting in a feeling of diminished involvement. Contrary to this, the positive validation of shared information in Consensus triad would allow triad members to fully achieve shared acceptance, which would increase the perception of involvement within the triad.

### 4.12.2 Normative Manipulation Check Metrics

Based on the results obtained from Table 4.2, there is insignificant difference between
most of the questions used to verify the success of the normative manipulation. The
only questions to show significant difference was C1 and C11. Although this indicates
that statistically, the normative manipulation was not successful, we argue that this is
not the case.

Firstly and most importantly, the number of requirements of shared and unshared
requirements were significantly different between the two groups, indicating that de-
spite the lack of statistical difference on the items in the critical manipulation check
questionnaire, there is a distinct difference in the performance of the Critical and Con-
sensus triads.

Secondly, the results of this manipulation check corroborates the findings of Postmes
et. al.'s study, from which it was derived [25]. Using this same manipulation check,
Postmes et. al. also did not find any significant difference using the same questionnaire
in the original study. However, they also found that the manipulation did indeed create
a positive difference between the Critical and Consensus groups within their study.

Thirdly, in the items that did show a significant difference, there are indications
that the normative manipulation was successful. We cover these items in order.

**C1:** Critically primed triads felt that their fellow triad members were not critical enough; Consensus primed triads felt that their fellow triad members were too critical. Group norms are the accepted behaviour among group members, and it appears that the triads had successfully been manipulated to highly value either criticality or consensus in their respective cases. Since the triads have been told that, in their respective cases, criticality or consensus was essential in task success, this led to the perception that extreme levels of criticality or consensus was required. In turn, this led to the counter-intuitive finding of critical triads feeling that their fellow triad members were not critical enough, and consensus triads finding that their fellow triad members were not consensus enough.

**C11:** Critically primed groups felt less inclined to conform to the opinions of their fellow triad members when compared to members of Consensus primed triads. This perception is also consistent with group norms that have been introduced to the respective triads. It is more acceptable for Critically primed triad members to hold uncommon opinions and share this position with their fellow triad members - they would feel less pressured or inclined to adopt a conformed position or reach consensus when compared to Consensus primed triads.

### 4.12.3 Cohesiveness, GEQ and manipulation

The GEQ responses showed that critical and consensus triads did not show significantly different levels of cohesion. Therefore, we can say that the critical and consensus manipulation undergone by the respective triads did not significantly affect the cohesiveness within the triads. Based in the results of the individual questions of the GEQ, we can also surmise that the different triads also did not significantly differ in the types of productive norms that they would have instituted. We can therefore remove team cohesiveness as a confounding variable in this experiment.

### 4.12.4 Critical and Consensus Manipulation: Manipulation Check Issues

Although our manipulation check produced insignificant results, these findings do indeed corroborate with the manipulation checks conducted in the experiment by Postmes et. al. [25]. The study by Postmes et. al. showed a positive bias for sharing previously unshared information by critically primed group; In the experiment conducted on Software groups there are indications that this bias is present as well, even though the sample size is too small to show statistical significance.

The experiment was conducted in a class setting due to ethical constraints. Subse-
quently, it is unclear that the manipulation check questions referred to the perception of the class environment "group" or the transient triads that the students were placed in. As part of the requirements analysis course that was conducted, the participants were repeatedly drilled with the phrases "it depends" and "as long as it adds value", which has a strong implicit critical thinking emphasis. This may have carried over to the perceptions of the other triad members in the experiment, distorting the results and thereby affecting the internal validity of the manipulation check metrics.

4.12.5 Triads primed with critical norms shared more common information

Comparing the performance of the critically primed triads to the performance of consensus primed triads in the experiment, critically primed triads specified more shared information in general than the consensus primed triads. Therefore, we can say that it is possible that critical priming promotes better sharing and specification of common information than a consensus prime, and that the priming (see Section 4.12.3) does not substantially affect the normal operations of group intra-relationships.

4.12.6 Triads primed with critical norms shared more unique information

Comparing the performance of the critically primed triads against the performance of consensus primed triads in the experiment, critically primed triads specified more unique (or unshared) information in general than the consensus primed triads. Therefore, we can say that it is possible that critical priming promotes better sharing and specification of member unique information than a consensus prime, and that the priming does not substantially affect the normal operations or the intra-relationships of group members.

4.12.7 Critical and Consensus primed triads did not have noticeably different error rates

Comparing the performance of the critically primed triads against the performance of consensus primed triads in the experiment, neither triads showed significantly different rates in specifying contradictory or non-existent requirements. Therefore, we can say that neither priming activity noticeably increases the rate of error in the triads.
4.13 Limitations and Threats to Validity

Critical/Consensus Norm Measurement Internal Validity. We have been unable to show a significant difference between the measures of critical and consensus norms using the results of a measurement questionnaire. The results, however, seem to indicate that the critically primed triads do indeed share more information than the consensus primed triads as predicted by theory. These results corroborate those of Postmes et. al. [25], who were also unable to show statistical significance in those measures. In spite of this, Postmes et. al. also showed, similar to our experiment, in their experiments that as predicted by theory, critical groups do share more information than consensus groups. However, unlike Postmes et. al., we did see a difference in a handful of metrics used: C1 and C11 metrics (see Section 4.12.2).

It is possible that one of the reasons that affected the critical/consensus norm measurement instrument was that of the context - as mentioned in section 4.12.4, the potentially ambiguous (due to experimental setting) word “group” was used, where it would have been more appropriate to use the word “triad” instead for clarity. However, even though the metrics for the checks were insignificant, the results were still consistent as predicted by Postmes et. al. [25].

Small Sample Size. The experiment was conducted with a total of 8 triads, of which the results from 1 triad had to be discarded. Although with bootstrapping, we have shown statistical significance in the difference in performance, it must be noted that caveats to do with analysing small sample sizes could still apply. Therefore, for software groups that have the same characteristics as our participants, the results should hold for performance in a similar type of task.

Convenience Sample VS True Population Sample. The sample used in this experiment was a convenience sample of Masters students enrolled in a requirements engineering course. Whilst it may be possible that this convenience sample is a typical sample of the general population of Software Engineers, we have not proven this to be the case. Similarly to the small sample size limitation covered above, it is possible that this sample is biased in some way, and therefore the generalisability of the results of this experiment should be limited to groups that share a similar profile to the triads studied.

Specificity of Work. This experiment studies the difference between critical group norms and consensus group norms in requirements elicitation and specification, and therefore the results should be limited to this activity, or activities that share similar characteristics.

Group Lifetime. The Group Environment Questionnaire was designed for be applicable to teams with a long timet ime (i.e. over multiple sports seasons). Although the
§4.14 Discussion

In the first experiment, we showed that, in spite of the lack of agreements to what type of people Software Engineers are, it is possible that Software Engineers in groups do indeed exhibit the same sorts of characteristics as any other group for particular tasks. As might be expected (from other group types), Software Engineers put into groups organically develop norms. Further, in our experiment, software groups that developed norms consistent with those identified as increasing productivity also performed better than other software groups. Given the relationship between conducive norms and productivity, we followed up our experiment with another to see if we could deliberately and explicitly manipulate the formation of norms predicted to aid the performance of groups in some task.

Our experiment showed that critical triads do indeed specify more requirements than consensus primed triads. This increased information sharing was positive for both shared and unshared information. Additionally, the priming that was performed did not seem to affect the cohesion of members, or the formation of other normative behaviour.

The normative manipulation in this experiment was performed by a simple, integrated task that fit within the environment of the experiment. The participants were told to perform tasks with the priming instructions that could conceivably have been presented naturally as part of the course. In addition to being unobtrusive, the priming tasks used were also of a short duration.

For future work with small samples, particularly the replication of this particular experiment: in addition to applying the bootstrapping technique for significance testing in small sample sizes, the reader is advised to collect ancillary data to further bolster the strength of the results. For instance, data on actual conversations could be collected via electronic media and a coding system for supporting the shared and unshared number of requirements could be developed and applied.
4.15 Related Work

Our work in this experiment is related to work in 3 areas, namely social norms, groupthink and software requirements.

Social Norms. This work is closely related to the work undertaken by researchers studying social norms [24; 23; 19; 21; 20; 25; 10; 110; 86; 85]. The pertinent related literature in this area is focused on how group norms are developed and enforced, how group norms affect tasks such as decision making, and also how groups share and manage information. In contrast to our work, however, the research is done on generic groups [25], whereas our work aims to see if the theories and predicted effects from the area of Social Psychology can be harnessed and replicated in software groups.

Groupthink. The area of research focused on groupthink attempts to identify why groupthink happens, and the mechanisms involved that promote it [59; 111; 56; 11; 112; 61; 57]. In contrast, this work attempts to use some of the results and recommendations from that research in order to prevent the formation of groupthink in software groups, particularly in tasks that require the sharing of information.

Software Requirements. Research into the area of Software Requirements looks at the different techniques that can be used to capture requirements from stakeholders [113; 114; 94; 95], how to identify stakeholders [115; 92; 116] and how to specify requirements [93]. Our work looks at promoting the sharing of requirements by selecting and implementing conducive norms.

4.16 Conclusion

This experiment in normative manipulation shows that it is possible to explicitly introduce particular norms to Software groups which can then affect productivity in certain types of tasks. Depending on the type of task characteristic, it may be beneficial to introduce a norm that promotes certain types of actions in members in order to increase the chance that a more successful outcome can be obtained from the group. In particular, it was shown that information shared and subsequently used by groups is improved for tasks that require this characteristic, such as requirements specification. Tasks that also require information sharing such as planning, or rapid prototyping could possibly also benefit from the establishment of such norms.

This finding has immediate relevance to software engineering. In practice, it is difficult to identify the “perfect” individuals to work together in a group. Even if such individuals can be identified, there are other factors that may come into play that would limit the formation of such groups, such as limited resources or personal commitments. The luxury of creating this ideal group from scratch, therefore, can be viewed as a
rare occurrence that is enjoyed by few. In contrast, it may be possible to nudge an existing group by implementing norms that are conducive to the task being performed to increase the chance of obtaining a better result. In addition, the finding shows that a short, simple, integrated and organic task may be used for normative manipulation, which would minimize disruption and resistance to the activity and yet produce the required results.
Study 3: Adoptability

The final part of this thesis is to show that the technique of normative manipulation is adoptable.

This thesis makes the claim that normative manipulation is unobtrusive and can be performed in an active group of Software Engineers in industry. Therefore, the experiment presented in this chapter attempts to answer the following questions about the normative manipulation techniques in a real, operating software group through an experiment:

- Did the participants feel that the process was intrusive (i.e. disruptive/unnatural/uncomfortable/irrelevant)?
- Did the participants feel that the process motivated themselves to share information?
- Did the participants feel like that they shared relevant and enough information in the process?
- Was any collected data useful, and did this data yield any insights into groups and the environment?

The experiment was conducted as part of a task to elicit requirements for visualisation and tool support for assembly language programmers. As part of the requirements elicitation task, normative manipulation was performed on the participants, with the aim to increase the amount of information shared during the requirements elicitation session. Participants in the task were employees of a large international software company who supported the company’s mainframe products coded in assembly language.

The structure of the experiment comprised of activities at points shown in Figure 5.1. First, the group was profiled through the administration of the Need for Closure scale and the Individualistic/Collectivistic scales. Second, the results obtained from these scales were used to select the norms that were to be introduced to the participant group to increase information sharing through a priming activity. Observation
data was collected during the requirements elicitation activity, and then finally, an exit survey is administered.

The focus of the experiment is not on the effectiveness of Normative Manipulation, which was covered in Chapter 4, but rather, the focus is on how well normative manipulation was received by an active software group in industry. The requirements elicitation process was used as a driver for examining the adoptability of the normative manipulation technique, and the technique's intrusiveness in a real industry situation. A total of three consecutive half days were used for the process. Normative manipulation was carried out as part of the entire requirements elicitation process, and not as a separate, discrete activity. The participants were also not made aware that normative manipulation was carried out for the duration of the process.

The reason for choosing requirements elicitation for this experiment is twofold. Firstly, as covered in Section 4.2, requirements elicitation has a large bearing on final product quality, and continues to be relevant to the Software Engineering field. Secondly, the use of requirements elicitation allows us to reuse conclusions obtained from the experiment in Chapter 4.

5.1 Norm Manipulation Calibration

Current research shows that particular types of group characteristics serve to promote particular outcomes. For instance, whilst collectivistic attitudes increase intra-group cooperation [117], it also runs the risk of reducing innovation and unique thinking [118].

Manipulation of group norms, however, is not as straightforward as simply implementing the theoretical ‘best’ norm for the outcome. Take for instance creativity. It may be possible to argue that individualistic groups would be expected to be more creative than collectivistic groups. On the other hand, since collectivistic groups tend to respond better to norm manipulation [117], it may be that collectivistic groups respond better to normative manipulation and in turn become more creative. However, Goncalo & Staw [119] show that individualistic groups in fact are more creative than collectivistic groups when given the same instructions, if creativity is the goal of the manipulation.

Before performing any normative manipulation, a profile of group characteristics was obtained. First, we examine the group involved in the requirements elicitation task in terms of their job roles, interactions, knowledge distribution, tasking, task allocation and task assignment using interviews and surveys. Then, the Need for Closure scale (NFC) and the Individual-Collectivistic (IC) scale is administered.
Figure 5.1: Experiment 3 Setup

**Requirements Elicitation Task**
- Contact with Task Group
- Observation of Individuals
- Group Elicitation Activity
- Requirements Prototype

**Case Study Activities**
- Group Profiling
- Priming Activity
- Observation Data
- Exit Survey

**Key:**
- In Scope of Study
- Requirements tasks
- Case Study Activity Points
5.1.1 Need for Closure

The Need for Closure Scale (NFC) was proposed by Kruglanski [120]. It is defined as a need to obtain “an answer on a given topic, any answer... compared to confusion and ambiguity”. It covers two aspects - firstly, it pertains to the need to obtain closure quickly, and secondly, the need to maintain this closure. Individuals high in NFC prefer order, structure and predictability. These individuals also have an urgent desire to reach swift decisions and experience discomfort when faced with ambiguity. They also tend to be closed-minded and do not appreciate having their opinions challenged [121]; they are prone to groupthink effects and stereotyping.

A reduced and remapped 15 item NFC scale was administered in order to provide some insight into how suggestions or even the requirements process may be received by the existing staff. The 15 item NFC scale was based on a shortened and verified version of the original developed by Roets & Van Hiel [122].

5.1.2 Individualistic/Collectivistic Measure

Collectivism of a group binds and mutually obligates individuals, whereas individualism is commonly defined as the opposite of this phenomenon [123]. Collectivistic groups are communal and share common fates, common goals and common centralized values; In such groups, the person is regarded as a component of the group, and therefore the key unit of analysis is that of the group [124]. In contrast, in individualistic groups, relationships and group memberships are simply used to attain self-relevant goals; the individual is not bound by, or to, the group and weighs the cost of membership to the benefits conferred [123]. The measure used for Individualism and Collectivism is that proposed by Oyserman et. al. [123]. It consists of 15 total items which are in turn made up of eight Individualistic statements and seven Collectivistic statements.

5.2 Hypothesis and effects

This third experiment was conducted to determine if normative manipulation can be carried out in an unobtrusive manner in the field. Thus, we state the hypothesis for this experiment below:

Hypothesis 1: There is no evidence to suggest that individuals in groups that are undergoing normative manipulation would not notice any irregularities in the process. (i.e. individuals in groups would notice irregularities in a process due to the introduction of normative manipulation)
5.3 Task Characteristics

The main task for this experiment is requirements elicitation, and in this experiment, we're concerned mainly with any assembly programming that was undertaken by the participants in the course of their jobs. The focus was to identify any areas where visualisation or development tools would aid productivity or ease of work in the area of assembly programming. The participants were explicitly told the purpose of the experiment and to focus in particular on their assembly language development or maintenance process.

5.4 Participant Characteristics

The group studied is part of a large multinational company employing more than 13,000 staff. The company develops software products that are used to manage complex mainframe, distributed, virtualised or cloud environments. The group studied consists of seven co-located members working on assembly language programming on the company's mainframe product. Every member of the group is a long time employee in the area of relevance.

5.5 Group Characteristics

The GEQ used in the previous experiments was not used in this particular experiment. It was decided that measuring the cohesiveness of the group would not be useful as there was no baseline to use for comparison. However, the descriptions of job roles and interactions were obtained from a questionnaire in order to obtain an idea as to how much interaction was required within the group.

5.6 Method: Group Profile

In this section we discuss our experimental set up with regards to obtaining a group profile which is then used to decide the norms that are to be introduced to the experimental group.

5.6.1 Group Profile: Participants

All seven members of the assembly programming group mentioned above participated in the group profile. The participants are very experienced and all have worked in the group for a significant number of years.
5.6.2 Group Profile: Materials

Each participant was given an electronic key that can be used to access a questionnaire online generated using a survey package. For the purposes of profiling, this questionnaire was administered with a double-blind set up. The survey package used allowed participants to do the questionnaire anonymously. The package also subsequently collated the results for the experimenters for analysis.

Questions for each individual participant were randomised. A sample questionnaire is presented in Appendix D.

The survey administered consisted for the following sections:

1. Expertise.
2. Roles, Team Structure and Knowledge Distribution.
3. Tasks, Responsibility and Interactions.
4. Modified Need for Closure Scale.
5. Modified Individual/Collectivistic Scale.

5.6.2.1 Modifying the Need for Closure Scale for the Experiment

The shortened 15 item NFC was modified in order to bring it closer to the context of the experiment. A requirements elicitation expert was consulted which indicated that participants in the software engineering field are more likely to be receptive to surveys worded as questions instead of statements. Furthermore, the presentation of the NFC had to match the format of the other material administered to participants so far in the experiment.

The modified 15 items were derived in three steps. First, two experimenters examined the original items and developed a mapping key that was relevant to the context of the participants. The mapping key was developed by mapping the original items to development specific contexts (e.g replace all mentions of the generic “situation” with “requirements”). Secondly, a requirements elicitation subject matter expert was also told to tailor to the best of her ability, the original questions to the context of the experiment. Finally, both transformations of the NFC were compared and a new mapping key was developed that incorporated the mappings that were deemed the most appropriate. It was also decided to change the format of the items in the NFC items from statements to questions. Some questions became reverse coded during the mapping process, that is, a highly positive response indicated a low NFC response instead of a high NFC response. In addition, on consultation with the requirements subject matter expert, both statement 4 (containing the word “uncomfortable”) and statement
5 (containing the word "irritated") were modified to use a milder form ("concerned") remove what were viewed as strong expressions in the original questions.

The final statements are shown in Table 5.1. Each question was ranked by participants on a six point Likert scale, with 1 being the most negative response possible to the question, and 6 being the most positive.

5.6.2.2 Modifying the Individualistic/Collectivistic Measure for the Experiment

The shortened 15 item IC was also modified in order to bring it closer to the context of the experiment. A requirements elicitation expert was consulted which indicated that participants in the software engineering field are more likely to be receptive to surveys worded as questions instead of statements. Furthermore, the presentation of the IC had to match the format of the other material administered to participants so far in the experiment.

The modified 15 item items were derived in three steps. First, two experimenters examined the original items and developed a mapping key that was relevant to the context of the participants in the experiment. The mapping key was developed by mapping the original items to development specific contexts (e.g. replace all mentions of the generic "situation" with "requirements"). Secondly, a requirements elicitation subject matter expert was also told to tailor to the best of her ability, the original questions to the context of the experiment. Finally, both transformations of the IC were compared and a new mapping key was developed that incorporated the mappings that were deemed the most appropriate. It was also decided to change the format of the items in the IC items from statements to questions.

The final statements are presented in Table 5.2. Any reverse coded items in the original were retained and remain as reverse coded items in the modified questions. Each statement of the Individualistic-Collectivistic (IC) scale was administered on a six point Likert scale, with the lowest score being the most negative response possible, and six being the most positive. For reverse coded questions, a higher scoring response indicates tendency towards collectivism instead of individualism as normal.

5.7 Group Profile: Procedure

The survey electronic key was emailed to each participant a week before norm manipulation and requirements elicitation was scheduled to take place. Each participant

† For these questions, the question has been reverse coded in the questionnaire. i.e., if the response is 6 (strongly agree), then the correct response when calculating the total score should be reversed as 7 - 6 = 1.
<table>
<thead>
<tr>
<th>Original</th>
<th>Modified</th>
</tr>
</thead>
<tbody>
<tr>
<td>I don’t like situations that are uncertain(^a)</td>
<td>What is your opinion of requirements that are uncertain?</td>
</tr>
<tr>
<td>I dislike questions which could be answered in many different ways(^a)</td>
<td>What is your opinion of coding tasks which could be implemented in many different ways?</td>
</tr>
<tr>
<td>I find that a well ordered life with regular hours suits my temperament.</td>
<td>Do you find that a well ordered development routine helpful?</td>
</tr>
<tr>
<td>I feel uncomfortable when I don’t understand the reason why an event occurred in my life.</td>
<td>Are you concerned when you don’t understand the reason why code behaved unpredictably during development?</td>
</tr>
<tr>
<td>I feel irritated when one person disagrees with what everyone else in the group believes</td>
<td>Are you concerned when a team member holds a unique opinion from everyone else in the team?</td>
</tr>
<tr>
<td>I would quickly become impatient and irritated if I would not find a solution to a problem immediately.</td>
<td>Would you quickly become impatient or irritated if you would not find a solution to a coding problem immediately?</td>
</tr>
<tr>
<td>I don’t like to go into a situation without knowing what I can expect from it(^a)</td>
<td>What is your opinion of being assigned a requirement to implement without knowing what you can expect from it?</td>
</tr>
<tr>
<td>I don’t like to be with people who are capable of unexpected actions(^a)</td>
<td>What is your opinion of working with a team that lacks process?</td>
</tr>
<tr>
<td>When I have made a decision, I feel relieved.</td>
<td>Do you feel relieved when you have made a decision on how to implement a requirement?</td>
</tr>
<tr>
<td>I dislike it when a person’s statement could mean many different things(^a)</td>
<td>What is your opinion of when a team member’s technical information is ambiguous?</td>
</tr>
<tr>
<td>When I am confronted with a problem, I’m dying to reach a solution very quickly.</td>
<td>When confronted with a coding problem, do you need to reach a solution very quickly?</td>
</tr>
<tr>
<td>I find that establishing a consistent routine enables me to enjoy life more</td>
<td>Do you find that establishing a consistent routine enables you to enjoy development more?</td>
</tr>
<tr>
<td>I enjoy having a clear and structured mode of life</td>
<td>Do you enjoy having a clear and structured development process?</td>
</tr>
<tr>
<td>I do not usually consult many different opinions before forming my own view(^b)</td>
<td>Do you consult many different opinions before forming your own view?</td>
</tr>
<tr>
<td>I dislike unpredictable situations(^a)</td>
<td>What is your opinion of unpredictable requirements?</td>
</tr>
</tbody>
</table>

\(^a\) *Original item is not coded, but modified question adds a reverse coding in the phrasing*

\(^b\) *Original item is reverse coded, modified question retains this property*
<table>
<thead>
<tr>
<th>Original</th>
<th>Modified</th>
</tr>
</thead>
<tbody>
<tr>
<td>I tend to do my own thing, and others in my family do the same.</td>
<td>Do you tend to work individually, and do others in your team do the same?</td>
</tr>
<tr>
<td>I take great pride in accomplishing what no one else can accomplish.</td>
<td>Are you proud of accomplishing tasks that others have not accomplished?</td>
</tr>
<tr>
<td>It is important to me that I perform better than others on a task.</td>
<td>How important to you is competitive spirit in a team?</td>
</tr>
<tr>
<td>I am unique - different from others in many respects.</td>
<td>Are your work habits or skills different from others on your team?</td>
</tr>
<tr>
<td>I like my privacy.</td>
<td>Do you value work privacy?</td>
</tr>
<tr>
<td>I know my weaknesses and strengths.</td>
<td>Do you know your weaknesses and strengths?</td>
</tr>
<tr>
<td>I always state my opinions very clearly.</td>
<td>Do you always state your opinions clearly?</td>
</tr>
<tr>
<td>To understand who I am, you must see me with members of my group.</td>
<td>How easily can your work be understood independently from the work of others on your team?</td>
</tr>
<tr>
<td>To me, pleasure is spending time with others.†</td>
<td>Do you enjoy working with others?</td>
</tr>
<tr>
<td>I would help, within my means, if a relative were in financial difficulty.†</td>
<td>Would you help others with their work tasks, within your means?</td>
</tr>
<tr>
<td>Before making a decision, I always consult with others.†</td>
<td>Do you consult other team members before making an decision on how to implement a requirement?</td>
</tr>
<tr>
<td>How I behave depends on who I am with, where I am, or both.†</td>
<td>Do you approach problems differently depending on the team, the project or both?</td>
</tr>
<tr>
<td>I have respect for authority figures with whom I interact.†</td>
<td>Do you respect the work decisions made by those senior to you?</td>
</tr>
<tr>
<td>I make an effort to avoid disagreements with my group members.†</td>
<td>Do you make an effort to avoid disagreements with group members?</td>
</tr>
<tr>
<td>I would rather do a group paper or lab than do one alone.†</td>
<td>Would you rather work on a task as part of a team than alone?</td>
</tr>
</tbody>
</table>

† Original item is reverse coded and modified question retains this reversal in the phrasing.
was instructed to complete the survey to the best of their ability. No time limit was imposed for this questionnaire, and participants could save incomplete sessions for resumption later. A reminder email was also sent to all participants a few days after the initial email with the same keys.

5.8 Group Profile: Results

This section presents the results obtained from the group profiling questionnaire.

5.8.1 Expertise

On a scale of one to six, with one being ‘novice’ and six being ‘expert’, all participants rated themselves as five or better on ‘reading and understanding assembly language’, and ‘implementing features or maintaining assembly language’.

5.8.2 Roles, Team Structure and knowledge distribution

The group is divided into two types of employees: the first consisting of three employees as implementers that focus on creating new products and software, and the remainder as maintainers focusing on responding to customer issues and issuing fixes and patches.

Roles - In addition to formal appointments as Software Engineers or Maintainers, some of the group have stated that they fill additional secondary roles. One of the implementers performs “maintenance & testing”. One of the maintainers fulfils a “quality assurance and test[ing] program[s] to reproduce issues” and another maintainer performs “trouble-shooting”. These three responses are taken to consist of the same general nature consisting of issue reproduction and issue analysis. One of the maintainers indicated that they are the sole responsible person for a single product that was inherited by the present company when it bought over another company. This single member was taken to also perform issue reproduction and analysis but in contrast to the other three maintainers who were responsible for majority of the work of the group, his activities were restricted to only a single product.

Team Structure - The participants were asked how the team was structured. They were to choose from managerial centred (wherein project leader acts as a central clearing house processing and coordinating the work of each engineer, while each engineer works ignorant of the larger whole) or team centred (wherein completing the work depends more on the group than any particular one engineer, and where much of the work is done collectively) or expertise centred (wherein engineers depend on each other to coordinate the work based on their areas of specialized knowledge). Out of the seven
participants, six felt that the team was structured in an expertise-centred way, whereas one felt that it was structured in a team-centred way.

**Knowledge Distribution** - Participants were also asked how the knowledge in the team was distributed, in terms of modules in the system. Four participants felt that members of the team had overlapping knowledge of modules, whereas the remainder felt that there was no overlap. The majority (five) of the group members felt that there was distinctive module ownership, while the remainder felt that there was no ownership at all. Two of the three implementers indicated that they felt that there was a person with knowledge that overlaps all modules, while the remainder responded in the negative. In general, we can say that the team knowledge is distributed among specialists, with responsibilities centred among particular individuals, likely the specialists themselves. A very small number of individuals (likely one) is a system wide expert that has information that overlaps many responsibility and specialist areas. This view is confirmed by site observations, with one particular engineer filling a “consultant-type” role and offering aid to engineers who encounter any issues in their own areas of work.

5.8.3 Tasks, Responsibility and Interactions

The participants were asked to describe a recent task they were asked to complete, and if appropriate, the sequence of steps that were taken in order to resolve it. The responses are included below to allow the reader a glimpse into some typical operations undertaken by members in this group. Participants were also asked to give percentages of time they spent on particular segments of tasks where possible. The responses include references to internal programs and short hands. This tasking information was taken at face value from the participants, no confirmation via observation was made. The text used for analysis is provided in Appendix E.

The responses from the participants were analysed with the following objectives:

- **Work differences** - What are the differences in the type of work undertaken in assembly language programming by implementers and maintainers?

- **Approach differences** - What are the differences (if any) in the approach of undertaking assembly language programming by implementers and maintainers?

- **Time differences** - What are the differences (if any) in the way time is spent during the assembly language programming task by implementers and maintainers?

**Work Differences** - The participant responses were scanned for phrases that indicated that the participants were talking about working with existing code. Phrases alluding
to modifying and extending an existing code base (such as "customer enhancement request" or "old code does not support...") were defined as maintenance work, regardless of whether the task was performed by an implementer or maintainer. Implementation was defined as working on a totally new feature that had not previously existed in the product, for example, "implement SSM compliance across all products starting with Solve:OPS".

Of the seven participants interviewed, only one response could be interpreted as a totally new feature. Regardless of the job role indicated by the participant (Implementer or Maintainer), the majority of the work involved modifying and extending an existing code base.

**Approach differences** - The participant responses were examined to understand the sequence of steps that were taken in approaching the task. The following areas were of particular interest: (a) predominant task types, (b) subtask types that were common among all participants, and (c) predominant areas of expertise revealed by the way the participants approached the problem.

The approaches by all the participants share the following broad categories of subtasks - (1) Finding (characterised by words in the interview such as "finding", "identify" and "research", amongst others), (2) consulting (characterised by words such as "discuss", and "consulted"), (3) Implementation/Testing (characterised by phrases that are describe coding and testing) and (4) Base lining (characterised by phrases that indicate that the code is sent off to either internal or external parties for further testing). Of the seven participants, the "finding" subtask is evident in all seven participant responses. The remaining subtasks all appeared in five responses each.

**Time differences** - The participant responses were scanned for extreme responses that indicate that the participant either spent the entire process working alone, or with another team member. In general, from the responses, the main part of the task is indeed completed alone, although consultations and interaction are common (evident in five of seven responses). The participants were also asked to list the amount of time that they spent on various interactions with other members. The results are presented in Table 5.3 and Figure 5.2.

On the whole, the participants in this organisation tend to spend the majority of their time working alone, although there is still a significant amount of the remaining time spent on interactions with other engineers.

The participants were asked how their tasks were assigned, how task decisions are
### Table 5.3: Percentage of time spent in interactions

<table>
<thead>
<tr>
<th>Group Member ID</th>
<th>Managerial</th>
<th>Engineer</th>
<th>Group Meetings</th>
<th>Pair Programming</th>
<th>Alone</th>
<th>Other</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implementor 1</td>
<td>0%</td>
<td>10%</td>
<td>0%</td>
<td>10%</td>
<td>80%</td>
<td>0%</td>
</tr>
<tr>
<td>Implementor 2</td>
<td>10%</td>
<td>10%</td>
<td>10%</td>
<td>0%</td>
<td>50%</td>
<td>20%↑</td>
</tr>
<tr>
<td>Implementor 3</td>
<td>1%</td>
<td>5%</td>
<td>4%</td>
<td>2%</td>
<td>88%</td>
<td>0%</td>
</tr>
<tr>
<td>Maintainer 4</td>
<td>0%</td>
<td>10%</td>
<td>10%</td>
<td>0%</td>
<td>80%</td>
<td>0%</td>
</tr>
<tr>
<td>Maintainer 5</td>
<td>10%</td>
<td>10%</td>
<td>10%</td>
<td>0%</td>
<td>60%</td>
<td>10%↑</td>
</tr>
<tr>
<td>Maintainer 6</td>
<td>3%</td>
<td>10%</td>
<td>1%</td>
<td>1%</td>
<td>85%</td>
<td>0%</td>
</tr>
<tr>
<td>Maintainer 7</td>
<td>4%</td>
<td>10%</td>
<td>3%</td>
<td>0%</td>
<td>75%</td>
<td>8%</td>
</tr>
</tbody>
</table>

### Figure 5.2: Graphical Representation of time spent in interactions
made and if there is any approval process that is undertaken. This information was taken at face value from the participants, no confirmation via observation was made. The text used for analysis is provided in Appendix F.

The responses from the participants were analysed with the following objectives:

- **Origin of Work** - Does the work come from an internal source or an external source?

- **Independence** - Does the participant have the ability to decide on how the task is performed, and is there any need to explicitly validate their work with others?

**Origin of Work** - The interview responses were analysed for phrases that indicate the source of the sample assignment. For instance, phrases such as “management” or the word “manager” were taken to mean that the work originated from an internal source. Of the seven participants, six participants have tasks assigned internally via a manager, while one participant indicated that he assigns tasks to himself.

**Independence** - The interview responses were analysed for phrases that indicated that the participant required excessive supervision to make changes to the code, above that of any general process model. This extra supervision that can be characterised by checks or any validation that imply that the participant was reliant on interaction with their peers, instead of mainly being independent and adhering to process. Phrases that related to process such as “documenting”, or any phrases deemed related to development tools (e.g. “STARTRACK system” were determined to be relevant to process and discarded as supervision related activity. Where the word “discussion” was used, it was decided to treat that as part of a clarification process for work, and therefore not part of a supervisory requirement.

All of the seven participants had responses that indicated that they enjoyed independence in carrying out a task. Four out of the seven participants had responses that alluded to process mechanisms, which could indicate also that in general, development is not ad-hoc and a process exists in the group.

### 5.8.4 NFC Scale

The NFC results are collated and shown in Figure 5.3. The scores presented for each individual item in the figure are the raw scores - any reverse coded items do not have the responses corrected.

The overall score for NFC is the summation of all the scores per respondent divided by the total score possible (i.e. \( \frac{\sum_{i=1}^{NFC-15} i}{NFC-1} \div 6 \times 15 \times 100 \)). The overall score is
computed with the corrected scores of each individual response. For instance, if a question is reverse coded then the corrected score is applied by applying the formula $final\text{score} = 7 - response$. For any score above the midpoint of $6 \times 5 \div 2 = 45$ (or 50%) the respondent is generally regarded as higher on the NFC scale. In general, we can say that the respondents generally score above average (see Figure 5.4) on the NFC scale.

5.8.5 IC Scale

The results are shown in Figure 5.5. The results shown in the graphs for each individual question are the raw scores - any reverse coded items do not have the responses corrected.

The totals for each of the respondents are computed and then plotted on a percentage graph with the formula $\left(\sum_{i=IC-1}^{IC-15} i\right) \div 6 \times 15 \times 100$. For this overall score, the corrected scores are used (i.e. if a question is reverse coded, then the response is corrected before it is added to the formula (using the equation $corrected\text{score} = 7 - response$). A score of more than 50% indicates a tendency toward individualism, while a score below 50% indicates a tendency towards collectivism. The more extreme the scores, the more the respondent tends towards a particular aspect (higher percentage is more individualistic, lower is more conformist). A boxplot of the results is shown in Figure 5.6. In general, we can say that the respondents generally score on the individualistic side on the IC scale.

5.9 Group Profile: Conclusion

Based on the results obtained from the IC scale (see Section 5.8.5), the group in general is individualistic and also slightly higher on the need for closure than truly neutral (see Section 5.8.4). Individualistic groups that are explicitly told to be more creative generate more ideas [119]. Therefore, it was decided to encourage a critical thinking norm in order to promote the sharing of information in a group (see Chapter 4) and also to avoid promoting an environment of groupthink and stereotyping [121].

5.10 Method: Adoptability

In this section we discuss our experimental set up with regards to the adoptability of the normative manipulation technique that was subsequently applied based on the results obtained from the group profile.
Figure 5.3: Raw Need For Closure results.
The process for examining the adoptability of normative manipulation can be broken down into the following sequential component phases:

1. Normative Manipulation phase, administered via exercise sheets.
2. Guided elicitation process with discussion.
4. Exit Survey

In addition, the guided elicitation process was videotaped and analysed for any negative reactions by participants of the process.

### 5.10.1 Interaction Process Analysis

The Interaction Process Analysis (IPA) is a method that was developed by Bales [126] for the study of small groups. The interaction Process Analysis can be used to derive a set of empirical generalizations about participant behaviour.

The technique consists of observers rating the interactions of participants according to a scale and a set of simple rules. The observer codifies each response from each participant to a against 12 categories. The twelve categories are grouped into four broad types of reactions: (A) Positive Reactions, (B) Attempted Answers, (C) Questions and
Figure 5.5: Raw Individual-Collectivistic Measure Results.
(D) Negative Reactions. Each of the 12 categories are also paired to produce six distinct sets of problems that may promote responses that fall into the codified categories. The categories, reactions and problems are shown in Figure 5.7.

The rules for codifying responses are presented below (reproduced from Bales [126]):

1. The act to be scored is the smallest discriminable segment of verbal or non-verbal behaviour to which the observer, using the present set of categories after appropriate training, can assign a classification under conditions of continuous serial scoring. (Normally an uttered sentence)

2. View each act as a response to the last act of the last other, or as an anticipation of the next act of the next other.

3. Favour the category more distant from the middle. Classify the act in the category nearer the top or the bottom of the list.

Each act directly influences the rules for codifying the subsequent act. The full guidelines for classifying responses are provided in Bales [126]. The reader is encouraged to follow up on this publication for more details as the classification rules are complex and will not be covered here.

For the purposes of this experiment, the interactions of the participants were first recorded on video and then the IPA was applied by the experimenter.
Figure 5.7: Interaction Process Analysis Categories - reproduced from Bales [126]

1. Shows solidarity, raises other's status, gives help, reward:
2. Shows tension release, jokes, laughs, shows satisfaction
3. Agrees, shows passive acceptance, understands, concurs, complies:
4. Gives suggestion, direction, implying autonomy for other:
5. Gives opinion, evaluation, analysis, expresses feeling, wish:
6. Gives orientation, information, repeats, clarifies, confirms:
7. Asks for orientation, information, repetition, confirmation:
8. Asks for opinion, evaluation, analysis, expression of feeling:
9. Asks for suggestion, direction, possible ways of action:
10. Disagrees, shows passive rejection, formality, withholds help:
11. Shows tension, asks for help, withdraws out of field:
12. Shows antagonism, deflates other's status, defends or asserts self:

KEY:
a Problems of Communication
b Problems of evaluation
c Problems of Control
d Problems of Decision
e Problems of Tension Reduction
f Problems of Reintegration
A Positive Reactions
B Attempted Answers
C Questions
D Negative Reactions

5.11 Adoptability: Participants

The same participants that were present in the group profiling experiment were present in this experiment, although only five out of the seven participants responded to the Exit Survey.

5.11.1 Adoptability: Materials

Each participant was given a priming and individual brainstorming exercise to complete, followed by an electronic key that can be used to access the exit survey. The exit survey was generated by the same survey package used in the group profile experiment and allowed a double-blind experiment to be conducted. The exit survey questions were randomised for each participant.

The priming and individual brainstorming exercise is presented in Appendix G, and a sample exit survey questionnaire is presented in Appendix I.

5.12 Adoptability: Procedure

This experiment was carried out over three consecutive half-days. The first half-day consisted of on-site interviews and observations. At the end of the first half-day, the participants were given a priming and brainstorming exercise to complete. The second half-day consisted of a guided elicitation exercise, which was carried out in the presence of the entire group. The final day was used for an exit survey and also for clarification. All contact with the participants was recorded on video.

5.12.0.1 Interviews and Observation

The first half-day session comprised of in-situ observations of two participants. The first subject observed was a Maintainer, and the second subject was a Software Engineer. Both sessions and interviews were videotaped. The participants were told to work normally on an assembly programming task - experimenters then used body language cues to ask question about the process of working on assembly programming in order to identify any areas where visualization or tool support would be useful. The requirements elicited is not in the scope of this experiment and will not be covered here.

5.12.0.2 Normative Manipulation

The normative manipulation and brainstorming exercise was administered via a hand-out to all participants at the end of the first half-day session. The participants were
instructed verbally, and also via written instruction, to complete the forms in order and to the best of their ability before the requirements elicitation activity. The participants were instructed to complete the priming and brainstorming exercise throughout the rest of the day while they were doing any assembly programming work, and before the guided elicitation exercise the next day. In the event the participants did not have any assembly programming scheduled, they were told to complete it to the best of their ability, while revising their experience with their most recent assembly programming task.

The normative manipulation consisted of a series of questions adapted from those used by Goncalo & Staw [119]. The Individualistic Manipulation questions are as follows:

1. Write three statements describing your particular area(s) of expertise (i.e. topics that other team members consult you on).

2. Write three statements about how your area(s) of expertise does not overlap with that of other team members.

3. Write three statements about why you think it is advantageous for individuals to have an area of expertise.

The critical thinking manipulation consisted of a single exercise as follows:

- Critically comment on one particular tool, does not matter which one, which you use in the course of your work (e.g. how effective it is, or how its effectiveness could be improved). please provide at least three comments.

The full handout is presented in Appendix G.

5.12.0.3 Guided Elicitation Exercise

The guided elicitation exercise was based on the nominal group technique [127; 128](previously covered in section 2.3.1). The process that was conducted in the presence of all the participants, as a group and videotaped.

The videotape of the activity was first transcribed, and then the Interaction Process Analysis 5.10.1 was applied.

The guided elicitation process itself is not in the scope of this experiment and will not be covered here.
5.12.0.4 Exit Survey

The exit survey was administered to the participants using the same survey platform as the questionnaire in Section 5.6.2. The exit survey focused on the experiences of the participants during the whole process, as well as how well the process was received and followed.

The exit survey is presented in Appendix I.

5.13 Adoptability: Results

The seven participants that completed the questionnaire attended the guided elicitation exercise. The exit survey, however, was completed by only five participants. We present below first the results from the IPA (see Section 5.10.1), then the results of the exit survey, which measures the receipt of the normative manipulation process. In particular, the exit survey focused on how well the participants performed the normative instructions as presented in the priming and individual brainstorming exercise, and how the participants felt about the quality and quantity of communication during the guided elicitation exercise.

5.13.1 Interaction Process Analysis Results

The Interaction Process Analysis was used to codify the participant responses into the 12 categories specified. The IPA was applied to recorded video of the Requirements Elicitation Process. The codified responses are shown in the following Figure 5.8. The largest percentages of the interactions are in the category “Gives Opinion” at 30.9%, followed by “Shows Tension Release” at 16.1%. No interactions were codified in the final categories of “Disagrees”, “Shows Tension” or “Shows Antagonism”.

In general, we can say that most of the interactions by participants are related to “Gives Opinion”, or sharing information with the rest of the group.

The interactions were also divided into 5 intervals of approximately 12 minutes each. These intervals were then analysed with respect to the four broad types of interactions to see if any patterns could be observed. The results are shown in Figure 5.9.

The majority of the interaction types were in group (B) Attempted Answers (57%), followed by group (A) Positive Reactions (31%) and group (C) Questions (12%). There were no (D) Negative Reactions. In addition, there appears to be a trend of increasing interactions of (B) Attempted Answers over time. For (A) Positive Reactions, there appears to be an increase from Interval 1 (9 interactions) to Interval 2 (22 interactions), after which the number of Positive Reactions remain approximately constant (between
Figure 5.8: Interaction Profile of All Participants as a Percentage of all Interactions
Figure 5.9: Interaction Type against Number of Responses against Interval

Interaction Type against Number of Responses against Interval

A: Positive Reactions
B: Attempted Answers
C: Questions
D: Negative Reactions
22 and 27 interactions per Interval). For (C) Questions, the number of interactions remains approximately constant between 6 and 11 per interval.

In general, we can say that the majority of the interactions were of the interaction type “Attempted Answers”, which increased over the duration of the requirements elicitation process. Also, after an initial period of time, there was a significant amount of “Positive Reactions” from the participants in the process. There were no “Negative Reactions” by the participants during the entire period of the requirements elicitation process.

5.13.2 Questionnaire Results

The questionnaire results are used to drive the normative manipulation that was applied on the participants. We divide the results into various sections related to the area of interest. First, we cover a profiling of sorts of the participants, including how knowledge is distributed in the team, and how tasks are assigned and carried out, either in isolation or in groups. Then, we present the metrics collected on the Need For Closure scale and the Individual-Collectivistic scale.

5.13.3 Exit Questionnaire

The respondents were handed an exit questionnaire after the requirements elicitation process was over. The purpose of the Exit Questionnaire was to determine if the participants followed the normative manipulation instructions and also how they felt about the entire requirements elicitation process.

The questionnaire contained a section on the norm manipulation process, followed by participant opinions on the quality and quantity of communication, and finally a free response section for comments.

5.13.3.1 In order instructions

To determine how comfortable and intuitive it was to follow the instructions in order, the exit questionnaire asked “Were you able to fill out the yellow sheets [priming activity] before beginning the exercise in the blue sheets [requirements elicitation activity]?”. Of the five responses received, two participants indicated they did not. The first said that “I hadn’t done any assembler work on the day I was to fill it out. The instructions indicated that the sheets were to be filled out while doing assembler work.”, while the second responded “3 statements... exceeded my ability for at least one of the questions”.

Although two respondents were not able to complete the priming section before the requirements elicitation activity, only 1 respondent (“The instructions indicated..”) conceivably did not attempt the priming section before the requirements elicitation.
Table 5.4: Quality and Quantity of Communication Poll (1 as extremely insufficient, 6 as extremely sufficient)

<table>
<thead>
<tr>
<th>Question</th>
<th>Participant 1</th>
<th>Participant 2</th>
<th>Participant 3</th>
<th>Participant 4</th>
<th>Participant 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>Was there sufficient communication during the activity? (Quantity of Communication)</td>
<td>3</td>
<td>4</td>
<td>5</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>Was there sufficient communication during the activity? (Quality of Communication)</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td>5</td>
</tr>
</tbody>
</table>

Therefore, on the whole, participants did attempt and consider the instructions in the priming section before the activity. In addition, there were not any comments that indicated that participants felt adversely to the priming instructions.

5.13.3.2 Quality and Quantity of Communication

Participants were polled on how much they felt the quality and quantity of communication was affected in the requirements elicitation exercises. Their responses are shown in Table 5.4.

The quantity of communication had a mean and median response of 4 (sufficient) and the quality of communication had a mean response of 4.2 (better than sufficient) and a median of 5 (very sufficient). Therefore, in terms of quality and quantity of communication we can say that the participants felt that it was more than sufficient for this activity.

In addition, the respondents were asked if they felt motivated to share their personal opinions, and why. The responses are shown in below.

- **Respondent 1** - “*Yes I did, I said most what I had in mind at the moment, time allowing.*”
- **Respondent 2** - “*Yes - hoping something good may come of it.*”
- **Respondent 3** - “*Yes, the discussions that took place drew me in.*”
- **Respondent 4** - “*I did.*”
- **Respondent 5** - “*Yes, no concerns with sharing my opinions.*”

In general, we can say that the respondents felt comfortable in sharing their opinions and information during the elicitation activity.
5.13.3.3 Participant Comments

The respondents were asked if they had any comments about the entire process. One respondent indicated that he did not always understand what was required. Even so, he filled in all the sheets and participated in all the activities. Another respondent was regretful that he turned up late and had to play catchup.

In terms of process, the respondents were also quizzed explicitly on if they felt the process was useful for the organization. The responses are below:

- **Respondent 1** - “I do not think there is any benefit for OUR organisation in this exercise. It will not change the ways we do things in Assembler, as it was not the objective of the activity. There, I hope, is the benefit for YOUR organisation. The question is abigous [sic] anyway."

- **Respondent 2** - “Drew out people’s ideas and if management invested some effort in doing some things that were suggested then that would be a welcome change, regardless of the value of any subsequent tools.”

- **Respondent 3** - “Possibly. Some of the ideas presented could be very useful if implemented.”

- **Respondent 4** - “Not really - there were no managers present to pick up on issues and drive them forward. And techos [sic] are too busy in their own little worlds to attend to big picture stuff.”

- **Respondent 5** - “Yes, some points were brought up which otherwise would not have been so. Communication of problems of others can spark ideas for new innovation.”

Three out of five responses can be classified as positive. Of the remaining two, one response was mainly concerned with the fact that the activity may not be followed up on - indicating that the respondent felt that the process was useful for the organisation if further work was done. In general, therefore, four out of five respondents felt that the process used for elicitation might potentially be useful for the organisation if follow up work was done.

The respondents were then asked if they found the process useful for themselves. The responses are below:

- **Respondent 1** - “Not really, other than state few opinions which are known to my peers anyway."

- **Respondent 2** - “A little bit. Being interviewed helped me to think about my work.”
• **Respondent 3** - “Yes. [Some of the ideas presented could be very useful if implemented].”

• **Respondent 4** - “Yes. It made me focus on tools for development - used to be one of my hobbies.”

• **Respondent 5** - “Yes, some points were brought up that would not otherwise have been so.”

Four out of the five respondents felt that the process was useful on a personal level. And out of the five respondents, two felt that very relevant unshared information was covered.

The respondents were then asked if there were any ways in which the process could be improved. Three respondents responded in the negative, while Respondent 1 stated “The assumptions that we code Assembler every day is not correct, we do other things during our office hours: research, documentation/help writing, code design, technical discussions, meetings, e-mail, code testing, fix raising and testing, etc.” and Respondent 2 stated that “[I was] not sure what was wanted on the sheets for the meeting”. Overall, with the exception of Respondent 1, who felt neutrally towards the entire experience, the other Respondents felt positive.

Finally, the respondents were asked if they felt if some part of the process could be adopted within the group. While Respondent 1 and 5 replied negatively, Respondent 2 said that “Yes - we have put very little effort into process improvement”, Respondent 3 said that “We should have more discussions on how to improve our assembler programming environment.” and Participant 4 said that “Yes - any part really - we do too little in the way of process improvement.”.

### 5.14 Insights to Analysis

**Hypothesis 1 is rejected.** Based on the comments obtained in Section 5.13.3.3, the participants indicated that they were motivated to share their personal opinions. This result is corroborated by the observations in Section 5.13.1 where not only are the majority of the interactions by the participants of type “Gives Opinion”, it was also shown that the interactions of type “Attempted Answers” increasing over the time of the requirements elicitation process.

Further, based on the responses shown in Section 5.13.3.3, the participants found the process useful for themselves and the organisation in general. This result is also corroborated by the findings in Section 5.13.1, where no negative reactions were identified, and where the second largest interaction type by category were of the type “Positive
Reactions”. Finally, throughout the entire process, the participants did not notice any oddity of the process, and did not comment on or challenge any normative manipulation instruction. We therefore reject the hypothesis and state there is evidence to suggest that individuals in groups that are undergoing normative manipulation would not notice any irregularities in the process.

5.15 Study Limitations and Threats to Validity

This experiment was conducted with the aim of examining if profiling and the subsequent normative manipulation can be done non-intrusively on a practising software group. As such, it faces the following limitations and threats to validity.

5.15.1 External Validity

Although this group is a practising software group, it may still be atypical or inapplicable to some other practising software groups. For instance, since this group comprised of experienced older developers in a highly formal team environment, it may not be as effective on groups comprised of younger, less experienced and less formal developers.

Further, this elicitation was for the development of tools that were completely new to the operating environment of the group, and not simply the elicitation of enhancements to existing tools in use. The nature of open ‘unknown’ improvements could possibly be very different to simple enhancements or additions.

5.15.2 Internal Validity

Process - The threat to internal validity lies within the process of this experiment. As part of the experiment, participants were videotaped. In addition, they were aware of observers for the purposes of requirements elicitation. Therefore, it may be possible that this would introduce a Hawthorne Effect [129] to the conclusions of this experiment - namely participants performing ‘better’ than expected simply because they are more aware of observers. Recently, however, the Hawthorne effect itself has come under contention [130; 131].

In this experiment, we addressed the observation-inhibition problem (also known as Social Facilitation in the area of Social Psychology) with the following steps:

1. Although the participants would like to have a positive outcome, they were not directly involved with the success of the activity.

2. The participants did not receive, at any time, judgemental feedback.
3. The participants indicated that they were used to meetings of this nature informally.

These steps are in line with the findings of Klehe et. al. [132] - participants were not exposed to a high pressure environment which would have had either a promotional or inhibitory effect on their performance in the requirements elicitation task.

**Survey Transformation** - Another threat to internal validity lies in the use of the NFC and IC measures. Both scales used in this experiment were transformed from their original form and this transformation may have weakened the validity that have been previously been attributed to the scales. Further, the modified scales were not verified.

In this experiment, we addressed the scale transformation issue by creating a mapping key, and producing two independent transformations of the original scales. The two independent versions were then compared and the version that was deemed closest to the original version was chosen. In some cases, a mix of both versions was chosen as the final version.

### 5.15.3 Experimenter Bias

It is possible that experimenter bias would play a major role in skewing the results obtained in this experiment. This issue is addressed by having double blind responses, and also using a coding scheme for responses to determine the level of positivity in the results. In addition, all interaction with the participants are scripted clearly before they happen and adhered to strictly as part of the protocol for experimentation. In addition, the experiment described in this chapter was conducted as part of a larger experiment to elicit requirements for visualisation and development tools. Further, the nature of the experiment described in this chapter was not revealed to the participants, minimizing the participants anticipating responses that the experimenters were looking for.

### 5.15.4 Experimenter Reliability

The use of the IPA is normally paired with a test for observer reliability in order to account for the reliability of the codification process. In this experiment, however, due to NDA restrictions, it is not possible to cross check the reliability of the scale with another observer. There are two main issues that affect the reliability of IPA as applied by an observer. First, the observer may not be consistent, leading to similar interactions being coded into different categories. Secondly, the observer may not be correct in classifying interactions of a particular type into a particular category.
In this experiment, the experimenter attempted to mitigate the consistency issue by first transcribing actions from the videotape. This transcription was done with multiple passes to ensure that actions that were similar were coded into similar categories. Since a videotape was used, it was possible to revisit portions of the recording to compare participant reactions at different points in time to ensure consistency. The transcription was then used for coding the participant responses. For each coding, the guide provided in the appendix of Bales [126] was closely consulted to determine the appropriate category of the interaction transcribed. Since for this experiment the actions by the participants were all clearly covered by the rules provided by Bales [126], we believe that the correctness of the codification is largely intact.

5.16 Discussion

In the first experiment (see Chapter 3 we showed that Software Engineers do exhibit the same sorts of characteristics as any other group for some particular tasks, in spite of the lack of agreement in the literature as to the precise type of people that are Software Engineers. We also showed that Software Engineers put into groups develop norms organically, and groups that had developed norms associated with higher productivity indeed performed better at the experimental task, as predicted by theory.

In the second experiment (see Chapter 4), we showed that, as predicted by theory, specific norms can be introduced to groups to increase performance in specific tasks.

The third experiment attempts to address some of the threats to validity that were faced by the first two experiments, by focusing on its adoptability. Firstly, instead of Masters students at a University, the participants were a real, practising Software Engineering group. Secondly, it focuses on how the process of manipulating norms, namely, applying and adapting scales in order to build a group profile, selecting a norm to implement in a group that would improve the performance in a task, and implementing it in a non-intrusive manner. In this respect, this third experiment appears to be successful - none of the engineers voiced any suspicion on the process, nor did they feel uncomfortable or pick up on any non-ordinary happenings.

A part of the experiment that is of note is that some of the adapted questions appear to bring out some differences (albeit insignificant) in attitude due to roles. It also appeared to bring out a particular organizational norm with regards to how much work is performed individually. Further examination of roles and attitude differences is outside the scope of this work, although it may be an area that should undergo future examination. In spite of this, we include the results, comparison and analysis of the differences of the responses between the Software Engineers (referred to here as Implementers) and Maintainers.
5.16.1 Implementers vs Maintainers

Although it may seem from Section 5.8.3 that the division of group members into maintainers and software developers to be largely one of title and less important in practice, some of the metrics collected in the NFC and IC scales (see Section 5.1.1 and 5.1.2) indicate there may be some difference in the attitudes held by participants fulfilling different roles. It is important to note here that the small sample size does not provide the statistical power to generalise these results, nevertheless we provide all the data here that may indicate differences for discussion and consideration.

5.16.1.1 NFC-13: Do you enjoy a clear and structured development process?

Maintainers and Implementers differed on the 13th question on the NFC scale. The question asked was “Do you enjoy having a clear and structured development process?”. The implementers on the whole answered more positively than maintainers, although this difference was not statistically significant. The responses are shown in Figure 5.10.
5.16.1.2 IC-3: How important to you is competitive spirit in a team?

Maintainers and implementers showed different response patterns on the 3rd question on the IC scale. The question asked was "How important to you is competitive spirit in a team?". In general, implementers showed a larger spread of responses than maintainers to this question. The responses are shown in Figure 5.11.

5.16.1.3 IC-4: Are your work habits or skills different from others in your team?

Maintainers and implementers showed different responses to the 4th question on the IC scale. The question asked was "Are your work habits or skills different from others in your team?". In general, implementers answered more positively that maintainers to this question, although this difference was not statistically significant. The responses are shown in Figure 5.12.

5.16.1.4 Response differences between Maintainers and Implementers - A Discussion

The minor differences in the responses can possibly be explained by the factors of "nature of work" and "group organisation". Nature of work can be used to possibly
Figure 5.12: IC-4: "Are your work habits or skills different from others in your team?"
Implementers vs Maintainers distribution boxplot

explain the differences of NFC-13 and IC-3, while group organisation can possibly be used to explain the differences of IC-4.

Nature of work - For maintainers, the work involved in implementing changes and fixes is more ad-hoc than implementers, who have indicated that they generally have some direction for the features that were required to be implemented. Therefore, this difference in attitudes between implementers and maintainers on having a structured process (NFC-13) could possibly be attributed to this difference in job description. In addition, since the experimental group consisted of long serving group members, job preference could also play a role in explaining this difference. It may be possible that individuals that are not comfortable with a reactive work attitude would be less willing to accept a long term job that consisted of work that is less structured and would therefore be less likely to remain as maintainers.

Nature of work may also possibly be used to explain the differences between the attitudes of implementers and maintainers on competitive spirit (IC-3). The large spread of opinion on implementers on this question indicates that perhaps the attitudes that the individual holds is not self-selecting or has any bearing on the group performance or group norm. In contrast, the tight cluster of responses of maintainers on the neutral response indicates that highly competitive maintainers may not be effective when working on customer fixes perhaps due to customer interaction issues. Low competi-
tive spirit maintainers may also be undesirable as they may not exhibit the required urgency for particular critical customer issues.

**Group organisation** - Although both implementers and maintainers answered positively for the question on the difference on work habits (IC-4), implementers seemed to answer even more positively than maintainers in this regard. This difference could possibly be due to the way the group is structured on products. Some participants have indicated that they have product specialization and areas of expertise of which they are responsible. It is possible that there are maintainers that function as a “catch-all” point of contact for issues, leading them to feel that they have a more similar skill set to other members in the group than in practice.

### 5.17 Related Work

Related work to this experiment can be classified into three broad areas: Ethnographic Studies of Software Engineers performing various tasks, Requirements Elicitation and Normative Manipulation. These areas are covered in turn below.

**Ethnographic Studies of Software Engineers performing various tasks** - Various ethnographic studies have been conducted to examine the behaviour of Software Engineers in various tasks, activities and processes. Tasks such as copying and pasting in programming [133], activities such as development [134] and even full processes eXtreme Programming methods [135] have been studied. Further, ethnography has also been used to derived requirements of systems such as that of a flight control system [136] and air traffic control system [137]. In this experiment, we use some ethnographic techniques to observe the participants in situ, in order to build up a basis for normative manipulation, in order to aid requirements engineering. The ethnographic technique was used as a tool in order to perform effective normative manipulation in the group, which would then improve the occurrence of information sharing, and therefore indirectly, the requirements elicitation process.

**Requirements Elicitation** - In addition to the use of ethnographic techniques for requirements elicitation above, various other techniques have been proposed to improve the outcomes of requirements elicitation activities such as Protocol Analysis [94] and the Knowledge Acquisition Grid [95]. In terms of this experiment, we did not use any of these techniques in order to minimize the appearance of confounding variables.

**Normative Manipulation** - Norm manipulation and its effects on task performance has been studied in the field of Social Psychology. Postmes et. al. [25] examined the effects of critical and consensus norms on information sharing. Myer [138] studied how a perceived performance norm affects task performance. However, these studies are largely conducted on Psychology undergraduates. In the experiment covered in this


chapter, normative manipulation was performed in an industrial setting, in order to
gauge the obtrusiveness and acceptability (i.e. *adoptability*) of this technique.

**Interaction Process Analysis** - Interaction Process Analysis is used in the area of So-
cial Psychology to examine group interactions with respect to various activities such as
dispute resolution [139], discussions [140], persuasion [141] and task performance [142].
In the area of computer science, the Interaction Process Analysis has been used to
examine online and face-to-face processes [143], Computer Mediated Communication
systems [144; 145] and technical review meetings [146]. In terms of this experiment, we
use the Interaction Process Analysis to obtain a metric of satisfaction and opinion shar-
ing during the requirements elicitation process, which is different from a methodical
breakdown of interactions to derive models for a particular activity as IPA is generally
used for in Social Psychology [139; 140; 142]. The use of IPA in this experiment
is similar to the use of IPA in Computer Science, where it is used as a metric - to
categorise the types of interactions for a particular activity in order to obtain a general
overview of what goes on when a group performs a particular task.

## 5.18 Conclusion

The purpose of this experiment is to experiment the intrusiveness of the Normative
Manipulation technique, and not the effectiveness of the technique, which was covered
in Chapter 4. Although ideally, it would be very desirable for us to present not only
the results of the adoptability of the technique, but also the success of the normative
manipulation, the results of the requirements elicitation is not due for a significant
number of years.

From this work, we can draw the following conclusions on process intrusiveness and
participants attitudes to information sharing:

In general the comments received from the participants about the process from the
exit questionnaires were positive, although there was a neutral response (although this
was more due to how he felt that the responses would not be followed up on) and a
single negative response. The majority of the participants (five of seven) also took
time off from their weekday to complete an exit questionnaire. Therefore, we conclude
that although we cannot say that the process was a resoundingly positive experience,
nor entirely streamlined into participant workloads to promote complete response, the
process was not overwheleingly negative or intrusive. None of the participants had
any negative comments with regard to the normative manipulation, nor picked it up as
an extraordinary or suspicious activity that should not have been part of requirements
elicitation.

The comments obtained from the participants that attempted the exit survey with
regard to information is overwhelmingly positive (see 5.13.3.1). Even assuming that the participants who did not attempt the exit survey in fact felt negative about the process and chose no longer to participate, we are left with a majority positive outcome. Opinions and ideas did not seem to be impeded, and one participant even indicated that he felt increasingly drawn into the discussion over the session. Participants also indicated that they felt motivated to share what they were thinking.

In terms of the original questions posed for the experiment, we can say the following in general:

- The participants did not feel that the process was intrusive.
- The participants felt that the process did motivate them to share information.
- The participants felt that they shared relevant and enough information in the process.
- The responses collected did yield some insight into participant roles and participant responses to particular responses in the NFC and IC scale, although a larger and more statistically significant study would need to be done to examine this further.

This finding has immediate relevance to software engineering. In practice, the administration of normative manipulation techniques could be met with issues ranging from being unwieldy and resource consuming to being contrived. We showed in this experiment that not only did the participants react positively to the activities related to profiling and requirements elicitation, they also did not find that the process was in any way abnormal or uncomfortable. In particular, we showed the applicability of this technique on a real, practising software group. Therefore, it may be possible that normative manipulation can be performed in a non-intrusive way on long lived, existing groups.
Synthesised Analysis

In this chapter, we revisit the thesis of this research together with the cumulative results of all the three studies that were conducted. We will discuss the three parts of the thesis in turn, starting with applicability, followed by effectiveness and finally adoptability.

6.1 Applicability

The first part of the thesis of this research is that Software Engineering group behaviour can be predicted by Social Psychology theory.

In the first experiment, we showed that Software Engineering groups do indeed perform better than individuals on a letters-to-numbers task, as predicted by Social Psychology research. In addition, we also showed that, in line with predictions from Social Psychology research, Software Engineering groups that adopted norms consistent with better groups performance also performed better than the groups that did not have such norms.

In the second experiment, we showed that Software Engineering groups that adopted the critical norm did indeed perform better than Software Engineering groups that adopted the consensus norm with regards to a requirements elicitation task, as predicted by Social Psychology research. We defined performance by the number of specified shared and unshared requirements. Critical groups specified more shared and unshared requirements than consensus groups. Further, we showed that, as predicted by Social Psychology research, normative manipulation can be achieved by a sequence of steps (priming) that consisted of exposure to a precursor activity.

6.2 Effectiveness

The second part of the thesis of this research is that performance of a task by Software Engineering groups can be improved by normative manipulation.
In the second experiment, we showed that the critical norm groups elicited more shared and unshared requirements than compared with a consensus norm group, without affecting the number of errors that were generated. A requirements elicitation task was chosen as it was a real task that has to be performed by Software Engineers, and requirements elicitation has been shown to have a large bearing on the resulting quality of the system.

6.3 Adoptability

The final part of the thesis of this research is that normative manipulation should be unobtrusive manner.

In the second experiment, normative manipulation was performed by a simple, integrated task that fit within the environment of the experiment. The participants were told to perform tasks that could possibly have been presented as part of the course that they were undertaking. In addition, the tasks were of a short duration.

In the third experiment, normative manipulation was conducted on the participants of an experiment for visualization and developer tools for assembly language programming. The participants were to participate in a requirements elicitation activity that would specify and drive the design of better development tools and visualization tools to aid their work. The participants remained positive throughout the entire process, and did not pick up on the normative manipulation, showing the unobtrusiveness of the technique in an industrial context.

6.4 Summary

Chronologically, the layout of this thesis shows, in order, the applicability, effectiveness and adoptability of social norms.

This research makes the following contributions:

6.4.1 Proof of Generalizability of Social Psychology research

Although Software Engineering research does not agree on the profile of typical software engineer, we show that this may not affect the application of normative manipulation on Software Engineering groups. In addition, Social Psychology does not explicitly test of the applicability of normative manipulation on typical Software Engineering groups, but we show that Software Engineering groups may perform similarly as predicted by Social Psychology research in some tasks.
6.4.2 Applicability of Normative Manipulation to Improve Task Performance

In this research, we show firstly that there exist some tasks in which a group of Software Engineers may outperform individual Software Engineers, regardless of inherent ability. Secondly, we show that within those groups, particular norms may be correlated to increased performance. Finally, we show that we may directly manipulate the performance of groups in a requirements elicitation task by influencing the norms adopted by groups.

6.4.3 Normative Manipulation is Non-Intrusive

The industrial group which underwent normative manipulation did not pick up on any oddity nor feel resentful of the process. On the whole, the participants felt that they were motivated to contribute, and the process was helpful for eliciting requirements. Therefore, we say that normative manipulation, when applied to practising software groups, may be non-intrusive.
"When a team outgrows individual performance and learns team confidence, excellence becomes a reality."

- Joe Paterno, Football Coach (1926 - )

It is not always clear why groups perform the way they do on tasks. We know that as software systems increase in size, it becomes less likely that the entire development cycle is undertaken by a lone individual. Today, software systems are developed and delivered by groups of people working together. Together, these assembled individuals often have the combined technical abilities to complete the task set for them. However, the failure rates for software systems seem to suggest otherwise [147] - that there exists, potentially, other variables that predict the success of groups.

The motivation of this dissertation is to examine the possibility of improving software development from the human perspective. In the area of tools, processes, techniques and equipment, software development continues to improve substantially. However, arguably the largest and most important area [2] that affects the success of software development continues to lag behind [5]. In order to address this deficiency, some work has been done by various researchers in characterising Software Engineers [30; 31; 26; 38], but the research results do not agree [28; 37; 27]. It has been argued that the reductionist approach taken by researchers into this area is not as effective as an integrative approach [40]. This research is focused on the application of an integrative approach in social psychology research (group norms) to improve the chances of success of software engineering groups when they approach tasks.

The thesis of this research is threefold. First, there exists techniques and research from the area of Social Psychology that may be applicable to Software Engineering groups which produce similar, predicted results. Second, it may be possible to change the effectiveness of Software Engineering groups by applying a select technique (normative manipulation) from Social Psychology. Finally, this technique may be adoptable in an industrial context - i.e. it is unobtrusive and does not generate participant resistance.
To address the three areas of the thesis, three experiments were conducted. First, to test the *applicability* of Social Psychology research to Software Engineering groups, we administer the letters-to-numbers task to a group of Software Engineers and corroborate the results obtained by Laughlin et. al [63]. Further, we found that within the groups that performed the letters-to-numbers task, groups that adopted group norms that were predicted by Social Psychology research to enhance group performance, did in fact perform better than groups that did not. To address the *effectiveness* of normative manipulation, two sets of Software Engineering groups were primed to adopt contrary norms. We showed that Software Engineering groups that adopted critical norms performed better than Software Engineering groups that adopted consensus norms in a requirements elicitation task. Finally, to address the *adoptability* of normative manipulation, we applied this technique in an industrial setting, to a group that was undergoing requirements elicitation. We showed that the group remained generally positive during the process and also generally felt motivated to participate.

### 7.1 Contributions

This research makes the following four contributions:

First, group norms identified as related to productivity were shown to be correlated with better performance. In our experiments, we learned that Software Engineering groups that had adopted group norms, identified in Social Psychology literature as having a mitigating effect on group performance loss, did indeed perform better in the letters-to-numbers task studied.

Second, Software Engineering groups were shown to behave as predicted by Social Psychology research in a letters-to-numbers task. This corroboration of results was obtained in spite of the fact that the original study used Social Psychology undergraduates, and the replicated experiment used Software Engineers. In addition, even though there is some contention as to the ‘type’ of person that is a Software Engineer, we showed that in this case, this contentiousness did not matter, and the Software Engineering group performed as predicted.

Third, this research provides evidence that group norms in Software Engineering groups can be manipulated, which in turn can affect the performance of groups on particular tasks. In our second experiment, we showed that priming groups to adopt a critical norm promoted information sharing and subsequently improved the performance of these groups in a requirements elicitation task, as opposed to groups primed with a consensus norm.

Fourth, this research provides evidence that the manipulation of group norms in an industrial environment is unobtrusive. Normative manipulation was performed on a
group as part of a larger requirements elicitation exercise, and group response remained positive. No participants detected or expressed any discomfort at the experimental activities carried out.

7.2 Future Work

There are many avenues within which this work can be extended. Some of these are discussed below.

7.2.1 Harnessing the Assembly Bonus Effect

The assembly bonus effect was demonstrated on a letters-to-numbers task in experiment 1. It may be possible that other tasks sharing the same characteristics of the letters-to-numbers tasks, (or even tasks with different characteristics) may indeed benefit from the work of a focused, concerted group effort instead of the efforts of a nominal group or a single, high performing individual.

Harnessing the Assembly Bonus Effect could prove useful in situations where the group makeup is fixed and cannot be changed. In such cases, the tasks could be organised so that the highest performing members are set to attempt tasks where raw ability does make a significant difference to the outcome, and the remaining lower performing members could be set to attempt tasks that have been shown to exhibit the Assembly Bonus Effect, therefore producing a greater likelihood of higher overall performance.

7.2.2 Normative Manipulation for other task types

This research was done firstly on a letters-to-numbers task (Experiment 1) and requirements elicitation tasks (Experiment 2 and 3). It is possible that different types of norms are required for tasks that have characteristics different from the ones studied. A taxonomy of the trifecta of individual characteristics, group characteristics (norms) and task characteristics conducive to increased group performance may be a useful reference for practice. Armed with the knowledge of the appropriate norms that should be instilled in a group based on group make-up, groups could increase the performance of tasks at will, or at the very least, increase the likelihood of good performance in upcoming tasks.

7.2.3 Normative Identification and Manipulation Framework

In this research, identification of suitable norms for a particular task and then the manipulation of group norms in order to boost group productivity was done manually with
knowledge of the relevant areas of Social Psychology research. It may be possible to develop a framework that guides future users of normative manipulation in the steps of identifying relevant Social Psychology research for testing, and coming up with related and relevant scientific measures and metrics to use on Software Engineering groups for verification. A researcher or user should be able to use the framework to quickly identify the areas of Social Psychology that are relevant to the area of interest, including results, measures, theories and implications, using some manner of terminology mapping that relates general concepts into Social Psychology research specific terms. The framework should then also guide the researcher or user through the process of verification of applicability within Software Engineering through (1) experimental set up and (2) relevant measures, and then recommend how the corroborated results should be applied, or in the case of lack of corroboration, the appropriate further experiments that should be conducted.

7.2.4 Further adoption of Social Psychology Literature

This research shows that in spite of issues that may arise due to the population involved in the experiments conducted in Social Psychology research, in some cases, this does not matter. Regardless of the lack of agreement surrounding a ‘type’ for Software Engineer, Software Engineering groups do, in some cases, perform identically to the experimental groups used in Social Psychology. Therefore, more social psychology experiments could repeated on Software Engineering groups in order to attempt to bring the research and theories of social psychology together with the Software Engineering domain.

7.3 Human Ethics Approvals

The work contained in this thesis was conducted under the Australian National University Human Ethics Protocol IDs 2010/097 and 2011/101. The author is grateful to all subjects for their participation.
Original Group Environment Questionnaire
Figure A.1: Original Group Environment Questionnaire, Pg 1

**Group Environment Questionnaire**

Directions: The Group Environment Questionnaire (GEQ) helps you assess your perceptions of an athletic team of which you are a member. If you are not currently participating on a team, answer the questions with respect to a team from your past. There are no right or wrong answers, so please give your immediate reaction. Some of the questions may seem repetitive, but please answer them all and be as honest as possible.

The following questions help assess your feelings about your personal involvement with your team. On a scale of 1 through 9, 1 indicating the strongest agreement, and 9 indicating the strongest disagreement, answer each question.

1. I do not enjoy being part of the social activities of this team.
2. I'm unhappy about the amount of playing time I get.
3. I am not going to miss the members of this team when the season ends.
4. I'm unhappy with my team's level of desire to win.
5. Some of my best friends are on this team.
6. This team does not give me enough opportunities to improve my personal performance.
7. I enjoy other parties more than team parties.
8. I like the style of play on this team.
9. This team is one of my most important social groups.
10. Our team in united in trying to reach its performance goals.
11. Members of our team would rather go out on their own than get together as a team.
12. We all take responsibility for any loss or poor performance by our team.
13. Our team members rarely party together.
14. Our team members have conflicting aspirations for the team's performance.
15. Our team would like to spend time together in the off-season.
16. If members of our team have problems in practice, everyone wants to help them so we can get back together again.
17. Members of our team do not stick together outside of practices and games.
18. Our team members do not communicate freely about each athlete's responsibilities during competition or practice.
Figure A.2: Original Group Environment Questionnaire, Pg 2

Scoring:
The GEQ measures these five elements regarding how attractive a group is to its individual members:
1. Attraction to group – task
2. Attraction to group – social
3. Group Integration – task
4. Group Integration – social

To determine your score simply add your numbers in the brackets below. However, for questions 1,2,3,4,6,7,8,11,13,14, and 17 you should reverse the score (1=9 and 9=1)

Attraction to group – task = 2,4,6,8 (range 4-36)
Attraction to group – social = 1,3,5,7,9 (range 5-45)
Group Integration – task = 16,12,14,16,18 (range 5-45)
Group Integration – social 11,13,15,17 (range 4-36)

The higher your score on each subscale, the greater you reflect that dimension with the team.
Appendix B

Raven’s Advanced Progressive Metrics & Letters-to-Numbers Experimenter script
Figure B.1: Raven’s Advanced Progressive Metrics and Letters-to-Numbers Experimenter script, Pg 1

Assembly Bonus Effect Pilot Study

**DO** Hand out booklets Set I and Set II

**SAY** Welcome to the first pilot study for COMP8100. Before we begin, we would like to say that we very much appreciate your participation in these studies. As an icebreaker, we will begin by doing some puzzles. These puzzles are simple exercises to get your mind moving after the long lecture that you have endured. Please do not turn over the booklets until we say so. Remember, this is not a test, just have fun. The results will not be recorded.

**DO** Hold up Set I

**SAY** This is a set of puzzles that will encourage you to use logic. The first set in front of you, is used to show you how the puzzles work. If you have seen these puzzles before, we will remind you how they work. Open the set of materials labelled “Set I” to the first page. You will see two problems on the first page. Please focus your attention on puzzle number 1.

The top part of puzzle 1 is a pattern with a bit cut out of it. Look at the pattern, think what the piece needed to complete the pattern correctly both along and down must be like. Then find the right piece out of the eight bits shown below.

Only one of these pieces is perfectly correct. Option 1 completes the pattern correctly going downwards, but is the wrong going the other way. Option 4 is correct going along, but is wrong going downwards.

Put your finger on the piece that is correct both ways.

Option 8 is the correct puzzle, isn’t it? So the answer for this puzzle is Option 8 and you should circle it on the sheet. Please do that now.

**DO** Check that everyone has circled option 8.

**SAY** Now try puzzle 2 yourself.

**DO** Allow approximately 20 seconds

**SAY** The correct answer for puzzle 2 is option 4. Now, you will find that the puzzles will get more and more difficult as you go along. Whether the puzzles are easy or difficult, to solve them you have to use the same method all the time.

This is a practice test, and it is not important to get it all done. If you make a mistake, write down the final correct answer on the side of all the options and circle it. If you make multiple changes, make sure that you cross out the ones that you don’t want so that your final response
can be seen clearly.

DO Allow 5 Minutes

SAY Please stop now. Close “Set I” and put it aside.

Now take “Set II” and place it in front of you. This set of puzzles will now challenge your logic and reasoning ability. Each puzzle requires the same method of working as “Set I”. The puzzles will get progressively more challenging. Attempt each puzzle in turn, do your best to find the correct piece to complete the puzzle before moving on to the next one. You may skip to the next problem if you are stuck, but remember, in every case, the next problem is much harder.

You will have 10 minutes to complete the puzzles.

Are there any questions?

DO Pause Briefly

SAY Please open Set II and begin

DO Start timing 10 minutes

SAY [after 10 minutes]

Please stop doing the puzzles.

Before we begin giving out the solutions, would anyone like to compare the puzzles to requirements analysis?

DO Pause Briefly to allow some discussion

SAY We will now reveal the answers. Please total up the number of correct answers and write them on the top of “Set II” in the space indicated.

DO Reveal solutions and allow some time for totalling to complete

ACTIVITY PERFORM LECTURE ON TASKING, ASK STUDENTS IF BRAINSTORMING WORKS [5 min]

DO Break up class into 2 groups of approximately equal trials and individuals

SAY We will now perform an exercise to see if your hypothesis is right.

DO Collect the response sheets, but let the students keep the cover sheet. Remind the students to keep the cover sheet as they will be required to award the marks.
Figure B.3: Raven's Advanced Progressive Metrics and Letters-to-Numbers Experimenter script. Pg 3

**DO**  
Hand out ARE material.

**SAY**  
Write down the response code where indicated

This is puzzle in problem solving. The objective is to figure out a code in as few a trials as possible. The numbers 0-9 have been coded as the letters A-J in random order. You will be trying to find out which letter corresponds to which number. It is important to remember that all we are doing is changing the characters used to represent the numbers. We are not changing the way the number system works. That is, we are still using the same decimal number system you've been using all your life.

**DO**  
Open example sheet to page 1.

**SAY**  
[point to sheet] Suppose that this is the code.

First you will come up with addition or subtraction equations using the letters A-J that will be solved by you and I will give you the answer in letter form. Then you will make a guess as to what one of the letters represents and write it in, and I will tell you whether or not the guess is correct (True) or incorrect (False). You will circle the answer I provide (True or False). When you feel that you have correctly mapped all ten letters to all ten numbers, you fill in the row below in grey. When you have correctly mapped all ten letters to all ten numbers, you will have solved the problem. Remember, the objective is to solve the puzzle in as few trials as possible.

I will now show you an example using the random code.

**DO**  
Flip through the example sheet.

**SAY**  
While flipping, say: on the first trial, the person chooses the equation A + B = 7, and I tell them that the solution is C. This is because A = 3 and B = 5, which sums to 8, the number represented by C. The person then guesses that A represents the number 1, and I tell him that this is not the case (False). On the second trial the person asks the solution to the equation B + C = 7 and I tell him that the answer is EA. This is because B = 5 and C = 8, which sums to 13 or EA. Note that on the next trial the person tries to add the three letters together. You may use as many letters as you desire in your equations. On the fourth trial the person uses a subtraction equation. You may use either addition or subtraction equations as you see fit.

You have 10 minutes to solve it, and only 10 trials.

**DO**  
Pause briefly

**SAY**  
We will now begin
**Figure B.4:** Raven's Advanced Progressive Metrics and Letters-to-Numbers Experimenter script, Pg 4

<table>
<thead>
<tr>
<th>DO</th>
<th>Begin timing, do experiment as described – (participants must do the question twice). Record time taken on sheet where indicated.</th>
</tr>
</thead>
<tbody>
<tr>
<td>DO</td>
<td>Hand out the course improvement comments sheets to trials (GEQ).</td>
</tr>
</tbody>
</table>
Critical and Consensus
Normative Manipulation Script
Figure C.1: Conformist Norm Priming Script

Priming Sections

**DO**  Hand out materials for study

**SAY**  Welcome to the third activity for COMP8100. Before we begin, we would like to say that we very much appreciate your participation in these studies.

Please get into groups of 3, and please do not turn over the handout until you’re told to do so.

**DO**  Hold up handout & Materials

**SAY**  Does everyone have a copy of the handout & materials?

This is first part is a course review exercise. With the provided materials, you are to design an A4 Poster that is about what you have learned in COMP8100 Software Requirements Elicitation and Analysis, and your comments.

You will be given 15 minutes *stress* to complete the exercise.

The detailed instructions for this exercise are contained within the handout.

*Your identities will be kept anonymous. *stress*

The activity is a warm up revision exercise.

The posters will be used for introduction for COM8100 Software Requirements Elicitation and Analysis in 2011.

The posters can be designed to your own taste, but you are to make sure that every member contributes to your result *stress*.

Before we begin, does anyone have any questions?

**DO**  Pause briefly

**SAY**  You may now begin.

**DO**  Allow 15 minutes

**SAY**  Please fill in the Group Activity Questionnaire before we begin the main part of this exercise. Note that these questions are different from the last time and we appreciate it if you would answer them honestly and to the best of your ability.

**DO**  Pause Briefly; collect materials and *break for 5 mins*
Figure C.2: Critical Norm Priming Script

Priming Sections

DO  Hand out materials for study

SAY  Welcome to the third activity for COMP8100. Before we begin, we would like to say that we very much appreciate your participation in these studies.

Please get into groups of 3, and please do not turn over the handout until you're told to do so.

DO  Hold up handout & Materials

SAY  Does everyone have a copy of the handout & materials?

This is first part is a course review exercise. With the provided materials, you are to discuss a topic that is about what you have learned in COMP8100 Software Requirements Elicitation and Analysis, and your comments.

You will be given 15 minutes /stress/ to complete the exercise.

The detailed instructions for this exercise are contained within the handout.

Your identities will be kept anonymous. /stress/

The activity is a warm up revision exercise.

The responses will be used for discussion in COM8100 Software Requirements Elicitation and Analysis in 2011.

Please discuss the statement to the best of your ability with your group members, and record your response in the sheet provided /stress/

Before we begin, does anyone have any questions?

DO  Pause briefly

SAY  You may now begin.

DO  Allow 15 minutes

SAY  Please fill in the Group Activity Questionnaire before we begin the main part of this exercise. Note that these questions are different from the last time and we appreciate it if you would answer them honestly and to the best of your ability.

DO  Pause Briefly; collect materials and break for 5 mins/
Appendix D

Experiment 3: Introductory Survey Example
### Introductory Survey

Thank you for your interest in the study entitled "Assembly Code Visualization and Analysis". Prior to completing this questionnaire, the research group and the Human Research Ethics Committee of the University of Victoria require that you be advised of the following:

- Your participation in this research is voluntary.
- If you decide to participate, you may withdraw at any time without any consequences or any explanation. If you decide to withdraw before submitting your responses, simply close the browser. Your responses will not be collected. It will be impossible to remove the data after it has been submitted.
- There are no known anticipated risks to you by participating in this study.
- Giving personal identification information is optional and up to you. If you choose to remain anonymous, no one will be able to identify you with your responses. Any personal information provided will be anonymized in the results. Your responses will only be used for the purpose of this study.
- Your responses are protected under the Freedom of Information and Protection of Privacy Act. Study data will be kept for three years in secure machines. At the end of this time, the computer data files will be deleted. All responses are confidential.
- All transcription of the primary data to aggregate data will be undertaken by Jennifer Baldwin. No information will be disclosed about any of the gathered data.
- Results from this questionnaire will be published in computer science and software engineering conferences and journals, presented at scholarly meetings, and will form part of the thesis one dissertation. Results will only ever be presented in aggregate form.
- If you need help filling out the questionnaire, send an e-mail to jbles@cs.uvic.ca.
- This research is being funded by the Department of Computer Science at the University of Victoria.
- Please contact Jennifer Baldwin (jbles@cs.uvic.ca) and/or Dr. Coady Lyons (lyons@cs.uvic.ca) for more information.
- In addition to being able to contact the study researchers, you may verify the ethical approval of this study, or raise any concerns you might have, by contacting the Associate Vice-President, Research at the University of Victoria (1-250-472-5418).

By completing and submitting the questionnaire, YOUR FREE AND INFORMED CONSENT IS IMPLIED and indicates that you understand the above conditions of participation in this study and that you have had the opportunity to have your questions answered by the researchers.

There are 22 questions in this survey

## Section A

1. **What is your expertise with?**

   Please choose the appropriate response for each item:

<table>
<thead>
<tr>
<th>Reading and understanding assembly language</th>
<th>Novice</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>Expert</th>
</tr>
</thead>
<tbody>
<tr>
<td>Implementing features or maintaining assembly language</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Section B

2 What is your primary role? *
Please choose only one of the following:
- Implementation
- Maintenance
- Testing
- Management
- Other

3 Other than your primary role, if you perform secondary roles, please list them:
Please write your answer here:

4 Percentage of time you spend on the following: *
- Total of all entries must equal 100
Please write your answer(s) here:
- Managerial interactions
- Engineer interactions
- Group meetings
- Pair programming
- Working alone
- Other
5. For your answer for 'Other' above, please describe how that time is spent. *

Only answer this question if the following conditions are met:
* Answer was greater than at question 4 (Percentage of time you spend on the following: (Other))

Please write your answer here:

6. How would you best describe your team structure?

Managerial-centered: Where the project leader acts as the central clearing house processing and coordinating the work of each engineer while the engineers each provide the necessary input based on their area of expertise but with little awareness of the larger whole.

Team-centered: Where completing the work depends more generally on the group than on any one particular engineer or project leader, and where much of the work is done collectively as a group.

Expertise-centered: Where engineers depend on each other to coordinate the work based on their areas of specialized knowledge, and the project leader only oversees the process to ensure that the work comes together in the end.

Please choose only one of the following:
- Managerial-centered
- Team-centered
- Expertise-centered

7. How is knowledge in your team distributed? *

Please choose all that apply:
- People in your team have overlapping knowledge of modules.
- People 'own' modules.
Figure D.4: Introductory Survey, Pg. 4

8 Please enter your description of "Other" above. *

Only answer this question if the following conditions are met:
* Answer was of question 7 [If How is knowledge in your team distributed?]

Please write your answer here:
Section C

9 To help us understand your typical work, can you tell us about some recent work tasks you were asked to complete? *

Please write your answer here:

[Blank space]

5 of 14
Figure D.6: Introductory Survey, Pg. 6

10 Pick one task above and describe the specific steps you performed to complete it. *

Please write your answer here:

11 For the task you selected in 7a, indicate what percentage of time you spent on each step. *

Please write your answer here:

12 How were the above tasks assigned to you? *

Please write your answer here:
For example, did you use a job ticketing system?

13 Once assigned a task, who decides how that task will be carried out? *

Please write your answer here:

This might include how it is implemented.

14 Is there an approval process for how the task is carried out? *

Please choose only one of the following:

- Yes
- No

* Answer was "Yes" at question '14 [10]' (is there an approval process for how the task is carried out?)

15 What is the approval process? *

Only answer this question if the following conditions are met:

- Answer was "Yes" at question '14 [10]' (is there an approval process for how the task is carried out?)

Please write your answer here:
Figure D.8: Introductory Survey, Pg. 8

16 Do you follow a process for validation of changes to the code? *

Please choose only one of the following:
- Yes
- No

17 What is the validation process? *

Only answer this question if the following conditions are met:
* Answer was "Yes" at question 16 [11] (Do you follow a process for validation of changes to the code?)

Please write your answer here:

For example, are tickets closed in a system by customers? Do you perform code reviews with team members?
Section D

Directions: The following statements concern your perception about yourself in a variety of situations. There are no "right" or "wrong" answers, so select the answer that most closely reflects you in each statement. These results are being used in scientific research, so please try to give accurate answers. Take your time to consider each statement carefully.

Purpose: Jennifer's PhD research is to build visualization and support tools for assembly language. Within the framework of her PhD, requirements for the tool need to be elicited in such a way that their inclusion in the tool can be scientifically validated and traced. This includes providing the full research context and using interdisciplinary best current practices to corroborate findings in requirements gathering. To this end, the following four questions study how cognitive transfer and information sharing affect the gathering of requirements. Once the study is complete, we will provide further explanations as to why we are asking these questions.

This section of the survey should take you less than 5 minutes. Thank you SO MUCH for participating!

18. Do you: *

<table>
<thead>
<tr>
<th>Question</th>
<th>Never</th>
<th>Rarely</th>
<th>Occasionally</th>
<th>Frequently</th>
<th>Always</th>
</tr>
</thead>
<tbody>
<tr>
<td>Consult many different opinions before forming your own view?</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Tend to work individually, and others in your team do the same?</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>State your opinions clearly?</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Make an effort to avoid disagreements with team members?</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
<tr>
<td>Consult other team members before making a decision on how to implement a requirement?</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
<td>☐</td>
</tr>
</tbody>
</table>

19. Do you: *
### Figure D.10: Introductory Survey, Pg. 10

Please choose the appropriate response for each item:

<table>
<thead>
<tr>
<th>Item</th>
<th>Not at All</th>
<th>Very Little</th>
<th>Little</th>
<th>Somewhat</th>
<th>Much</th>
<th>Very Much</th>
</tr>
</thead>
<tbody>
<tr>
<td>Find that a well-ordered development routine is helpful?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Quickly become impatient or irritated if you would not find a solution to a coding problem immediately?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Feel relieved when you have made a decision on how to implement a requirement?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Find that establishing a consistent routine enables you to enjoy development more?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enjoy having a clear and structured development process?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Consider your work habits or skills as different from others on your team?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Value work privacy?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Enjoy working with others?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Approach problems differently depending on the team, the project, or both?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Respect the work decisions made by those senior to you?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Know your weaknesses and strengths?</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Figure D.11: Introductory Survey, Pg. 11

Section E

20 What is your opinion of: *
Please choose the appropriate response for each item:

<table>
<thead>
<tr>
<th>Requirements that are uncertain?</th>
<th>Strongly Dislike</th>
<th>Dislike</th>
<th>Somewhat Dislike</th>
<th>Somewhat Like</th>
<th>Like</th>
<th>Strongly Like</th>
</tr>
</thead>
<tbody>
<tr>
<td>Coding tasks which could be implemented in many different ways?</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
<tr>
<td>Being assigned a requirement to implement without knowing what you can expect from it?</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
<tr>
<td>Working with a team that lacks process?</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
<tr>
<td>When a team member’s technical information is ambiguous?</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
<tr>
<td>Unpredictable requirements?</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>

21 Please rate the following: *
Please choose the appropriate response for each item:

<table>
<thead>
<tr>
<th>Concerned</th>
<th>Not at All</th>
<th>Very Little</th>
<th>Little</th>
<th>Somewhat</th>
<th>Much</th>
<th>Very Much</th>
</tr>
</thead>
<tbody>
<tr>
<td>When you don’t understand the reason why code behaved unpredictably during development?</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
<tr>
<td>When a team member holds a unique opinion from everyone else in the team?</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
</tr>
</tbody>
</table>
Figure D.12: Introductory Survey, Pg. 12

| When confronted with a coding problem, do you need to reach a solution very quickly? | Not at All | Very Little | Little | Somewhat | Much | Very Much |
| Are you proud of accomplishing tasks that others have not yet accomplished? | | | | | | |
| How important is competitive spirit on a team? | | | | | | |
| Would you help others with their work tasks, within your means? | | | | | | |
| How easily can your work be understood from the work of others on your team? | | | | | | |
| Would you rather work on a task as part of a team than alone? | | | | | | |

https://wsproject.limequery.com/admin/admin.php?action=s...
Figure D.13: Introductory Survey, Pg. 13

Section F

22 What is your opinion on taking this survey? *

Please choose only one of the following:

- Strongly Dislike
- Dislike
- Somewhat Dislike
- Somewhat Like
- Like
- Strongly Like

Make a comment on your choice here:

For example, are there any changes you would make? Do questions you didn't agree with? Likes or dislikes?
We really appreciate you taking the time to respond to this survey; your contributions are incredibly helpful.

01.01.1970 - 01.00

Submit your survey.
Thank you for completing this survey.
Experiment 3: Task, Responsibility and Interactions Responses

Where possible, a brief dictionary is provided below:

- **$SDMLUIFREQ** - variable name used in code
- **ABEND** - ABnormal END (system code)
- **ASINFO** - control block method
- **ASMON** - Address Space MONitor, a command used in a program that monitors network activity on z/OS mainframes
- **CDS** - Compare Duplicate Swap
- **DSECT** - theoretical data structure used in assembly programming, used for ease of reference to remove need for the notation of register + offset
- **IPCS** - Interactive Program Control System (outputs information about active interprocess communication facilities)
- **IRB** - Interrupt Request Block
- **MVS/JES** - Multiple Virtual Storage (an IBM developed mainframe operating system)/Job Entry Subsystem (receives jobs into the operating system and schedules them for processing by MVS)
- **MF 2.0 initiative** - new mainframe development product
- **NMSSI** - network focused product in cross-system resource management
• PSW - Program Status Word (used to contain the address of the next program instruction and control information about the program that is running)

• Solve:OPS - Product used to automate management and resources of mainframes

• SOPS - see Solve:OPS

• SRB - Service Request Block mode, or cross-memory mode (in zIIP CPUs, this mode allows the use of structures in other access spaces to increase the amount of space available)

• SSM - see System State Manager

• System State Manager - a framework used for cross-system resource management

• TCB - Task Control Block (Designates a normally executing a task structure. A program can be broken down into multiple tasks to explicit multiple processors)

• VTAM - Virtual Telecommunications Access Method, a software package that provides communications via telecommunication devices for mainframe environments.

• Debbie - Fellow staff member (non-development staff, and not involved in experiment)

Implementer 1 task - “I was recently asked to add a feature to Solve Operations (our automation product) where an event is generated when an address space terminates. This was a customer enhancement request... Customer has had issues where MVS/JES job ended message was not reliable and caused problems with on-processing. They would like a more reliable trap that ‘job ended’. They also mentioned that the MVS commands used for displaying jobs is irritating, and would prefer a control block method (ASINFO). ASMON is a red herring. Passed back to Debbie to examine if job (really, really) ended is available as ENF event that we can use, use of ASINFO as alternative to display command. Delivery mechanism would be a sample template for SOPS customers demonstrating use of new display/ended methods.’

Implementer 1 task approach - “The steps involved in completing this task were Firstly to find the module that was called as part of End Of Memory processing... This involved finding the binder deck with the list of modules used by this part of the product along with comments that described that function. Secondly, [I] Discuss[ed] with product architect the best approach for this. [Subsequently], make coding changes. [Followed by] test[ing] on an isolated system. [Finally], hand[ing] over changes to applications programmer to enable them to make their changes and to test, fixing bugs as required.”
**Implementer 1 time allocation** - “Finding the module - 5%; Discussion with architect - 5%; Coding changes - 60%; Test - 20%; Handover - 10%”

**Implementer 2** - “Implement [redacted] MF 2.0 initiative modifying all NetMaster Lab’s products to interact with [redacted] System State Manager. It encompasses coding in 3 different environments, Management Services, NMSSI and Stand Alone, although specific, application… The task, verbally expressed by the Project Manager was like this: ‘[Implementer 2], you will implement [redacted] SSM compliance across our products starting with Solve:OPS.’”

**Implementer 2 task approach** - “The rest was up to me, finding SSM documentation, contacting SSM Team, write the prototype, research SSM interfacing impact on our code, design the implementation, register application names with the SSM team, code the implementation, test it, create messages help, provide technical writer with necessary details for the documentation, present the implementation to the Quality Assurance Team, address QA questions, fix detected problems.”

**Implementer 2 time allocation** - “It is really very hard to quantify this, I will not provide the percentage as it would be inaccurate. The process is dynamic, tasks overlap, I do not keep exact track of time spent on each activity.”

**Implementer 3** - “[Example 1 - Maintenance Issue]: Customer issue - high CPU usage. Identify and redevelop code to improve CPU usage. [Example 2 - Development Issue]: Old code does not support new Operating System features, causing incorrect logic. Require to research new methodology and to redesign interface to Operating system internals. [Example 3 - Development Issue]: data sampling of statistics of a particular type of network connection. One of a large team, I was given one component to address. Research topic, design and document design, team analysis of design, initial code pass 1, configure test environment, test code, internal presentation of component to development and quality testing teams, quality team testing, iterations of analysis/design/code/test as required, final internal technical documentation”

**Implementer 3 task approach** - “[Using the first example of] high CPU usage customer issue: [The] original thinking at [the] time of design was to handle number of inter-system connections within the one organisation in the order of hundreds or less. Years later [(now)], some customers [are] running tens of thousands of these connections. Code still working OK but CPU usage [is] unacceptable. [I’ve identified] bottlenecks, in this case, a major factor was optimal buffering of passing messages from one address space to another. Other factors [include] excessive loops through available
data accumulating via different criteria. Solution - increase buffering size, compress record size, optimise looping control. [Currently] awaiting customer verification, as our environment cannot reproduce the excessive size of the customer environment. [Although this is a] single developer solution, some team advice [was] sought."

**Implementer 3 time allocation** - "Team discussion - 2%; Management meetings discussing high priority customer issues - 2%; Design and code - 20%; Testing (and correcting) new code - 70%; Document change - 4%; Formalised writing of the fix, to be delivered to customer - 2%"

**Maintainer 4 tasking and approach** - "Usually, I look at dumps from customers. The PSWs and contents of registers will show where I should look at first. If they are within our code, the listings of the assembler modules and cross references of the fields can be used to understand the problems more. Some data may be corrupted by others. The data will be scanned through the dump to see where does it come from and the address will be closed to the contents of a register or so."

**Maintainer 5** - "I recently resolved a problematic timing issue where queue chains were getting corrupted in a multi-cpu environment. The code was doing a CS (compare and Swap) instruction on a double worded chain and the chain header queue was getting corrupted... Code was changed to do CDS in order to maintain a valid queue header."

**Maintainer 5 task approach** - "I searched on all occurrences where the $SDM-LUIFREEQ had been updated. This is a cross reference that we keep on all DESCTs and where fields are used."

**Maintainer 6** - "Debugging an ABEND, consulting development and creating a fix in the form of a source change for a customer issue."

**Maintainer 6 task approach** - "[I] used IPCS to analyse a dump provided by a customer [to research] why the ABEND had occurred by examining the source listings of the ABENDimg module. [I] consulted development on the source change to correct the ABEND. [Then,] I wrote the source change, test[ed the change, and finally] created a Program Temporary Fix (PTF) from the source change (IBM z/OS system)"

**Maintainer 6 time allocation** - "Analysis - 70%; Consult Development - 10%; Source change - 5%; Test - 10%; Create PTF - 5%"
Maintainer 7 - “ABEND [in program execution]. Code had been unchanged for many years and was now failing owing to exploitation of new hardware (zIIP CPU) that required code to run in SRB not TCB mode”

Maintainer 7 task approach - “[Steps taken:] 1 - Analyse dump, symptoms could be explained from timing issue and lack of re-entrancy. IBM VTAM manual was ambiguous on scheduling of IRB exits. 2 - Discussed findings with architect, agreed to make all VTAM exits re-entrant (3 programs). 3 - Implement code change and create fixes for failing and current release of the product. 4 - Test changes (regression test only - problem could not be re-created). 5 - Send fix to customer for testing.”

Maintainer 7 time allocation - “Read dump and research problem - 60%; Discuss with architect 3%; Implement 17%; Test 20%”
Experiment 3: Task assignment, Decisions and Approval

Implementer 1 task assignment - "[The task is assigned by] an email from [the] development manager"

Implementer 1 task execution - "[How the task is carried out is decided by] discussion between myself, architect and applications programmer."

Implementer 1 task approval process - "None"

Implementer 2 task assignment - "The tasks were not assigned to me, We do not assign the individual tasks, just a main task. We know how and what to do implementing complex software implementations."

Implementer 2 task execution - "I do [decide on how to approach the task]."

Implementer 2 task approval process - "Yes, [it includes] interactions with the Project Manager, discussions with the Main Architect, documenting and sending of the design for comments in the form of the DDS (Detailed Design Specification), formal implementation presentation (live code demonstration) to Project Manager, peers (sometimes), Quality Assurance, Technical Support, Technical Writer."

Implementer 3 task assignment - "[The tasks are assigned by] management discussions... We have a development methodology process in place."

Implementer 3 task execution - "In general, the assignee [does, although he] can request advice from other team members"

Implementer 3 task approval process - "No."

Maintainer 4 task assignment - "I can pick up issues from a queue for all support persons. Manager and other colleagues can also assign some to me."
Maintainer 4 task execution - "I will decide to work on which jobs first and for how long. If an issue is at priority 1 or requested by customer or my manager or account manager, I will do it first and for most of my time."

Maintainer 4 task approval process - "No."

Maintainer 5 task assignment - "I have knowledge on the particular area of code and therefore it was assigned to me."

Maintainer 5 task execution - "My manager."

Maintainer 5 task approval process - "No."

Maintainer 6 task assignment - "I assigned the tasks to myself."

Maintainer 6 task execution - "I decide."

Maintainer 6 task approval process - "No."

Maintainer 7 task assignment - "Issue was assigned to another engineer and then assigned to me by my manager after it was not resolved. All issue management via STARTRAK system (CICS 3270 application with web interface for customers and engineers)."

Maintainer 7 task execution - "The assigned engineer (possibly seeking advice from development engineers as required and according to their expertise)."

Maintainer 7 task approval process - "No."
Experiment 3: Priming and Individual Brainstorming Exercise
INSTRUCTIONS

The following pages include individual exercises to be completed before you begin your work day. **Please complete them in order.**

The first four exercises, on yellow paper, should take no longer than 15 minutes. The last exercise, on blue paper, is one that you can work on throughout the day.

These pages will be collected tomorrow after the group session. Only the last exercise, on blue paper, will be discussed within the group.

Feel free to be as **creative as you like** with your responses.
Figure G.2: Priming and Individual Brainstorming Exercise, Pg. 2

A. Write three statements describing your particular area(s) of expertise (i.e., topics that other team members consult you on).
B. Write three statements about how your area(s) of expertise does not overlap with that of other team members.
C. Write three statements about why you think it is advantageous for individuals to have an area of expertise.
D. Critically comment on one particular tool, does not matter which one, which you use in the course of your work (e.g., how effective it is, or how its effectiveness could be improved). Please provide at least three comments.
E. Circle the option that best suits you.

| Maintenance | Development |

F. While working with Assembler, there may be times when you wish you had additional tool support or visualization support.

Please write down the tasks for which you wish you had that support. For example, note times when you think "I WISH I COULD SEE..." or "I WISH I COULD FIND..." An example might look like "locating every modification of a particular DSECT".

Please add to the list during the day as you work. Anything you can come up with is fine, but the focus should be on problems and not solutions.
Experiment 3: Group Session Script
0 min Introduction

**SAY**

Hi, I'm Jennifer Baldwin from the University of Victoria in Canada. For my PhD in Computer Science, I am exploring how visualization and tool support for assembly language might be useful. My work is being funded by the Department of Defense and CA in Canada.

Since you are experts in the area, we really value your experience and expertise in defining the issues.

This is Alvin from the Australian National University. For my PhD in Software Engineering, I explore productivity in Software Engineering Teams.

To get started, I'd like to collect the ethics forms that you were given yesterday.

**DO**

Collect the ethics forms.

**SAY**

This session should take no longer than 2 hours, including a 20 minute break. The aim is to discuss and critically rank all of the items from the exercise yesterday. If you come up with new ideas during the session, please add them to your list. Feel free to be creative.

Does everyone have the blue pages?

First of all, I'd like to go around the table and have everyone introduce themselves and tell us about your job. We also know from the survey that your teams are expertise-centered, so it would be great to hear about that, as well as your interests.

10 min Listing of Ideas

**SAY**

Now to begin the group exercise, we will go around the table and each person will share one item from their list at a time. At this time, please avoid discussion or talking out of turn.

After all of the items are listed, we will have a discussion to clarify the items. If you have any new ideas then feel free to add them to your sheet. If you want to skip a turn, that is also fine.

**DO**

Record word for word what each person says on the power point slide.
### Figure H.2: Experiment 3 Group Session Script, Pg. 2

<table>
<thead>
<tr>
<th>Time</th>
<th>Action</th>
<th>Notes</th>
</tr>
</thead>
<tbody>
<tr>
<td>30 min</td>
<td><strong>SAY</strong> 30 min</td>
<td>We will now have a 30 minute discussion on all the ideas generated.</td>
</tr>
<tr>
<td></td>
<td><strong>DO</strong></td>
<td>Now is the time to ask for clarification or elaboration on an idea, or dispute or defend an item.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You are also welcome to suggest new items during this time, but no items can be eliminated.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>We’ll go through them item by item.</td>
</tr>
<tr>
<td>60 min</td>
<td><strong>SAY</strong> 60 min</td>
<td>Announce each item on the list and ask if there’s and ask what it means, or how people feel about it. Record any new ideas on the power point slide.</td>
</tr>
<tr>
<td></td>
<td><strong>DO</strong></td>
<td>Now if everyone could take out their yellow sheet for preliminary ranking.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>You can see there are 10 spaces to be filled in. You can select 10 items that are the most important for you from all of the options. Then assign them a rank which is a numbering between 1 and 10, where 10 is the most important.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Once you are finished, please turn it face down on the table and then you are free to take a break for about 20 minutes.</td>
</tr>
<tr>
<td>70 min</td>
<td><strong>DO</strong></td>
<td>Go around the table and transcribing and summing up the points from the ranking sheets onto the power point slides. Then reorder them on the slide based on the greatest number of points.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Collect everyone from after their break.</td>
</tr>
<tr>
<td>90 min</td>
<td><strong>SAY</strong> 90 min</td>
<td>We have reordered the items according to rank and you can see the score for them. We have also highlighted the top-ten.</td>
</tr>
<tr>
<td></td>
<td><strong>SAY</strong></td>
<td>We will now have a free-for-all discussion about the nature and content of the top-ten.</td>
</tr>
<tr>
<td></td>
<td><strong>SAY</strong></td>
<td>We would also like to hear how you feel about items that should have been included or excluded from this list.</td>
</tr>
</tbody>
</table>
Figure H.3: Experiment 3 Group Session Script, Pg. 3

110 min  SAY  Now if everyone could take out their green sheet for final ranking. Here you will again list the top ten items that you think are the most important.

Re-Ranking and Rating Revised 'Top-Ten' Items

This may be the same ten, or feel free to modify which items are in your top ten.

The ranking here is different in that 100 points will be given to the most important item. Every other item can have a value between 0 and 100. Two items can have the same ranking.

Once you are finished, please hand in your sheets to me face down, and then we're all done!

DO  Collect the green sheets from everyone and tally up the final scores based on the 0 to 100 ranking.

END CASE STUDY AT (START + 120 MIN)
Experiment 3: Sample Exit Questionnaire

Note: Question 4 ("Do you think that being videotaped during the process inhibited or promoted your ability to participate fully in making suggestions or discussion?") and Question 6 ("Do you think having an external person drive the discussion process inhibited or promoted your ability to participate fully in making suggestions or discussion?") were discarded from analysis. The purpose of these questions were as a medium of communication and the value of these questions lie in identifying extreme dissatisfaction in participant experience.

Both question 4 and question 6 mixes the of the presence of the experimenters and the process, which make it hard to isolate if (1) the presence of the observers (referring to the videotaping equipment and the experimenters) was the main driver for participant comfort/discomfort, or (2) the process introduced by the observers was the main driver for participant comfort/discomfort. A better set of questions that can be used for future work could be to make these constructs distinct for analysis. A suggestion for this is breaking down the questions into two parts, the first part of the question addressing only the presence of the observers, and the second the process itself.
Exit Survey

Thank you for your interest in the study entitled "Assembly Code Visualization and Analysis". Prior to completing this questionnaire, the research group and the Human Research Ethics Committee of the University of Victoria requires that you be advised of the following:

- Your participation in this research is voluntary.
- If you do decide to participate, you may withdraw at any time without any consequences or any explanation. If you decide to withdraw before submitting your responses, simply close the browser. Your responses will not be collected. It will be impossible to remove the data after it has been submitted.
- There are no known or anticipated risks to you by participating in this study.
- Giving personal identification information is optional and up to you. If you choose to remain anonymous, no one will be able to identify you with your responses. Any personal information provided will be anonymized in the results. Your responses will only be used for the purpose of this study.
- Your responses are protected under the Freedom of Information and Protection of Privacy Act. Study data will be kept for three years in secure machines. At the end of this time, the computer data files will be deleted. All responses are confidential.
- All transcription of the primary data to aggregate data will be undertaken by Jennifer Baldwin. No information will be disclosed about any of the gathered data.
- Results from this questionnaire will be published in computer science and software engineering conferences and journals, presented at scholarly meetings, and will form part of at least one dissertation. Results will only ever be presented in aggregate form.
- If you need help filling out the questionnaire, send an e-mail to Baldwin@cs.uvic.ca.
- This research is being funded by the Department of Computer Science at the University of Victoria.
- Please contact Jennifer Baldwin (Baldwin@cs.uvic.ca) and/or Dr. Coats (Coats@cs.uvic.ca) for more information.
- In addition to being able to contact the study researchers, you may verify the ethical approval of this study, or raise any concerns you might have, by contacting the Associate Vice-President, Research at the University of Victoria (1-250-472-5418).

By completing and submitting the questionnaire, YOUR FREE AND INFORMED CONSENT IS IMPLIED and indicates that you understand the above conditions of participation in this study and that you have had the opportunity to have your questions answered by the researchers.

There are 18 questions in this survey.

Section A

1 Were you able to fill out the yellow sheets in entirety before beginning the exercise on the blue sheet? *

Please choose only one of the following:
- Yes
- No

As a reminder, the questions on the yellow sheets were:

1. Write three statements describing your particular area(s) of expertise (i.e. topics that other team members consult you on).

2. Write three statements about how your area(s) of expertise does not overlap with that of other team members.

3. Write three statements about why you think it is advantageous for individuals to have an area of expertise.
2. If not, was there any particular reason? (i.e. a question didn’t make sense) *

Only answer this question if the following conditions are met:
* Answer was ‘no’ at question 1 [?] (Were you able to fill out the yellow sheets in entirety before beginning the exercise on the blue sheet?)

Please write your answer here:


3. Were the objectives of the overall exercise and group session clearly explained and easy to understand? *

Please write your answer here:


4. Do you think that being videotaped during the process inhibited or promoted your ability to participate fully in making suggestions or discussion? *

Please choose the appropriate response for each item:

<table>
<thead>
<tr>
<th>Videotaping</th>
<th>Strongly Inhibited</th>
<th>Inhibited</th>
<th>Slightly Inhibited</th>
<th>Slightly Promoted</th>
<th>Promoted</th>
<th>Strongly Promoted</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>□</td>
<td>□</td>
<td>□</td>
<td>□</td>
<td>□</td>
<td>□</td>
</tr>
</tbody>
</table>


5. Any additional comments?

Please write your answer here:
Figure I.3: Experiment 3 Sample Exit Questionnaire, Pg. 3

6 Do you think that having an external person drive the discussion process inhibited or promoted your ability to participate fully in making suggestions or discussion? *

Please choose the appropriate response for each item:

<table>
<thead>
<tr>
<th></th>
<th>Strongly Inhibited</th>
<th>Slightly Inhibited</th>
<th>Slightly Promoted</th>
<th>Promoted</th>
<th>Strongly Promoted</th>
</tr>
</thead>
<tbody>
<tr>
<td>External Person</td>
<td>□</td>
<td>□</td>
<td>□</td>
<td>□</td>
<td>□</td>
</tr>
</tbody>
</table>

7 Any additional comments?

Please write your answer here:

8 Did you feel motivated to share your personal opinions? (i.e. you said every issue that came to mind). Why or why not? *

Please write your answer here:
**Figure I.4: Experiment 3 Sample Exit Questionnaire, Pg. 4**

<table>
<thead>
<tr>
<th>Question</th>
<th>Options</th>
</tr>
</thead>
<tbody>
<tr>
<td>Was there sufficient communication during the activity? *</td>
<td></td>
</tr>
<tr>
<td>Please choose the appropriate response for each item:</td>
<td></td>
</tr>
<tr>
<td>Quantity of Communication</td>
<td>Very Insufficient</td>
</tr>
<tr>
<td>Quality of Communication</td>
<td>Very Insufficient</td>
</tr>
<tr>
<td></td>
<td>Insufficient</td>
</tr>
<tr>
<td></td>
<td>Sufficient</td>
</tr>
<tr>
<td></td>
<td>Somewhat Insufficient</td>
</tr>
<tr>
<td></td>
<td>Somewhat Sufficient</td>
</tr>
<tr>
<td></td>
<td>Sufficient</td>
</tr>
<tr>
<td></td>
<td>Very Insufficient</td>
</tr>
</tbody>
</table>

**10 Any additional comments?**

Please write your answer here:

**11 Did you think the process was useful for the organisation? If so, how? * **

Please write your answer here:

**12 Did you think the process was useful for yourself? If so, how? * **
Figure I.5: Experiment 3 Sample Exit Questionnaire, Pg. 5

13 Is there any way in which you think the process could be improved? *

14 Overall, was it a positive or negative experience? *

15 Do you think that some part of the process should be adopted within your group? For example, this might include open discussion of issues, future work, review etc. *
Figure I.6: Experiment 3 Sample Exit Questionnaire, Pg. 6

LimeSurvey - Your online survey service - Exit Survey
https://savproject.limesurvey.com/admin/admin.php?action=a...

Please write your answer here:

16 If you have any additional comments, please enter them below.

Please write your answer here:
Figure I.7: Experiment 3 Sample Exit Questionnaire, Pg. 7

LaneService - Your online survey service - Exit Survey

https://wwwproject.limequery.com/admin/admin.php?action=s...

Thanks for responding, your contributions are very helpful!
01/01/1970 - 01:00

Submit your survey.
Thank you for completing this survey.
Bibliography


189


