Computational Studies on the Role of Social Learning in the Formation of Team Mental Models

A thesis submitted in the fulfilment of the requirements for the degree of

Doctor of Philosophy

Vishal Singh

Design Lab
Faculty of Architecture, Design and Planning
The University of Sydney
2009
DECLARATION

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published and written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the university or another institute of higher learning, except where due acknowledgement has been made in the text.

VISHAL SINGH
28 August 2009
Acknowledgement

This thesis has been an enriching experience. In the process of conducting my research, I have learnt from my interactions with a number of people, and by observing the research activities of my peers and the broader research community. In that sense, this research is as much a result of social learning as it is a product of a scholarly effort.

I am particularly thankful to my supervisors, Dr. Andy Dong and Prof. John Gero for their constant support and guidance. By now, Andy seems to have a well-developed mental model of me, which he effectively used to keep me on track, and shepherd me whenever I tended to deviate. His enthusiasm, guidance and insightful comments have been critical to my research. John has been instrumental in shaping my interest in agent-based modelling, and his passion for design research is contagious. I have had a great time at The University of Sydney, developing friendships with many, especially Somwrita, Nick, Kaz, Ning, Jerry and Lucila. I am also thankful to Rob for his timely inputs on model implementation.

Embarking on a PhD research by itself needs motivation and interest. In that respect, I am thankful to my teachers at the Indian Institute of Science (IISc), especially Prof. Amaresh Chakrabarti, Prof. B. Gurumoorthy and Dr. Dibakar Sen. Their knowledge and humility inspired me throughout my stay at IISc.

I have been equally lucky to have an amazing family, which made me what I am. A fair bit of me is a reflection of my siblings, Pankaj and Niru, and their love and support has also contributed to this work in many ways. The years of my PhD candidature have also seen pleasant additions to my family, and Shantanu (my BIL) and Rukmini (my SIL) have also been a constant support. Gargi, my niece, is too young to read this at the moment, but seeing her grow and demonstrate amazing learning skills has been a lovely source of fun and excitement for the last year and a half. But above all, I can never be grateful enough to have the parents that I have. This thesis is a tribute to their years of efforts and sacrifices.
Abstract

This thesis investigates the role of social learning modes in formation of team mental models based on empirical data obtained from computer simulations. The three modes of social learning considered are: learning from personal interactions, learning from task observations, and learning from interaction observations. The contribution of each of the social learning modes to the formation of team mental models and how they relate to team performance is investigated for different cases. The cases used in these simulations vary in terms of the modes of learning available to the agents, the busyness levels of the agents, the team structure, the levels of team familiarity, and the task type.

The computational model is implemented in JADE (Java Agent Development Environment), using simple reactive agents. Modes of social learning, busyness levels, team structure and team familiarity are the control parameters used in computational simulations of the agents performing routine and non-routine tasks. Team performance is assessed in terms of the levels of team mental models formed and the ‘time’ taken to complete the tasks. The reduction in time is taken as the indicator for increase in team performance.

The findings validate the research’s main hypothesis that the modes of social learning have a statistically significant effect on team mental model formation. However, busyness levels and team structure also have a significant effect on team mental model formation. Learning from task observations has a greater contribution to increasing amounts of team mental model formation than learning from interaction observations. The efficiency of team mental models varies with the team structure. Compared to the flat teams, the efficiency of team mental model formation is greater in the task-based sub-teams. Higher busyness levels of agents are correlated with lower levels of team mental model formation, but, in general, busyness levels have no significant effect on the team performance. Higher levels of team familiarity are correlated with improved team performance. However, the pattern of increase in the team performance, with the increase in levels of team familiarity, is contingent on the task type and the learning modes available to the agents. In general, the rate of increase in team performance, with increasing levels of team familiarity, is greater at higher levels of team familiarity.

Conformity of the research findings to the literature on team mental model suggest that a computational study of team mental models can provide useful insights into the contribution of social learning modes to the formation and role of team mental models. These findings will be useful for the team managers in deciding the team composition (level of familiarity), work loads (busyness level), and the team structure, contingent on the nature of the design task, the available technical support for social interactions and observations (social learning) in distributed teams, and the project goals.
# Table of contents

Chapter 1  **Introduction** ......................................................................................................................... 15

1.1 Motivation.............................................................................................................................................. 16
  1.1.1 Conceptual motivation ..................................................................................................................... 19
  1.1.2 Methodological motivation ............................................................................................................ 20
1.2 Aim ......................................................................................................................................................... 22
1.3 Objectives ............................................................................................................................................. 22
1.4 Research claims, contributions and significance .................................................................................. 23
  1.4.1 Conceptual framework .................................................................................................................. 23
  1.4.2 Computational modelling.............................................................................................................. 24
1.5 Thesis structure ................................................................................................................................... 25

Chapter 2  **Background** ............................................................................................................................ 26

2.1 Social learning and social cognition ...................................................................................................... 26
2.2 Teams and organizations ....................................................................................................................... 28
  2.2.1 Team structures .............................................................................................................................. 30
  2.2.2 Teamwork and team building......................................................................................................... 32
    2.2.2.1 TMM and transactive memory ................................................................................................. 33
    2.2.2.2 Mental models and design teams ............................................................................................ 34
    2.2.2.3 Measuring TMMs: .................................................................................................................. 35
    2.2.2.4 Expertise and team performance.............................................................................................. 36
  2.3 Research method.................................................................................................................................. 38
2.4 Requirements for agent architecture and learning: ................................................................................. 40
2.5 Summary................................................................................................................................................. 43

Chapter 3  **Research Approach and Hypotheses** ...................................................................................... 45

3.1 Research framework .............................................................................................................................. 45
3.2 Hypotheses being investigated............................................................................................................... 47
  3.2.1 Correlation between social learning modes and busyness levels .................................................. 47
  3.2.2 Correlation between social learning modes and team familiarity ................................................. 49
  3.2.3 Correlation between social learning modes and team structure: ............................................... 50
  3.2.4 Correlation between social learning and task types: .................................................................... 54

Chapter 4  **Conceptual Framework and Computational Modelling** .......................................................... 63
4.1 Modelling decisions.............................................................................................................................. 63
  4.1.1 Team................................................................................................................................................. 63
    4.1.1.1 Team structure and social learning ......................................................................................... 64
### Chapter 6: Experiments designed to test the research hypotheses

#### 6.1 Calculating the value of TMM formed

6.1.2 Calculating the value of TMM formed ............................................................... 113
6.1.3 Discussion of simulation results: ................................................................. 114

#### 6.2 Experiments designed to test the research hypotheses

6.2.1 Details of experiments conducted: ............................................................... 119
- 6.2.1.1 Experiments with routine tasks and busyness ........................................... 119
- 6.2.1.2 Experiments with routine tasks and team familiarity .............................. 121
- 6.2.1.3 Experiments with non-routine tasks and busyness ............................... 121
- 6.2.1.4 Experiments with non-routine tasks and team familiarity ...................... 122

#### 6.2.2 Simulation results

- 6.2.2.1 Experiments with routine tasks and busyness level .............................. 123
- 6.2.2.2 Experiments with non-routine tasks and busyness level ....................... 126
- 6.2.2.3 Experiments with non-routine tasks and team familiarity .................... 127
- 6.2.2.4 Experiments with routine tasks and team familiarity .......................... 129
- 6.2.2.5 Experiments with busyness and team familiarity ............................... 132

### Chapter 7: Research Findings

#### 7.1 Social learning modes, busyness level, and level of team familiarity

- 7.1.1 Learning modes, busyness level and team performance ......................... 134
- 7.1.2 Learning modes, busyness level and TMMs .............................................. 136
- 7.1.3 Learning modes, team familiarity and team performance ...................... 137

#### 7.2 Social learning modes and team structure

- 7.2.1 Team structure, learning modes and team performance ......................... 143
- 7.2.2 Team structure, learning modes and TMM formation .............................. 144
- 7.2.3 Team structure and efficiency of formed TMM ................................. 145

#### 7.3 Social learning and task types

- 7.3.1 Task types, learning modes and team performance ............................ 151
- 7.3.2 Task types, busyness level and team performance ............................ 152
- 7.3.3 Task types, busyness level and TMM formation ............................ 153
- 7.3.4 Task types, team familiarity and team performance ............................ 154
- 7.3.5 Task types, team structure and team performance ............................ 155

### Chapter 8: Conclusions, Limitations and Future works

- 7.3.6 Task types, team structure and TMM formation ............................ 156
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>8.1</td>
<td>Review of research objectives</td>
<td>158</td>
</tr>
<tr>
<td>8.2</td>
<td>Summary of results</td>
<td>161</td>
</tr>
<tr>
<td>8.3</td>
<td>Strengths and limitations</td>
<td>164</td>
</tr>
<tr>
<td>8.4</td>
<td>Future research</td>
<td>166</td>
</tr>
<tr>
<td>8.4.1</td>
<td>Short-term extension</td>
<td>166</td>
</tr>
<tr>
<td>8.4.2</td>
<td>Long-term extension</td>
<td>168</td>
</tr>
<tr>
<td>8.4.3</td>
<td>In the end</td>
<td>170</td>
</tr>
</tbody>
</table>

References ........................................................................................................................................ 171
Glossary ............................................................................................................................................... 179
## Table of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 2.1</td>
<td>Types of team structures</td>
<td>31</td>
</tr>
<tr>
<td>Figure 2.2</td>
<td>Indicative mapping for required agent details to environmental complexity</td>
<td>42</td>
</tr>
<tr>
<td>Figure 3.1</td>
<td>Schematic representation of the research framework</td>
<td>46</td>
</tr>
<tr>
<td>Figure 3.2</td>
<td>Hypothesized influence of busyness on performance across the learning modes</td>
<td>48</td>
</tr>
<tr>
<td>Figure 3.3</td>
<td>Hypothesized influence of busyness on TMM formation across the learning modes</td>
<td>49</td>
</tr>
<tr>
<td>Figure 3.4</td>
<td>Hypothesized influence of team familiarity on performance across the learning modes</td>
<td>50</td>
</tr>
<tr>
<td>Figure 3.5</td>
<td>Hypothesized correlation of team familiarity and busyness in terms of performance</td>
<td>50</td>
</tr>
<tr>
<td>Figure 3.6</td>
<td>Hypothesized correlation of team structure and learning modes in terms of performance</td>
<td>51</td>
</tr>
<tr>
<td>Figure 3.7</td>
<td>Hypothesized correlation of team structure and learning modes in terms of TMM</td>
<td>52</td>
</tr>
<tr>
<td>Figure 3.8</td>
<td>Hypothesized correlation of team structure and busyness in terms of team performance</td>
<td>53</td>
</tr>
<tr>
<td>Figure 3.9</td>
<td>Hypothesized correlation of team structure and busyness in terms of TMM formation</td>
<td>54</td>
</tr>
<tr>
<td>Figure 3.10</td>
<td>Hypothesized correlation of familiarity and team structure in terms of performance</td>
<td>54</td>
</tr>
<tr>
<td>Figure 3.11</td>
<td>Hypothesized correlation of task types and learning modes in terms of performance</td>
<td>55</td>
</tr>
<tr>
<td>Figure 3.12</td>
<td>Hypothesized correlation of busyness and team performance for different task types</td>
<td>56</td>
</tr>
<tr>
<td>Figure 3.13</td>
<td>Hypothesized correlation of busyness and TMM formation for different task types</td>
<td>57</td>
</tr>
<tr>
<td>Figure 3.14</td>
<td>Hypothesized correlation of team familiarity and performance for different task types</td>
<td>58</td>
</tr>
<tr>
<td>Figure 3.15</td>
<td>Hypothesized correlation of team structure and performance for different task types</td>
<td>58</td>
</tr>
<tr>
<td>Figure 3.16</td>
<td>Hypothesized correlation of team structure and TMM formation for different task types</td>
<td>59</td>
</tr>
<tr>
<td>Figure 4.1</td>
<td>Social learning opportunities in a team environment</td>
<td>65</td>
</tr>
<tr>
<td>Figure 4.2</td>
<td>Matrix of solution space for a decomposable task</td>
<td>70</td>
</tr>
<tr>
<td>Figure 4.3</td>
<td>Sequential and parallel task allocations</td>
<td>72</td>
</tr>
<tr>
<td>Figure 5.1</td>
<td>Simulation environment implemented in JADE</td>
<td>76</td>
</tr>
<tr>
<td>Figure 5.2</td>
<td>Activity diagram for the R-Agents</td>
<td>78</td>
</tr>
<tr>
<td>Figure 5.3</td>
<td>Matrix representing the TMM of the R-Agents</td>
<td>79</td>
</tr>
<tr>
<td>Figure 5.4</td>
<td>Activity diagram for a team agent (non-routine task)</td>
<td>85</td>
</tr>
<tr>
<td>Figure 5.5</td>
<td>Pseudo codes for selecting task for rework (non-routine tasks)</td>
<td>86</td>
</tr>
<tr>
<td>Figure 5.6</td>
<td>Matrix representing the TMM of an agent working on non-routine tasks</td>
<td>87</td>
</tr>
<tr>
<td>Figure 5.7</td>
<td>Pseudo code for update of acceptable solution range</td>
<td>89</td>
</tr>
<tr>
<td>Figure 5.8</td>
<td>Capability of each agent is defined by a typical solution span</td>
<td>90</td>
</tr>
<tr>
<td>Figure 5.9</td>
<td>Pseudo code for selection of agent for task allocation (non-routine task)</td>
<td>91</td>
</tr>
<tr>
<td>Figure 5.10</td>
<td>Pseudo code for selection of task for rework</td>
<td>91</td>
</tr>
<tr>
<td>Figure 5.11</td>
<td>Pseudo code for selection of solution</td>
<td>92</td>
</tr>
<tr>
<td>Figure 5.12</td>
<td>Learning opportunities in a team environment</td>
<td>93</td>
</tr>
</tbody>
</table>
Figure 5.13: Typical interaction between two agents
Figure 5.14: Pseudo code for bid selection in non-routine tasks
Figure 5.15: Bids received by Client Agent compared against the desired range
Figure 5.16: Activity diagram for the Client Agent (routine task)
Figure 5.17: Activity diagram for the Client Agent (non-routine task)
Figure 5.18: Activity diagram for simulation controller
Figure 5.19: Interaction protocol among all agent types during the simulation lifecycle
Figure 5.20: Critical network formed because of prior-acquaintance
Figure 6.1: Dependencies in non-routine task used in the simulations
Figure 6.2: Pattern of message exchange across teams working on non-routine tasks
Figure 6.3: Pattern of message exchange across teams working on routine tasks
Figure 7.2: Busyness levels and TMM formation across different learning modes
Figure 7.3: Team familiarity and team performance across different learning modes
Figure 7.4: Team familiarity and team performance for agents (Routine task)
Figure 7.5: Team familiarity and busyness levels in terms of team performance
Figure 7.6: Team structure and modes of learning in terms of team performance
Figure 7.7: Team structure and modes of learning in terms of level of TMM formation
Figure 7.8: Team structure and efficiency of formed TMM
Figure 7.9: Team structure and % TMM formed
Figure 7.10: Team structure and % important TMM formation
Figure 7.11: Team structure and busyness levels in terms of team performance
Figure 7.12: Team structure and busyness in terms of TMM formation (Non-routine task)
Figure 7.13: Team familiarity and team structure in terms of team performance
Figure 7.14: Task types and learning modes in terms of team performance
Figure 7.15: Busyness levels and team performance for different task types
Figure 7.16: Busyness levels and level of TMM formation for different task types
Figure 7.17: Team familiarity and team performance for different task types
Figure 7.18: Team structure and team performance for different task types
Figure 7.19: Team structure and level of TMM formation for different task types
Table of Tables

Table 3.1: Matrix of hypotheses being investigated ................................................................. 60
Table 4.1: Team types and corresponding scope for task allocation or social observation .......... 64
Table 4.2: Causal relationships between agents’ enabling factors and actions ......................... 67
Table 5.1: Learning assumptions corresponding to learning opportunities shown in Figure 5.12 .... 94
Table 5.2: Parameters in a typical FIPA-ACL message envelope .............................................. 95
Table 5.3: Types of messages used and their description ............................................................. 97
Table 5.4: Implementing observations: conditions and updates .................................................... 101
Table 6.1: Summary of the number of messages exchanged in training set ................................ 114
Table 6.2: Summary of the TMM formation after training (60 runs) ............................................ 114
Table 6.3: Summary of the number of messages exchanged in test set (60 runs) ......................... 114
Table 6.4: Effects of social learning and individual learning ...................................................... 116
Table 6.5: Effects of team size across agents with social and individual learning ....................... 116
Table 6.6: Experiment matrix showing the combination of parameters used in different simulations 117
Table 6.7: Team compositions used for simulations with the routine tasks .................................. 119
Table 6.8: Team compositions used for simulations with non-routine tasks ............................... 121
Table 6.9: Experiments with routine tasks and busyness (15 agents, Set 1, Table 6.7) ............... 123
Table 6.10: Experiments with routine tasks and busyness (12 agents, Set 2, Table 6.7) ............. 124
Table 6.11: Experiments with non-routine tasks and busyness (12 agents) ............................... 126
Table 6.12: Experiments with routine tasks and team familiarity (Set 2, Table 6.7) ................. 127
Table 6.13: Experiments with non-routine tasks and team familiarity ........................................ 129
Table 6.14: Experiments with busyness and team familiarity (Set 2, Table 6.7) ......................... 132
Table 7.1: Difference in team performance across busyness levels (0, 25, 33, 50, 66 and 75%) .... 135
Table 7.2: Effects of team familiarity on team performance across the learning modes ............... 139
Table 7.3: Team familiarity and team performance (BL=0%) ..................................................... 139
Table 7.4: Effects of Team familiarity on team performance across busyness levels ................. 141
Table 7.5: Team performance across busyness levels at given team familiarity .......................... 142
Table 7.6: Differences in team performance across learning modes .......................................... 144
Table 7.7: Efficiency of TMM for teams working on routine task ............................................. 145
Table 7.8: Efficiency of TMM for teams working on non-routine task (TP is normalized) .......... 145
Table 7.9: Comparison of correlation of TMM and important TMM with team performance ....... 150
Table 7.10: Decrease in the TMM with the increase in BL ....................................................... 153
Table 8.1: Results for tested research hypotheses ...................................................................... 161
**List of Abbreviations and Symbols**

<table>
<thead>
<tr>
<th>Abbreviation</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>AMM</td>
<td>Agent mental model</td>
</tr>
<tr>
<td>AMS</td>
<td>Agent Management System</td>
</tr>
<tr>
<td>$A^{R_1}$</td>
<td>R-Agents that learn only from personal interactions, i.e., PI</td>
</tr>
<tr>
<td>$A^{R_2}$</td>
<td>R-Agents that have all modes of social learning available to them, i.e., PI+IO+TO</td>
</tr>
<tr>
<td>BL</td>
<td>Busyness levels</td>
</tr>
<tr>
<td>CMOT</td>
<td>Computational and mathematical organization theory</td>
</tr>
<tr>
<td>DF</td>
<td>Directory Facilitator agent</td>
</tr>
<tr>
<td>$d_{max}$</td>
<td>Used to identify the solution with value furthest from the mean of the acceptable range</td>
</tr>
<tr>
<td>$E_{ij}$</td>
<td>Element in the $i$th row and $j$th column of matrix $E$</td>
</tr>
<tr>
<td>FIPA</td>
<td>Foundation for Intelligent Physical Agents</td>
</tr>
<tr>
<td>$G^T$</td>
<td>Given (used for update of TMM)</td>
</tr>
<tr>
<td>$g$</td>
<td>Number of equal-sized task-groups</td>
</tr>
<tr>
<td>Grp_1</td>
<td>Group name used for affiliation of agents in simulation environment</td>
</tr>
<tr>
<td>IO</td>
<td>Interaction observations</td>
</tr>
<tr>
<td>JADE</td>
<td>Java Agent Development Environment</td>
</tr>
<tr>
<td>LM</td>
<td>Learning mode</td>
</tr>
<tr>
<td>$L^{max}$</td>
<td>Theoretical upper limit of the number of messages exchanged before the task is complete</td>
</tr>
<tr>
<td>$L^{max-cal}$</td>
<td>The calculated $L^{max}$ for a team with given expertise distribution</td>
</tr>
<tr>
<td>$L_R$</td>
<td>Lower range</td>
</tr>
<tr>
<td>$L_{Rmin}$</td>
<td>The minimum possible lower range for the solutions of a task</td>
</tr>
<tr>
<td>$C_{L_R}$</td>
<td>Acceptable lower range of solution for Client Agent</td>
</tr>
<tr>
<td>$Cur_{L_R}$</td>
<td>Current lower range in TMM (hypothesized) for an agent in a given task</td>
</tr>
<tr>
<td>$Cur_{L_O}$</td>
<td>Lowest range observed and registered (not hypothesized) for an agent in a given task</td>
</tr>
<tr>
<td>$i_{L_R}$</td>
<td>Lower range of proposed solution in $i$th bid of the bidlist</td>
</tr>
</tbody>
</table>
**L_T**  
Temporary lower value of solution range  

**MAS**  
Multi Agent System  

**NR-Agent**  
Agents working on non-routine tasks  

**N^A**  
Number of agents in the team  

**#N^A**  
Number of agents in each of the g groups  

**N^T**  
Total number of tasks in the team  

**kN^T**  
The number of tasks to be performed by kth group  

**N^TP (N^A_p)**  
There are N^TP tasks for which there are N^A_p agents than can perform the task  

**O**  
The maximum value for the number of messages observed  

**P^T**  
Performed (used for update of TMM)  

**PI**  
Personal interactions, also used for “learning from personal interactions”  

**PI+IO**  
Learning from personal interactions as well as interaction observations  

**PI+TO**  
Learning from personal interactions as well as task observations  

**PI+IO+TO**  
Learning from personal interactions, interaction observations as well as task observations  

**Q**  
Number of sub-tasks for TaskToCoordinate  

**R-Agent**  
Agents working on routine tasks  

**S^i**  
Solution for ith task, T^d  

**T**  
Task  

**T^d**  
Task  

**sT^r**  
Competence of the sth agent for the rth task, T^r  

**T^j(i)**  
Solution to sub task T^i with the value j  

**Task 1_c**  
Task nomenclature used in simulations with routine tasks such that agents can identify task-related groups. Since this task has 1 before underscore, it is related to Grp_1.  

**T^m [3;4]**  
Agent has a capability range 3-4 in the task T^m. This symbol is used for simulations with non-routine tasks.  

**T^m,b [3;8]**  
Agent has a capability range 3-8 in the task T^m,b. This symbol is used for simulations with non-routine tasks. The underscore in the superscript is used to identify the higher level task and group that this task belongs to. Thus, the “b” before underscore is used to
identify the group to which the task belongs, i.e., Grp_b. The “mb” before underscore is used to identify the higher level task, i.e., Tmb_b is a sub-task generated from task Tmb. Similarly, Tma_b is a task related to group Grp_a, and is generated from task Tma_a.

TF
Team familiarity

TMM
Team mental model

TO
Task observations

TS
Team Structure

[T', Lr, U_r]
Values for the competence (T'), lower range (Lr) and upper range (U_r) of the sth agent in the rth task

U_buffer
Current difference between the current upper range of expected solution span and the observed value in the solution provided.

UR
Upper range

URmax
The maximum possible upper range for the solutions of a task

UR
Acceptable upper range of solution for Client Agent

CUR
Current upper range in TMM (hypothesized) for an agent in a given task

CurUR
Highest range observed and registered (not hypothesized) for an agent in a given task

iUR
Upper range of proposed solution in the ith bid of the bidlist

UT
Temporary upper value of solution range

Vs
Value of overall solution
Chapter 1
Introduction

Learning is a basic skill, essential to the development of other skills such as design and teamwork. Learning accelerates progress, and forms an integral part of skill development in teams and organizations. Numerous organizational training and learning programs have been developed. Yet, training and “learning to learn” remains a challenging endeavour (Conlon, 2004). In contrast, social learning skills need not be trained because they are naturally acquired by humans (Tomasello, 1999). Social learning is embedded in the environment, and it is as much involuntary as goal-directed (Marsick & Watkins, 1997). Therefore, this research explores the role of social learning in the formation of team mental models and the team performance.

A computational model based on the conceptual foundations of the folk theory of mind (Gordon, 2009; Knobe, 2006; Malle, 2005; Ravenscroft, 2004; Tomasello, 1999) is developed. This model is used as a simulation test-bed to establish the role of social learning in the formation of team mental model and team performance across different cases. Three different social learning modes are discretely represented to include (1) learning from personal interactions, (2) learning from task observations, and (3) learning from interaction observations. This allows controlled investigation of the role of each learning mode in the formation of team mental models. A computational approach also allows control on what the agents learn. All the agents are assumed to be domain experts. Thus, during the simulations, agents learn only about the expertise of self and the other agents in the team while their knowledge about the task or processes, which are pre-coded into the agents, remain fixed. Agents’ ability to learn from social observations is mitigated by their cognitive busyness. Busyness determines whether an agent is able to attend to an environmental event or stimuli available a given instance. Team structure and team familiarity are the other variables considered in these simulations. Correlation of social learning modes,
formation of team mental models and team performance are investigated across the different team structures, the levels of team familiarity, the busyness levels of the agents, and the task types.

Prior research suggests that the team mental models mediate team performance. However, knowledge elicitation from team members and the assessment of formed team mental models remains the main challenge in studies with human subjects (Mohammed, Klimoski, & Rentsch, 2000). This research is built on the premise that computational simulations can provide useful insights into the research on team mental models while reducing some of the limitations with knowledge elicitation and representation. A computational approach is particularly suitable to study the effects of the different modes of social learning because these are difficult to control in practice.

1.1 Motivation

Designing is increasingly a team activity. Knowledge about the tasks to be performed, and the domain expertise, is distributed across a team. The Literature (Cross & Cross, 1998; LaFrance, 1989) suggests that experts possess both domain knowledge as well as tactical knowledge. In a team environment, this also includes the knowledge about the other team members and their competence. Thus, a mere collection of individual experts is not enough to produce an expert team (Candy & Edmonds, 2003). Team expertise is developed as the team members form different mental models based on the task, context, process, team membership and competence (Badke-Schaub, Neumann, Lauche, & Mohammed, 2007; Cannon-Bowers, Salas, & Converse, 1993; Mohammed et al., 2000), to perform the task collectively. Studies show that team mental models (TMM) mediate team performance (Klimoski & Mohammed, 1994; Langan-Fox, Anglim, & Wilson, 2004; Lim & Klein, 2006).

A TMM is defined as an individual agent’s knowledge of its own competence and the competence of all the other agents in the team to perform the different tasks (section 5.3.2, section 5.4.2). If an agent cannot perform a specific task, a well-developed TMM allows it to allocate the task to the agent that is most competent in performing the given task. The importance of knowing the knowledge source in a distributed system has also been emphasized in the research on transactive memory systems (Wegner, 1987).

This knowledge about each other is achieved through social interactions and observations. As proposed in the folk theory of mind (Gordon, 2009; Knobe, 2006; Malle, 2005; Ravenscroft, 2004; Tomasello, 1999), during these social interactions and observations, the ability of individuals to identify others as intentional beings, similar to them, allows the team members to
make assumptions (attributions) about each other. These assumptions facilitate social learning, and learning about each other’s mental states. Social learning contributes to the formation of a TMM (Badke-Schaub et al., 2007; Langan-Fox et al., 2004; Mohammed et al., 2000), thereby influencing the team performance. Studies have been conducted to investigate the relationships between TMMs and team performance, and different modes of social learning have been reported (Grecu & Brown, 1998, Wu, 2004). However, the contribution of the different modes of social learning to the development of the TMMs and the team expertise needs to be established.

The role of social learning in TMM formation and to the team performance may be influenced by various factors related to the team or to the team members. How the team is organized may determine the social learning opportunities. TMM formation and team performance may also be affected by the number of members that have worked together previously. Therefore, team structure and team familiarity are team level factors considered in this study. TMM formation and team performance may also be influenced by the learning abilities of each member as well their busyness levels, i.e., what social learning opportunities they attend to. Hence, learning modes and busyness are also considered as a parameter in this study. The requirements for TMM may vary with the task. Hence, task type is also considered in this study.

An enhanced understanding of the contribution of the different learning modes (section 3.2.1, section 5.5) will be useful for effective team organization and information management. If the team is flat, all the team members can interact with and observe all the other team members. If the team is organized into task-based sub-teams, the interactions and observations are primarily confined to the sub-team to which the member belongs. However, in collaborative projects, the quality of interaction amongst the team members may also vary with the mode of interaction (DeSanctis & Monge, 1999). The members of a collocated team may have most modes of social learning available to them because they can have face-to-face interactions. In contrast, the members of a non-collocated or virtual team, interacting across technological tools, may only have some of the social learning modes available to them (Beekhuyzen et al., 2006; DeSanctis & Monge, 1999; Hertel et al., 2005; McDonough et al., 2001). Such teams are distributed across geographies and they may have flexible team structure (DeSanctis & Jackson, 1994; Katzy, 1998; McDonough et al., 2001). In the non-collocated and virtual teams, the information exchange is discretely represented in some form, often limiting the modes of social learning (DeSanctis & Monge, 1999). Hybrid team structures may exist where the team is flat but distributed in non-collocated social cliques. In such teams, the members can interact with and observe all the collocated members but their opportunity for social learning about non-collocated team members may be limited. Thus, this research claims that the difference in the team structures is likely to
influence the formation of TMM, and may reflect in the team performance. Hence, the role of social learning across the different team structures are investigated (section 7.2). Findings from this research will be useful in the design and management of distributed and virtual teams.

Project-based teams are commonplace in large organizations as well as virtual teams (Devine et al., 1999; Hackman, 1987; Laubacher & Malone, 2002; Lundin, 1995; Packendorff, 1995). Team composition may vary and effect the formation of TMMs and the team performance in such teams. To achieve higher team performance, managers and project leaders strive to maximize the number of agents in the team with prior-acquaintance (or higher team familiarity), mostly in the form of agents who have previously worked together on a similar project (Hinds et al., 2000). However, it may not always be possible to form a team with high levels of team familiarity. Therefore, this research enhances our understanding of the significance of team familiarity in different team environments by exploring the relationship between the modes of social learning, the levels of team familiarity and the team performance (section 7.1.3).

In project-based teams in organizations, the team members may be engaged in multiple projects or other activities (Mcgrath, 1991). This busyness of the agent may influence the TMM formation and the team performance because the agent’s attention is diverted from the team activities, and the activities of the other agents (Gilbert & Osborne, 1989; Griffiths et al., 2004). Hence, this research explores the correlation of busyness levels, TMM formation and the team performance (section 7.1.1, section 7.1.2 and section 7.1.4). Findings of the study will be useful in understanding the influence of work load distribution and social engagement of the team members.

It is established that the team performance is affected by the TMM. However, the effects of the TMM on the team performance may vary with the task (Badke-Schaub et al., 2007; Mohammed & Dumville, 2001). Hence, TMM formation and team performance are assessed for the different task types (section 7.3). Understanding the relationship between the social learning modes, TMM formation and the team performance, in different task types, will facilitate the design managers to adapt their strategies to suit the design task.

Conceptually, this research is based on the foundations of the folk theory of mind, and methodologically, a computational approach is adopted. Section 1.1.1 provides a discussion on the conceptual motivations, and section 1.1.2 discusses the methodological motivations.

\[1\] In this thesis, team familiarity and prior-acquaintance are used interchangeably. Prior-acquaintance is used to refer to dyadic relationships, while team familiarity is used at the collective level. However, higher team familiarity need not necessarily mean prior-acquaintance between all the agents that were part of the same team earlier. Hence, all experiments and hypotheses are discussed in terms of levels of team familiarity and not prior-acquaintance.
1.1.1 Conceptual motivation

The research on TMM is rich and diverse, spanning across the disciplinary boundaries. Emphasizing the complexity of TMM, the current research in the field exhibits the following two trends:

1. Increasingly, a reductionist approach is being adopted that aims to distinguish the TMM from the mental models for task, process, context, and so on (Badke-Schaub et al., 2007; Cannon-Bowers et al., 1993; Druskat & Pescosolido, 2002; Langan-Fox et al., 2004; Mohammed & Dumville, 2001).

2. Greater emphasis is made on the need for multiple measures of TMM (Langan-Fox et al., 2001; Mohammed et al., 2000; O’Connor et al., 2004; Webber et al., 2000). This, in turn, highlights the concerns over the inaccuracies and difficulties in TMM measurement.

This research aims to address these two issues. Firstly, following the recent literature, this research distinguishes the TMM from the other mental models and specifically aims to investigate the theories on the formation of TMM. The issues on TMM measurement are addressed by the choice of the research methodology, as discussed in section 1.1.2.

In general, the TMM is viewed as a social and cognitive construct (Klimoski & Mohammed, 1994). The concepts of social cognition have significant overlap with the folk theory of mind (Malle, 2005) and the attribution theory (Irene Frieze, 1971; Jones & Thibaut, 1958). Yet there is little research on the contribution of common sense psychology\(^2\) of individuals and the attribution behaviour (Malle, 2005) in the formation of TMMs. Thus, while the significance of social learning is acknowledged (Druskat & Pescosolido, 2002; Langan-Fox et al., 2004), the contribution of the different social learning modes has not received enough attention. This research aims to investigate the contribution of each of the social learning modes, namely, learning from personal interactions, learning from task observations, and learning from interaction observations, to the formation of TMMs. As discussed in section 4.1.2, adopting a folk theory of mind and attribution theory as the conceptual underpinning facilitates the research enquiry because it allows a clear distinction between each of the learning modes. These learning modes can be represented as simplified rules based on assumptions (attributions) relying on a behavioural explanation (of intentionality and identification with the others as conspecifics).

---
\(^2\) The term common sense psychology (Ravenscroft, 2004) is used interchangeably with the term folk theory of mind
1.1.2 Methodological motivation

In adopting a computational approach, the motivation is to develop a test-bed that facilitates data collection (knowledge elicitation and representation), provides greater control on parameters, flexibility in team composition, and scalability for future research.

Data collection and analysis

Prior research suggests that team mental models mediate team performance (Lim & Klein, 2006; Ren, Carley, & Argote, 2006; Rouse, Cannon-Bowers, & Salas, 1992). The measures for team performance may include the evaluation of the overt factors such as task quality, team process and time (Ancona & Caldwell, 1989). However, measuring the TMM, which is viewed as a cognitive and a social construct (Klimoski & Mohammed, 1994), remains a challenging endeavour (Cooke et al., 2004; Klimoski & Mohammed, 1994; Langan-Fox et al., 2001; Mohammed et al., 2000). Various techniques have been proposed for measuring the TMM such as Pathfinder (Langan-Fox et al., 1999; Lim & Klein, 2006), multi-dimensional scaling (Mohammed et al., 2000), concept mapping (O’Connor et al., 2004), and so on. Mohammed et al. (2000) argue that measures for review of the team mental models should encompass both knowledge elicitation and representation. According to Mohammed et al. (2000), knowledge elicitation refers to the techniques used to determine the contents of the mental model (data collection), while knowledge representation refers to the techniques used to reveal the structure of data or determine the relationships between the elements in an individual’s mind (data analysis). In real world studies on TMMs, both the knowledge elicitation and the knowledge representation techniques are subjective, and prone to incompleteness and inaccuracies.

In contrast, a computational study allows objective determination of the agents’ knowledge content as well as the representation. In the computational models, changes to the agent’s knowledge base can be accurately traced and registered. Similarly, the agents can be designed so that the knowledge representation is also completely and accurately known.

The computational approaches adopt a simplified representation of the cognitive processes and abilities of the agent. However, the ability to accurately elicit the observable research data, and the contributions of computational studies to the theories in sociology and organizational science (Carley, 1994; Carley & Newell, 1994; Edling, 2002; Lant, 1994; Macy & Willer, 2002), provide a strong motivation for a complementary approach to the research on TMMs. Further, Ren et al. (2001; 2006) and Schreiber and Carley (2004) have demonstrated the usefulness of the computational models in the study of TMMs and transactive memory systems.
Control, flexibility and scalability

By considering different “what if” scenarios the research method can investigate the contribution of each learning mode specifically to the formation of knowledge about the competence of the other agents in the team. Hence, it is desired that each agent’s knowledge about the task and the processes remain fixed throughout the study. A computational approach allows control on all the experiment parameters and also on what the agents know and what they learn. This may not be possible in a real world study. A computational approach assures that the other social factors such as trust and motivation, which are not considered, do not implicitly influence the results.

Investigating “what if” scenarios require the flexibility to simulate the different experiment conditions, through combination and superposition of the team parameters (levels of team familiarity, team structure), and the agent parameters (learning modes, busyness levels). A computational approach provides this flexibility. The flexibility to scale up the computational model will be useful for planned future research. This may include adding more learning assumptions, use of cognitively richer agents that also learn about the tasks and the team processes, larger team sizes, and so on.

Further, there are five independent variables and two dependent variables in this research. With the different combination of the values for the independent variables, 288 experiments are conducted (section 6.2, Table 6.6), each multiple number (60) of times. In real world scenarios, setting up these combinations, conducting experiments, and collecting data is highly resource intensive and may not be practically feasible.

Methodological approach

Use of computational simulations is a well established research method across various disciplines in the social sciences and the organizational studies. However, few examples of computational studies are reported in the research on TMM (Ren et al., 2001; Ren et al., 2006). This research lays the foundation for computational studies of the theories on TMM. A computational approach eliminates some of the knowledge elicitation and representation issues typically faced in studies of the TMM in real world environments where accuracy and completeness of the research data is difficult. In this model, the TMM is represented as a matrix (Section 5.3.2, Section 5.4.2). Thus, the structure of the agents’ TMM and its default state are already known. Hence, changes to the agents’ TMM can be accurately traced and registered (Section 5.3.4). Similarly, agents’ knowledge and actions are based on the pre-defined causal relationships (learning rules and
assumptions) (section 4.1.3, Table 4.2, section 5.5, Table 5.1). This means the knowledge representation is also completely and accurately known.

Methodologically, this research is similar to Ren et al. (2001; 2006). However, this computational model is specifically developed to explore the theories on TMM from the perspective of the folk theory of mind. Computationally, this model also differs in that it distinctly represents the different social learning modes (section 5.6, Table 5.4) that are taken as experiment parameters.

1.2 Aim

The aim of this research is to explore the role of social learning in the formation of team mental models and team expertise using a computational test-bed.

1.3 Objectives

The objectives of the thesis are:

1. To develop a conceptual framework of social learning, formation of TMMs and team performance, in project teams (Chapter 1).

2. To identify and represent the different modes of social learning such that their influence on team performance and formation of TMM can be studied separately, and through superposition.

3. To include team structures, team familiarity and busyness (of agents) in the computational framework, as factors associated with social learning in teams, such that their correlation with the formation of TMMs and the team performance can be explored (Chapter 4).

4. To develop a symbolic representation of routine and non-routine tasks such that it captures some of the basic differences between the two task types (Chapter 4).

5. To implement, validate and test a simulation environment for the computational framework and representations discussed in objectives 1 to 4 (Chapter 5, Chapter 6).

6. To use the implemented simulation environment in furthering the understanding of the correlation between the social learning, the formation of TMMs, and the team performance (Chapter 6, Chapter 7).
1.4 Research claims, contributions and significance

1.4.1 Conceptual framework

From a conceptual viewpoint, this research has its roots in the folk theory of mind (or common sense psychology) (Knobe, 2006; Ravenscroft, 2004; Tomasello, 1999) and the attribution theory (Jones, 1958; Wallace, 2009; Irene Frieze, 1971; Iso-Ahola, 1977). These theories emphasize simplified models, rules, assumptions (attributions) and strategies that individuals adopt while interacting and dealing with their complex social environment such as teams and organizations (Levitt & March, 1988; Schwenk, 1995). Based on the assumptions of intentionality, and identification with the conspecifics (Tomasello, 1999), this thesis explores the contribution of simple deductive rules (typically made in social interactions) (section 5.5) to the formation of TMMs. For example, if an agent $A^3$ observes agent $A^1$ allocating a task $T^1$ to agent $A^2$, $A^3$ updates its TMM assuming that $A^1$ cannot perform the task $T^1$.

The validation of the conceptual framework is facilitated by the use of a computational approach that allows focusing on the specific aspects of TMMs (who knows what and who has what capability range in the tasks one can perform). The conformity of the research findings to the literature on TMM validates the usefulness of the adopted conceptual framework in advancing the theories on TMM.

The findings from this research support earlier observations that the TMM mediates team performance (section 7.2.6). However, the efficiency of TMM formation, in terms of their effects on team performance, varies with the team structure (section 7.2.3). The efficiency of TMM is higher in the teams organized as task-based sub-teams (section 7.2.3). In general, social observations (task observations and interaction observations) enhance the formation of TMM as well as the team performance. However, the contribution of interaction observations to increasing the team performance is significantly less than the contribution of task observations (section 7.1.3).

The team performance increases with the increase in the levels of team familiarity (section 7.1.3). However, the pattern of increase in the team performance, with the increase in the levels of team familiarity, is contingent on the task type, and the learning modes. In general, there exists a threshold point beyond which, the rate of increase in the team performance, with the increase in the levels of team familiarity, is higher. As the task complexity increases, this threshold point tends to move towards 100% team familiarity. The increase in the team performance, with the increase in team familiarity, tends to be more uniform when all modes of learning are available to the agents. In general, the rate of increase in the team performance, with the increase in team
familiarity, is greater at higher levels of team familiarity. The knowledge of the contingency factors, and their effects on the TMM formation and the team performance, will be useful for managers to adopt different team management and information management strategies, to suit their project requirements and task needs.

In general, the busyness levels have no significant effect on the team performance (section 7.1.1). However, TMM formation decreases significantly with the increase in busyness levels (section 7.1.2). The findings related to busyness levels are particularly useful for the organizations where the employees are simultaneously engaged in multiple projects.

The teams organized into task-based sub-groups show higher team performance than the flat teams or the flat teams with social cliques (section 7.2.1). However, TMM formation is highest in the flat teams, followed by the flat teams with social cliques, and then by the teams organized as task-based sub-teams (section 7.2.2).

1.4.2 Computational modelling

A computational model based on simple reactive agents is implemented (Chapter 5). These agents learn about each other based on basic “If-Then” rules (Table 5.1), which allows explicit representation of each of the learning modes separately (section 5.6, Table 5.1, Table 5.4). Thus, learning from personal interactions, task observations, and interaction observations are experiment parameters that can be independently investigated, and superposed according to the experiment requirements.

The learning rules for the agents are based on the folk theory of mind. However, rather than reasoning about intentionality [as in BDI agents (Rao & Georgeff, 1995)], the agents learn based on assumptions of intentionality of the others’ actions (section 5.5). The pre-defined causal relationships that determine the TMM formation (section 5.3.2, section 5.4.3) ensures that knowledge representation and accuracy of the TMM measurement is not a concern when analyzing the results. The representation of TMM as a matrix of elements, representing the competence of each of the team members in each of the tasks (section 5.3.2, section 5.4.2), facilitates complete extraction of the TMM (section 5.3.4).

This model is specifically aimed at studying the relative contribution of the social learning modes on TMM formation and the team performance. Team performance is measured in terms of the number of messages exchanged between the agents. A log of the messages exchanged between the agents allows measurement of the team performance. The findings from the preliminary simulations conform to the literature (section 6.1). This validates the underlying assumption that a computational model with simple reactive agents that learn from rules,
grounded in the folk theory of mind, simulates the social behaviour intended to be studied using the conceptual framework discussed in section 1.4.1.

1.5 Thesis structure

Chapter 2 presents a review of the related literature that provides the basis for the research framework and the research hypotheses discussed in Chapter 3. Chapter 4 discusses the conceptual framework and the modelling decisions for the development of the computational model. The conceptual framework is developed such that all the independent and dependent variables identified in Chapter 3 are incorporated into the framework. Chapter 5 presents the details of the implementation of the computational model. Chapter 6 presents the details of the simulations and experiments. Section 6.1 details the scenarios and the results of the experiments conducted to validate the computational model. Section 6.2 details the experiment scenarios used to test the research hypotheses, and presents the analysis of the empirical data. A discussion on the research findings is presented in Chapter 7 with respect to the research hypotheses proposed in Chapter 3. Chapter 8 is the concluding chapter that provides a brief review of the research objectives and a summary of the research findings, followed by a discussion on the limitations of this thesis and possible future works.
Chapter 2

Background

Team work and team building is a social process that develops over time as the team members gain experience working with each other (Cross & Clayburn-Cross, 1995; Tuckman, 1965). This teamwork and team building process also applies to the design teams (Cross & Clayburn-Cross, 1995), which may also be influenced by the nature of the design task and the structural characteristics of the team, such as the team size and the team structure. Hence, the research on teamwork and team building in design teams draws from the various complementary fields, such as design research, social cognition, social networks, social and organizational learning, and social and organizational behaviour. This chapter presents a review of the literature from these complementary research areas. This review provides the basis for the research framework and the research hypotheses discussed in Chapter 3, and the modelling assumptions made in the computational model discussed in Chapter 4.

2.1 Social learning and social cognition

The ability of humans to understand others as intentional beings, similar to oneself, allows individuals to learn from social interactions and observations (Knobe & Malle, 2002; Malle, 2005; Ravenscroft, 2004; Tomasello, 1999). Tomasello (1999) differentiates the basic forms of cultural learning as (1) imitative learning i.e., learning by reproducing others intentional actions (2) instructed learning i.e., learning through explicit instructions and guidance, (3) collaborative learning i.e., learning through collective engagement with a common task, and (4) emulative learning, i.e., learning from environmental events (changes in the state of the environment that others produce). Tomasello (1999) claims that in some scenarios emulative learning is more
adaptive than imitative learning. In all these forms of learning, joint attention (Malle, 2005; Tomasello, 1999) plays a critical part in which the learner is concerned with only a subset of all the things that can be perceived at the given moment.

The human ability to understand external events in terms of causal forces, mediated by intention, helps them to solve problems while facilitating social learning (Gordon, 2009; Knobe, 2006; Knobe & Malle, 2002; Malle, 2005; Ravenscroft, 2004; Tomasello, 1999). The understanding of the others as intentional agents requires the understanding of attention, strategies and goals, while the understanding others as mental agents requires the understanding of beliefs, plans and desires (Malle, 2005; Tomasello, 1999). An action is considered intentional “when the agent has a desire for an outcome, a belief that the action would lead to that outcome, an intention to perform the action, the skill to perform the action, and awareness of fulfilling the intention while performing the action” (Malle, 1997; Mele, 2001).

Malle (Malle, 2005) integrates the knowledge of attribution theory, which emphasizes on the cause and effect explanations of social context (Heider, 1958; Jones & Thibaut, 1958), with the folk theory of mind, which is a conceptual framework that relates the different mental states to each other, and connects them to behaviour. Mental states relate to the behaviour, either in the form of unintentional actions caused by internal or external events without the intervention of the agent’s decision, or intentional actions (Heider, 1958). Malle (Malle, 2005) differentiates between intentional and unintentional behaviours in terms of the way they are explained. Unintentional behaviours are explained using mechanical causal factors (causal explanation), while intentional behaviours are explained in various ways that include subjectivity and rationality (reason explanations), causal history of reason explanations, and the enabling factors such as skills and opportunities.

The behaviour explanation varies, depending on whether the events are observable or not, i.e., the publicly observable and the publicly unobservable events affect social cognition differently (Funder, 1987; John, 1993; Malle, 1997).

This research focuses on actions, interactions and observations in teams. Agents learn about the other agents in the team based on the actions of the others, which are observable (Irene Frieze, 1971; Wallace & Hinsz, 2009). Since these actions are assumed to be intentional\(^3\), agents are able to build a mental model of the other agents in the team. Agents learn about themselves based on their own actions and interactions. Observation is subject to an agent’s attention to the observable data, and, hence, mitigated by their level of busyness (Gilbert & Osborne, 1989; Gilbert, Pelham,  

\(^3\) Intentions within a work context, e.g. if an agent can perform a task, it will. Thus, agents do not reason in terms of intentions or beliefs rather they make assumptions of intentionality in actions.
& Krull, 1988). Since this research is focussed on teams, a review of the literature on teams and organizations is presented.

2.2 Teams and organizations

Teams are a subset of groups, and most of the issues relevant to groups are applicable to teams (Klimoski & Mohammed, 1994). However, teams differ from other kinds of groups in the sense that the roles and responsibilities are clearly differentiated in teams (Cannon-Bowers et al., 1993; Klimoski & Mohammed, 1994). Teams are usually formed such that the members have an overarching common goal (Salas et al., 1992). Most organizations use or plan to use teams to achieve their goals (Cohen & Bailey, 1997; Lawler et al., 1992). Various kinds of teams are discussed in the literature (Cohen & Bailey, 1997; Katzenbach JR, 1993; Mohrman et al., 1995; Sundstrom et al., 1990). Some of the typical dimensions across which teams can be differentiated are:

**Team structure e.g. flat teams, hierarchical teams**

Flat teams have no organizational structure, and the interaction is entirely horizontal in the flat teams. Hierarchical teams are organized into vertical layers (two or more), with an appointed team leader. Often, the layers at the bottom of the pyramid are organized into sub-teams, with appointed sub-team leaders, who together form the intermediate layer (Malone, 1987).

Informal social networks and structures may emerge within the formal teams, and they may have a significant role to play in the knowledge distribution and diffusion (Bobrow & Whalen, 2002; Borgatti & Cross, 2003; Brown & Duguid, 2001).

**Location: distributed, collocated, virtual etc**

Teams may be collocated, distributed or virtual. Distributed teams have members working across physical boundaries, which reduce the medium of interaction, while the members of collocated teams often interact face-to-face. Virtual teams are special case of distributed teams in which it is likely that the members may have never met each other in a face-to-face interaction (Griffith et al., 2003; Katzy, 1998; Leinonen et al., 2005; McDonough et al., 2001).

Distributed and virtual teams are generally project-based and may vary in their practice and composition. In some cases, the team members may have occasional face-to-face meetings for project updates and reviews. In other cases, the team members may have never met each other
physically. Such teams may also differ in terms of the information exchange, communication media, and the information dissemination across the team members (Griffith et al., 2003; Katzy, 1998; Leinonen et al., 2005; McDonough et al., 2001). For example, it is possible that all the project-related information is available to all the team members, through group emails, project boards or project wikis. This facilitates social learning and the formation of TMM. It is also possible that the members may only have access to the information related to their roles and responsibilities. This may be coordinated by the project leader, through telephones and personal emails, which reduce the scope for social learning and TMM formation.

Therefore, the use of technology and communication media in project-based teams may determine what modes of social learning are available to the team members. Thus, the teams relying on technology-mediated interactions can be organized in different ways, to suit desired modes of socialization.

**Scale: large scale teams, medium scale teams, small groups**

Large scale teams are common in complex projects and the size of such teams may run into hundreds and thousands. Such teams are mostly organized into hierarchies and sub-teams (Cusumano, 1997; Malone, 1987; Xu et al., 2004). Small teams generally have less than fifteen members (anonymous, 2006; Katzenbach, 1993). In general, for small teams, it is possible to have flat teams as well as teams organized as sub-teams (Katzenbach, 1993). Small teams enhance the likelihood of interactions, observations and cohesion among all the team members (Littlepage, 1991; Littlepage & Silbiger, 1992; Moreland et al., 1998; Wheelan, 2009). In a small flat team, agents can allocate tasks to any other agent in the team. If the agent is not busy, it can also observe the activities of any other agent in the team.

In general, humans have limited cognitive capacity for the size of their effective social network at any given time (Hill & Dunbar, 2003). However, in small teams it is likely that members will have the cognitive ability to maintain a mental model of all the other team members.

**Life-span e.g. regular teams, project-based teams**

Regular teams are those that remain more or less fixed in their composition for multiple projects. Regular teams allow more time for team building but are often criticized for fostering routine output. Hence, teams are increasingly becoming project-based (Devine et al., 1999; Guzzo & Dickson, 1996; Hackman, 1987; Laubacher & Malone, 2002; Lundin & Söderholm, 1995;
Project-based teams can either be in-house, where the members are rotated (co-opted on as-need basis), or collaborative, where the members are drawn from different organizations. In such a scenario, member familiarity and prior-acquaintance are critical factors that may influence the team performance (Huckman et al., 2008; McGrath, 1991).

**Composition: heterogeneous or homogeneous in terms of knowledge, culture, ethnicity etc.**

Teams can be classified as heterogeneous or homogeneous based on different criteria. In general, the homogeneous teams are expected to promote team cohesion. Within the heterogeneous teams, the team members with similarity may tend to form sub-groups and social cliques (Ancona & Caldwell, 1989; Hackman, 1987; Harrison et al., 2003).

This research focuses on small, project-based, work (design) teams, with varying levels of team familiarity. All the agents in a given simulation have similar learning capabilities. The knowledge is distributed across the agents such that each agent has specialized knowledge, different from the others. However, there might be more than one agent with the same specialized knowledge. The social learning opportunities may vary across the team structures, and, hence, team structure is taken as a parameter in this research. The simulated scenarios may correspond to distributed or collocated teams depending on the choice of observation and learning capabilities of the agent. Therefore, a review of the literature related to team structures is presented.

### 2.2.1 Team structures

The three kinds of team structures to be modelled in this research include flat teams, flat teams with social cliques, and the teams organized as task-based sub-teams.

**Flat teams**

Flat teams have no hierarchy and no sub-divisions, Figure 2.1(a). Such teams are generally used for consultation, task-force and design exploration (Katzenbach, 1993; OpenLearn, 2009; Perkins, 2005). Experts are drawn from multiple disciplines. In such teams, it is possible that there are no nominated leaders. A leader may emerge over time, based on the interactions within the team.

**Teams organized as task-based sub-teams**

---

4 Experimenter’s choice
Many work teams are organized into expertise-based sub-teams (functional teams) (Grant, 1996; Hackman, 1987; Malone, 1987; OpenLearn, 2009), Figure 2.1(c). The task is passed to the agents from the sub-teams with relevant domain expertise. Teams organized into sub-teams may or may not be hierarchical. This depends on the task complexity and the coordination required to manage the tasks and the information (Hackman, 1987; Malone & Herman, 2003).

Hierarchical teams are formed when leaders and sub-leaders are nominated to coordinate the tasks within the pre-defined groups. Even if the hierarchy is not pre-defined, hierarchical structures may develop as the task is decomposed into sub-tasks, and agents are chosen to coordinate those tasks, Figure 2.1(d). As represented by the broken lines in Figure 2.1(d), an agent from each sub-group may emerge as the group leader at the project runtime. At the higher level, each group-leader coordinates the activities of its group with the other agents, who are similarly chosen as group-leaders from the other sub-groups.

Flat teams distributed into social cliques

With the increased use of communication technology, project-based teams are often distributed across geographies (McDonough et al., 2001). In such teams, social cliques may develop, where the project team is divided into two to three collocated clusters, Figure 2.1(b). Even if such teams are flat for the purpose of task allocation, the opportunities for social learning are skewed due to
the physical boundaries (Leinonen et al., 2005; McDonough et al., 2001; Sutherland et al., 2007). Examples of distributed flat teams can be found in global product development teams (McDonough et al., 2001) and the current out-sourcing practice (Seshasai et al., 2006; Sutherland et al., 2007).

In summary, small, project-based work teams with specialized knowledge distribution is the focus of this research. Level of team familiarity, team structure (flat teams, flat teams with social cliques, and teams organized as task-based sub-teams), and social learning modes are taken as the critical variables that may influence the performance of such teams.

The structural variables of the teams that need to be modelled have been identified. However, team performance is also a social-cognitive issue, pertaining to the TMMs. The social structuring of teams, the structuring of their work, and how they work, is likely to affect the formation of TMM and the team performance. The goal of this research is study these effects. Therefore, a review of the literature on teamwork, and social-cognitive issues in teams, is required.

2.2.2 Teamwork and team building

Teams undergo different phases of forming, storming, norming, and performing (Tuckman, 1965). Effective teamwork requires various kinds of competencies that can be discussed in terms of the knowledge, skills and attitudes that are specific or generic to the task, and specific or generic to the team (Ancona & Caldwell, 2007; Cannon-Bowers et al., 1993; Cohen & Bailey, 1997). Team members need a well-developed mental model for the task, process, context, competence, and that of the team for effective team performance (Badke-Schaub et al., 2007; Cannon-Bowers et al., 1993; Druskat & Pescosolido, 2002; Klimoski & Mohammed, 1994; Langan-Fox et al., 2004; Lim & Klein, 2006; Mathieu et al., 2000; Mcgrath, 1991; Mohammed & Dumville, 2001; Moreland et al., 1998; Rouse et al., 1992). Badke-Schaub et al. (2007) differentiate the different types of mental modes as follows:

1. Task mental model deals with the internal representation of the related task.
2. Process mental model deals with the knowledge of the task handling.
3. Competence mental model deals with the understanding of what it means to be competent and the general confidence in the team’s capability to do the task.
4. Team mental model is the knowledge of the roles, responsibilities, capabilities and the preferences of all the agents in team.
5. Context mental model is the understanding of how and what works for the team in a given context.
Since this research primarily focuses on TMMs, a brief review of the literature on TMMs is presented.

2.2.2.1 TMM and transactive memory
Mental models are simplified internal representations of the world (Smyth et al., 1994), and, hence, mental models need not, necessarily, be accurate (Besnard et al., 2004). TMM provides a collective/shared knowledge base for the team members to draw upon. The collective/shared knowledge includes compatible knowledge (i.e., knowledge that is complementary and adds to each other), and should not be confused with knowledge overlap only (Cannon-Bowers et al., 1993; Klimoski & Mohammed, 1994; Langan-Fox et al., 2004). Badke-Schaub et al. (2007) discuss three main characteristics of TMMs: (a) sharedness (b) accuracy, and (c) importance.

Sharedness:
The term shared is used to mean both (a) knowledge held in common by the team members, and (b) knowledge divided across the team members to form complementary knowledge. The knowledge held in common does not necessarily mean that it is accurate (Rentsch & Hall, 1994). Sharedness, and in particular commonality, can be an important measure of the quality of TMM (Mohammed et al., 2000). Task interdependence may require input from the members with diverse expertise. While dealing with complex tasks or multi-disciplinary teams, it might actually be better to have knowledge divided across the team members, where each member might have specialized knowledge (Cooke et al., 2000). Distributed mental models improve the team performance in complex task environments (Sauer et al., 2006). Too much similarity in the mental models may also lead to reduced performance due to group thinking (Janis, 1972). The idea of divided and distributed knowledge in teams is also covered in the literature on transactive memory systems (Akgun et al., 2006; Griffith & Neale, 1999; Mohammed & Dumville, 2001; Wegner, 1987; Wegner, 1995).

Accuracy:
Accuracy of a TMM determines the quality of the TMM, i.e., how much of what is known is correct and precise, usually to a referent model. Accuracy is an important measure for assessing TMMs (Edwards et al., 2006). Though Besnard et al. (2004) suggest that mental models need not necessarily be accurate, accuracy influences the team performance (Edwards et al., 2006; Lim & Klein, 2006). Groups perform better when the members have an accurate model of each other’s expertise (Bunderson, 2003a, 2003b). In structured tasks, expert’s mental models have been used
as a benchmark to assess the accuracy of mental models of the other team members (Edwards et al., 2006; Lim & Klein, 2006).

**Importance:**
Mental models that capture the central attributes of a task or team have a greater influence on the team performance than ones that do not (Badke-Schaub et al., 2007). Thus, some aspects of the TMM may be more important than the others. For example, in a team with specialized experts, the TMM of each expert is developed as each of them identifies the competence of the other experts such that each expert has a well-developed mental model of the knowledge distributed across the team. However, it is likely that each expert will need to directly interact with, or allocate the tasks to, only a few of the other experts in the team. Thus, it is more important to identify the relevant experts rather than identifying the competence of the rest of the experts. Therefore, importance is a useful measure in this research because in this research, the team is modelled as a collection of experts, i.e., agents with specialized competencies.

### 2.2.2.2 Mental models and design teams
Design is often a multidisciplinary and complex task (Badke-Schaub & Frankenberger, 2004; Hacker et al., 1998). Hence, the design teams may need to have divided and specific domain knowledge, and shared (common) knowledge may not always be required (Badke-Schaub et al., 2007). Thinking in a team design activity is different to an individual design activity (Cross & Clayburn-Cross, 1995; Stempfle & Badke-Schaub, 2002). On the part of the individual designers, designing in team requires additional cognitive actions beyond those related to the design activity (Milne & Leifer, 2000). These actions corresponding to teamwork and socio-cognitive aspects are often related to information dissemination and task allocations (Akgun et al., 2006; Austin et al., 2001; Carrizosa & Sheppard, 2000; Mabogunje, 2003; Milne & Leifer, 2000). Thus, a well-developed TMM enables members of a design team to efficiently allocate tasks and responsibilities.

Larson and LaFasto (Larson & LaFasto, 1989) distinguish between tactical and creative teams. Tactical teams are well-defined, have well-define processes, and unambiguous role-clarity and accuracy. Creative teams require greater autonomy, and may require more common knowledge on teamwork processes and the context of design, across the team members (Gilson & Shalley, 2004).

Similarly, design tasks are distinguished as routine tasks and non-routine tasks (Gero, 2001). Routine tasks are well-defined, and have well-defined processes and well-defined solution space.
Routine tasks often have unique solutions such that two or more agents performing the same task will provide the same solution. Hence, in teams working on routine tasks, all that the agents need to know is who can perform what task.

On the other hand, non-routine tasks are defined as tasks that may have more than one solution (non-unique solutions). Non-routine tasks are further classified as creative and non-creative tasks. For creative tasks, the solution space may not be defined (Gero, 2001). However, only non-creative tasks are considered in this thesis. Non-routine tasks that are non-creative have a defined solution space. Such non-routine tasks can be modelled as combinatorial search problems (Campbell et al., 1999; Mitchell, 2001; Siddique & Rosen, 2001) such that the task performance requires finding one possible combination of a discrete set of sub-solutions that satisfy the specified requirements. Thus, two or more agents may provide different solutions for the same task. Hence, agents not only need to know who can perform what task, but they also need to know who is likely to provide what solution.

Thus, dealing with non-routine tasks in a team environment will require agents to develop a mental model of the capability range of the other agents.

Therefore, the role and requirements of TMM formation may vary according to the design tasks to be performed by the team. In either case, agents need to have shared process mental model and context mental model. Therefore, the process and context mental models are pre-coded into the agents for all the simulations. Task type is taken as an experiment parameter to study its correlation with TMM formation and the team performance.

2.2.2.3 Measuring TMMs:
A number of approaches based on different knowledge elicitation techniques, such as interviews, surveys, observations, and process tracing, have been proposed to measure TMMs in human teams (Langan-Fox et al., 2000; Langan-Fox et al., 2001; Lim & Klein, 2006; Mohammed et al., 2000; O’Connor et al., 2004; Webber et al., 2000). Aspects measured across different techniques include accuracy (Lim & Klein, 2006; Rentsch & Hall, 1994), sharedness (homogeneity and heterogeneity) (Cannon-Bowers et al., 1993; Langan-Fox et al., 2001; Lim & Klein, 2006; Mathieu et al., 2000; Woehr & Rentsch, 2003) and importance (Badke-Schaub et al., 2007; Mathieu et al., 2005). The literature suggests that, ideally, multiple measures should be used simultaneously to assess the TMM (Badke-Schaub et al., 2007; Langan-Fox et al., 2000; Mohammed et al., 2000; O’Connor et al., 2004; Webber et al., 2000). In the real world scenario, even applying one of these techniques is a complex process, and, hence, collecting data for multiple measures is rarely tried (Mohammed et al., 2000). Measuring the TMMs, which are
viewed as a cognitive construct, (Klimoski & Mohammed, 1994) remains a challenging
dean (Cooke et al., 2004; Klimoski & Mohammed, 1994; Langan-Fox et al., 2001;
Mohammed et al., 2000). Various techniques have been proposed for measuring the TMMs such
as Pathfinder (Langan-Fox et al., 1999; Lim & Klein, 2006), multi-dimensional scaling
(Mohammed et al., 2000), concept mapping (O’Connor et al., 2004), and so on. Mohammed et al.
(2000) argue that the measures for review of TMMs should encompass both knowledge elicitation
and knowledge representation. According to Mohammed et al. (2000), knowledge elicitation
refers to the techniques used to determine the contents of the mental model (data collection),
while knowledge representation refers to the techniques used to reveal the structure of the data, or
determine the relationships between the elements in an individual’s mind (data analysis). In real
world studies on TMMs, both the knowledge elicitation and the knowledge representation
techniques are subjective, and prone to incompleteness and inaccuracies.

2.2.2.4 Expertise and team performance

Expertise is closely related to the individual’s abilities and performance. While expertise is an
established attribute, there are no explicit measures for identification of expertise. In general, an
individual is deemed an expert based on one’s outstanding performance and reputation (Candy &
Edmonds, 2003). Expertise is directly proportional to the person’s domain knowledge and the
knowledge related to the practices and norms in the domain (Cross & Cross, 1998; Griffith et al.,
2003; Huber, 1999; Katzy, 1998; LaFrance, 1989; Leinonen et al., 2005; McDonough et al.,
2001). Expertise in any field comes with extended practice and knowledge gained from
experience (Seifert et al., 1997). Expert performers are adept at anticipating future events
(Ericsson & Charness, 1997). Literature (Cross & Cross, 1998; LaFrance, 1989) suggests that
experts possess both factual knowledge as well as tactical knowledge.

The term expertise can be extended to a group or a team, where, again, it relates to the domain
specific performance and abilities of the group as a unit (Cook & Whitmeyer, 1992). As with the
individual expertise, team expertise can also be said to be consisting of both factual and tactical
knowledge (Cooke et al., 2000; Cooke et al., 2004). In teams, the tactical knowledge not only
deals with the domain knowledge but also the intra-team tactics that facilitate coordination and
efficient usage of the expertise of the individual members. Candy and Edmonds (2003) state that:
“expertise in collaboration is a different experience…because it involves developing relationships
between the participating parties.” Hence, a mere collection of individual experts may not lead to
an expert team (Grecu & Brown, 1998; Huber, 1999). Individual experts need to interact and
communicate with each other to develop team expertise. Team expertise develops as agents learn
to efficiently utilize each other’s expertise, and allocate tasks to the agents that have the expertise in performing the given task.

Team expertise is measured through team performance, where the team performance is the performance and abilities of the team as a unit (Cook & Whitmeyer, 1992). Therefore, a review of the literature on measuring team performance is presented.

**Measuring team performance and effectiveness:**
Team performance and effectiveness can be measured across various dimensions, such as quality of the task output, growth in the team knowledge, increase in team cohesion, reduction in the internal conflicts, and so on (Cohen & Bailey, 1997). The dimensions considered should reflect the holistic team behaviour and not simply be an aggregate of the behaviours of the individual team members (Cooke et al., 2000; Cooke et al., 2004).

Analogous to the relationship of knowledge and performance in individual expertise (Chase & Simon, 1973; Glaser & Chi, 1988; Seifert et al., 1997), team performance is directly related to the team knowledge (Cooke et al., 2000). Most team knowledge measurement approaches tend to assess collective knowledge through knowledge elicitation, team metric, and aggregation methods. The holistic approach considers the team knowledge resulting from the application of team process behaviours (i.e., communication, situation assessment, coordination) to the collective knowledge (Cooke et al., 2000).

Team communication, being holistic team behaviour, can be used as a measure of team performance (Cooke et al., 2000; Cooke et al., 2004). Teams that require less communication between individuals to achieve same result are considered to perform better (Blinn, 1996; Langan-Fox et al., 2000; Langan-Fox et al., 2001; Margerison & McCann, 1984). High performance outcomes and seamless interaction are said to correspond with expert TMM (Edmondson, 1999).

In summary, the team expertise is distributed across the team. Team expertise is said to develop as the agents in the team develop mental models for task, process, context and the team. In expert teams, the task, process and context mental models are expected to be shared across the team members. Since this research focuses on TMM, the task, process and context mental models are pre-coded into the agents. Thus, each agent in the team has same task, process and context mental model. Therefore, in these simulations, as the TMM is formed, it leads to the formation of team expertise.

Hence, this research explores a specific aspect of team expertise. The focus of this thesis is to conduct a comparative study of the contributions of the different social learning modes on TMM.
formation and the team performance, given that other factors such as learning capabilities, efficiency, and expertise of the individual agents, remain the same for all the simulations.

2.3 Research method

A computational approach based on modelling the team members as agents is adopted. The team is represented as a multi-agent system (MAS) where the agents interact to perform the task. The MAS creates a simulation environment where the intended research parameters can be modelled and implemented. Experiments can be conducted using this simulation environment to simulate the different scenarios proposed for the research. This kind of computational approach is widely used across the different research domains for modelling and understanding societies (Conte & Gilbert, 1995; Goldstone & Janssen, 2005; Macy & Willer, 2002; Wooldridge, 2002). A review of the literature on social simulations is presented.

Computational sociology and CMOT:

With regards to the computational organization theory, Carley (Carley, 1994; Carley, 1999) suggests that the computational models are a suitable means for generating hypotheses of organizational models. These can be used as a guide to design the human lab experiments and suggest what data to collect in the field study. Carley (Carley, 1999) claims that these computational models are particularly interesting because they are themselves the theory that is being developed and tested. Unlike traditional organizational theories, which were primarily static, a computational approach allows development of an evolutionary and longitudinal theory of organizations (Lant, 1994). Such computational models should facilitate equivalency test and comparison with other models. This involves the process of ‘docking’ (Axtell et al., 1996) whereby the theory can be externally validated using some other comparable model.

Discussing artificial organizations, in the light of human organizations, Carley (Carley, 1996) suggests that:

1. Formal models facilitate organization theory by providing a means to explore the complex, adaptive, and non-linear nature of human organizations.

2. As the complexity of the organizational structure increases, the ability to predict organizational behaviour, with simpler computational agents, increases.

3. Different types of agent models compare differently to human subjects, under different settings. This emphasizes the usefulness of docking the models using different agent models.
Hence, there should also be an independent method to assess the reliability and effectiveness of a computational model used for studying social behaviour (Axelrod, 1997; Axtell et al., 1996; Carley, 1997; Levitt et al., 2005). Carley and Newell (Carley & Newell, 1994) propose a Social Turing test to assess a developed computational model. The test includes the following three steps:

1. Based on the hypothesis, construct a collection of social agents and put them in the social situation, as defined by the hypothesis. Recognizable social behaviour should emerge from the model.

2. Many aspects of the computational model that are not specified by the model can be determined at will. In general, such aspects should be set based on known human data or handled using Monte Carlo techniques.

3. The behaviour of the computational model can vary widely with such specification, but it should remain recognizably social. If so, then the Social-Turing test is met.

Over the years, various models of artificial societies, teams, and organizations have been developed. These models have contributed significantly to theory building, testing hypotheses, generating hypotheses, and other advancements in these areas. Some of the prominent ones, relating to teams and organizations, are VDT (Virtual Design Teams) (Kunz, 1998; Jin, 1995), ORGAHEAD (Carley & Svoboda, 1996), and TAC Air Soar (Tambe, 1996). These models are significantly different in their objectives. The work on VDT is focused on identifying the influence of organizational structure and information processing tools on team performance, assessed mainly from the perspective of project management and scheduling. The VDT involves modelling the processing time, work flow, and tool usage. ORGAHEAD is focused at developing the theories relating to organizational design and organizational learning. ORGAHEAD’s design is modular, involving building blocks for task assignment, organizational structure (hierarchy, flat etc), communication tools, etc. Unlike VDT, agents in ORGAHEAD can learn new skills. TAC Air Soar is focused at developing tools for facilitating real-time task coordination, in actual physical environment, involving both human and artificial agents. Though the focus of the models is different, each of these models emphasizes the importance of coordination and communication for effective teamwork.

ORGAHEAD has been used for studies similar to this research. In separate studies, ORGAHEAD has been used to study the: (1) influence of personnel turnover on the team performance (Carley, 1992), (2) influence of group training and individual training on the team performance and the TMMs (Ren et al., 2001; Ren et al., 2006), and (3) influence of transactive memory including aids such as books and external databases on the team performance (Schreiber
& Carley, 2004; Schreiber & Carley, 2003). However, these studies have not explored the relative influence of the different learning modes on TMM formation or the team performance.

Agents in ORGAHEAD are based on a fairly detailed cognitive agent architecture, called SOAR (Laird et al., 1987; Newell, 2002), and learn both about the task as well as the team. Additional assumptions are made in these studies to reproduce characteristics similar to attributes such as the short term and long term memory, and information loss in the organizations. Since the research reported in this thesis assumes the task knowledge to be fixed, and only the TMM needs to be learnt, there would be no impasse for task performance. Hence, the agent architecture required need not be as detailed as SOAR. The agent architecture to be used in this thesis should facilitate direct measurement of the formation of TMM. The model should also provide the ability to control the learning modes. Unlike the studies reported by Carley and others that have used ORGAHEAD, the model used in this thesis does not consider additional factors such as short term/long term memory, recency, or information loss. This allows greater control on the independent variables in the study, and the results reported are not influenced by these additional parameters.

The experiments reported in this thesis can be reproduced using ORGAHEAD for docking (Axtell et al., 1996). In such a scenario, some variations in results may be expected due to superposition of the studied parameters with the other parameters. The developed computational model is validated through docking by comparing the results from the preliminary simulations against similar simulations conducted earlier using ORGAHEAD (section 6.1).

As observed in the literature review, a range of agent architectures and learning approaches have been adopted in modelling teams. The agent design determines what kinds of experiments can be conducted and whether the chosen agent architecture is suitable for the desired study. Thus, a review of the literature on agent architectures and learning was conducted to identify the requirements for the agents suited to investigate the research questions.

2.4 Requirements for agent architecture and learning:

Various definitions of agents exist (Russell & Norvig, 2002; Shoham, 1993; Wooldridge & Jennings, 1995), and various types of agents have been defined. At the very least, an agent is autonomous and observes and acts in an environment. All the agent types can improve their performance through learning (Russell & Norvig, 2002; Wooldridge & Jennings, 1995). The type and capability of an agent depends on its architecture (Conte & Castelfranchi, 1995; Russell & Norvig, 2002; vandenBroek, 2001; Verhagen, 2000; Wooldridge, 2002). Bellifemine et al. (2007)
identify popular agent architecture styles as logic-based architectures, BDI architectures and layered (hybrid) architectures.

In logic-based architectures, the environment is represented symbolically and manipulated using reasoning mechanisms. BDI architectures (Rao & Georgeff, 1995) define mental attitudes of belief, desire and intentions. Beliefs are the agent’s knowledge about the environment, which may be incomplete or inaccurate. Desires are the agent’s objectives or goals, and intentions are the desires that the agent has committed to achieve. Plans are part of the belief that a particular action will lead to the desired goal. In general, BDI agents have hierarchically organized plans (mostly pre-coded) to choose from, and act upon.

Layered architectures allow both reactive and deliberative agent behaviours. Subsumption architecture (Brooks, 1991) is the best known reactive architecture, which is organized hierarchically as layers of finite state machines.

Intelligent agents are often discussed in terms of their cognitive architecture. Cognitive architecture consists of representational assumptions, characteristics of agent’s memories, and the processes that operate on the memories (Langley et al., 2009). Cognitive architectures can be symbolic, connectionist or hybrid. Some of the more popular cognitive architectures such as SOAR (Laird et al., 1987; Newell, 2002) and ACT-R$^5$ (Anderson & Lebiere, 1998) are based on a production system, which defines set of generic rules.

Other agent architectures have been proposed with different learning approaches. Numerous learning algorithms have been developed based on evolutionary learning, inductive learning, probabilistic learning, reinforcement learning, statistical learning, and so on (Mitchell, 1997; Russell & Norvig, 2002). The choice of learning approach should be based on what knowledge the agent has to access, recognize, process, and maintain, for later use and effective interaction with its environment.

Wooldridge and Jennings (Wooldridge & Jennings, 1995) distinguish between weak and strong agents. Weak agents, characterized by autonomy, social ability (Genesereth & Ketchpel, 1994), reactivity, and pro-activeness, are sufficient for most multi agent systems (Wooldridge & Jennings, 1995). Strong agents have additional properties, characterized by the mental attitudes, knowledge, beliefs, and so on (Shoham, 1993). Various characteristics of a social and cognitive actor for different environments are discussed in the literature (Carley & Newell, 1994; Helmhout, 2006).

---

$^5$ Learning in ACT-R occurs both at structural as well as statistical levels. Activation of declarative chunks can increase or decay based on a probability function relating to the observed behaviour.
**Social agent**

Carley and Newell (Carley & Newell, 1994) describe a social agent along two dimensions: (1) processing capabilities, and (2) differentiated knowledge of self, task, domain, and the environment. For studies in social sciences, the model social agents tend to have lower information-processing capabilities but higher knowledge (Carley & Newell, 1994; Wooldridge, 2002). The choice of an agent’s information processing capabilities and knowledge levels should be based on the complexity of the environment and the focus of the study. Carley and Newell (Carley & Newell, 1994) propose a mapping matrix (Figure 2.2) to facilitate this decision making.

![Figure 2.2: Indicative mapping for required agent details to environmental complexity](image)

**Cognitive agent:**

According to Langley et al. (Langley et al., 2009), the capabilities of a cognitive architecture include: recognition and categorization, decision making and choice, perception and interpretation, prediction and monitoring, problem solving and planning, reasoning and belief.

---

6 Adopted from Carley and Newell 1994
maintenance, execution and action, interaction and communication, and remembering, reflection and learning. Carley and Newell (Carley & Newell, 1994) identify similar requirements for a cognitive agent. As shown in Carley and Newel’s mapping matrix, a cognitive agent is close to being the most detailed social agent. Such agents may show more realistic and complex behaviours in terms of their proximity to reproducing human behaviour.

However, as suggested in the mapping matrix (Figure 2.2), a detailed cognitive model may not be necessary for the research topics (models of others, organizational goals, social cognition, and group making) being investigated in this thesis.

2.5 Summary

This research uses a computational approach to investigate the role of social learning in TMM formation and team performance for small, project-based, design teams, with specialized knowledge distribution. Social learning contributes to the formation of TMM and team performance. However, the role different modes of social learning on the formation of TMM and team performance are not well understood. In real world studies, learning modes may be difficult to distinguish and control because the experimenters may have to rely on their qualitative observations, as well as the feedbacks from the subjects. In a computational model, the different modes of social learning can be distinctly represented, grounded in the folk theory of mind. Team performance and TMM formation may also vary due to the structural and socio-cognitive aspects of the team. Team structure is identified as a critical structural variable. Since this research focuses on project-based design teams, team familiarity and task types are identified as the other important variables, which can be computationally modelled. Based on the review of the literature on the folk theory of mind, busyness levels are taken as another important variable that determine what events an agent can observe, and learn from. Busyness levels are particularly suitable for this computational study because, while they are expected to influence the level of TMM formation and the team performance, they are difficult to measure and control in real world studies. Thus, based on the literature review, learning modes, team structure, level of team familiarity, busyness level and the task types are identified as the five independent variables of interest in this research. TMM and team performance are the two dependent variables in this research. TMM is measured in terms of the amount of TMM formation, and team performance is measured in terms of the amount of team communication.

Therefore, this thesis is based on the premise that the contributions of the different social learning modes to the formation of TMMs and the team performance in project-based teams may
vary with the team structure, busyness level of the agents, the level of team familiarity, or the task type. Hence, the research aims to investigate the correlations between the social learning modes, the team structures, busyness levels, levels of team familiarity, and the task types, in terms of the level of TMM formation and the team performance.
Chapter 3

Research Approach and Hypotheses

This chapter presents the research framework and the hypotheses being investigated. Based on the literature review (Chapter 2), the research framework outlines the dependent and independent variables. The hypotheses section explains the likely correlations for the independent and dependent variables outlined in the research framework.

3.1 Research framework

The research framework takes the following definition for TMM (Badke-Schaub et al., 2007; Cannon-Bowers et al., 1993; Klimoski & Mohammed, 1994; Langan-Fox et al., 2004) and team expertise (Candy & Edmonds, 2003; Cooke et al., 2000; Cooke et al., 2004):

A TMM is the internal representation of the competence, and the capability range, of all the team members in the different tasks that the team needs to perform. Each agent develops a TMM by itself, based on its interactions (observations) with (of) the tasks and the other agents. A TMM is formed as team members create and maintain Agent-Mental-Models (AMM) of individual agents, including self. The AMM is learned through social interactions and observations.

Team expertise is assessed based on the collective performance of the team. Team expertise comprises of the knowledge about the task, process, team and context (section 2.2.2.4). This means, even if a team is formed with a collection of experts that have well-developed knowledge about the task, process and the context, the team may not collectively perform as an expert team because the team members do not have the knowledge about the team (who knows what, and what their capability range is in the tasks they can perform). This research assumes such a scenario. All the team members are considered to be individual experts such that the knowledge related to
task, process and context mental models is pre-coded into the agents. Hence, team expertise develops as the team members develop a TMM.

Figure 3.1 is a schematic representation of the research framework. At the core of the framework is social interaction. Interaction among the team members allows them to learn about each other and form individual AMMs. As AMMs for each agent are developed, the TMM is formed. Since it is assumed that the mental model for task, process and context is well developed (pre-defined) for each agent, the formation of TMM is expected to enhance team expertise.

Since social learning affects TMM formation, factors affecting social interaction, both at the agent and the team level, are likely to influence the amount of TMM formation, which in turn should affect the team expertise. At the agent level, two factors are considered: (1) Modes of learning available to the agents, and (2) Busyness levels of the agents. At the team level, the two factors considered are: (1) Team structure, and (2) The level of team familiarity. The two agent factors, and the two team factors, along with the task types, form the five independent variables.

Since expertise of team is reflected in high team performance and seamless interaction (Candy & Edmonds, 2003; Cross & Cross, 1998; Edmondson, 1999; Huber, 1999; Powell et al., 1993), team performance (rate of task completion) is taken as the measure of team expertise. Thus, the amount of TMM formation and rate of task completion (measured as the number of messages exchanged) are the dependent variables. It is expected that the teams that have higher level of formation of team expertise (amounts of TMM formation) should have a higher rate of task completion (lower number of messages exchanged), even though this is not explicitly modelled into the system.
Busyness levels are expected to influence the level of TMM formation (Cramton, 2001; Driskell et al., 1999; Gilbert & Osborne, 1989; Griffiths et al., 2004) and the team performance. If a new project (test project) is started, where some or all the agents are retained from the previous project (training project), the agents may have a pre-developed TMM at the start of the new project. The pre-developed TMM is likely to enhance the team performance in the new project. The amount of increase in the team performance with the pre-developed TMM should vary according to the level of team familiarity, and the busyness level of the agents in the training project.

3.2 Hypotheses being investigated

As discussed in the research framework, the different cases, parameters and variables considered in these empirical studies include:

1. Social learning modes (LM) (parameter)
2. Busyness levels (BL) (variable)
3. Levels of team familiarity (TF) (variable)
4. Team structure (TS) (case)
5. Task types (T) (case)

Team performance and levels of TMM formation are the endogenous variables of interest.

3.2.1 Correlation between social learning modes and busyness levels

Agents are capable of learning from social interactions as well social observations. Three cases of learning are distinguished:

1. Learning from personal interactions only (PI)
2. Learning from partial modes
   a. learning from personal interaction and task observations (PI+TO), or
   b. learning from personal interaction and interaction observations (PI+IO)
3. Learning from all modes (learning from personal interactions, task observations as well interaction observations) (PI+IO+TO)

Social learning is directly related to team performance (Ancona & Caldwell, 2007; Moreland et al., 1998; Ren et al., 2001). Hence, the increase in team performance should be lowest for teams where agents learn only from personal interactions, and lower for teams where in agents learn from partial modes when compared to teams where agents have all modes of learning available to them.
However, higher busyness levels should correlate with lower levels of social learning because busyness inhibits attention towards an observable interaction or task performance (Cramton, 2001; Driskell et al., 1999; Gilbert et al., 1988; Griffiths et al., ; Kirsh, 2000). Therefore, higher busyness levels should have a negative effect on team performance. Busyness levels should have greater negative effects on team performance in teams where social learning is likely to have greater contribution. Hence, for teams that have all modes of social learning available to agents, the decrease in team performance with increase in busyness levels should be higher, i.e., it is hypothesized that:

*When compared to the teams that have all modes of learning available to the agents, the decrease in team performance, with the increase in busyness levels, is lower in the teams that have partial modes of learning available to the agents. The decrease in team performance, with the increase in busyness levels, is lowest for the teams in which the agents learn only from personal interactions.*

**Hypothesis 1**

Figure 3.2 shows a graph to illustrate this hypothesis. The reduction in team performance with the increase in busyness levels is expected to be highest when the agents have all modes of learning available to them. The slope for “all learning modes” is the steepest and the slope for “only personal interaction” is zero. The slope for “only personal interaction” is zero because if the agents cannot learn from social observations at all, busyness levels should not have any influence on their social learning, and, hence, the team performance.

![Figure 3.2: Hypothesized influence of busyness on team performance across different learning modes](image)

Similarly, it can be argued that in the teams that have all modes of social learning available to the agents, the level of TMM formation will be higher. Hence, the decrease in TMM formation with the increase in busyness should also be higher for the teams in which all modes of social learning available to the agents, i.e., it is hypothesized that:

*When compared to the teams that have all modes of learning available to the agents, the decrease in levels of TMM formation, with the increase in busyness levels, is lower in the teams that have partial modes of learning available to the agents. The decrease*
in levels of TMM formation, with the increase in busyness levels, is lowest for the teams in which the agents learn only from personal interactions.

**Hypothesis 2**

Figure 3.3 shows a graph to illustrate this hypothesis. The reduction in TMM formation with the increase in busyness is highest when the agents have all modes of learning available to them. The slope for “all learning modes” is the steepest and the slope for “only personal interaction” is zero. The slope for “only personal interaction” is zero because if the agents cannot learn from social observations at all, busyness levels should not have any influence on TMM formation.

![Hypothesized influence of busyness on TMM formation across different learning modes](image)

**Figure 3.3: Hypothesized influence of busyness on TMM formation across different learning modes**

### 3.2.2 Correlation between social learning modes and team familiarity

Team familiarity is believed to enhance the team performance (Espinosa et al., 2002; Harrison et al., 2003; Hinds et al., 2000; Huckman et al., 2008). However, team familiarity is useful only when the team members have had the opportunity for social interactions and observations that could allow them to learn about each other. Therefore, the teams in which members have greater social learning opportunities should have a higher rate of increase in the team performance with the increase in team familiarity. Hence, it can be hypothesized that:

> When compared to the teams that have all modes of learning available to the agents, the increase in team performance, with the increase in levels of team familiarity, is lower in the teams that have partial modes of learning available to the agents. The increase in team performance, with the increase in levels of team familiarity, is lowest for the teams in which the agents learn only from personal interactions.

**Hypothesis 3**

Figure 3.4 shows a graph to illustrate this hypothesis. The increase in the team performance with the increase in team familiarity is highest when the agents have all modes of social learning available to them. The slope for “all learning modes” is the steepest, and “only personal interaction” has the least slope.
Further, since increase in the busyness level reduces the social learning opportunities, the increase in team performance with the increase in team familiarity should be lower for the teams when members have higher busyness levels. Even if the team members may have worked together in a previous project, the higher busyness level may not allow the team members to develop a TMM that could facilitate task allocation, coordination or audience design. Based on these arguments, it can be hypothesized that:

*The increase in team performance, with the increase in team familiarity, is higher when busyness levels are lower.*

**Hypothesis 4**

Figure 3.5 shows a graph to illustrate this hypothesis. When busyness levels are the lowest, i.e., X (where X, Y and Z are positive numbers such that X < Y < Z), the slope is steepest.

**3.2.3 Correlation between social learning modes and team structure:**

Besides busyness, social learning depends on the opportunities available to the agent for social interactions and observations. Since only formal interactions are considered in this thesis, agents are likely to have greater social learning if the team is flat because flat teams provide the opportunity to interact with and observe more agents than the teams organized into sub-teams. However, in flat teams, the TMM may require more time to be developed because the agents need to allocate and coordinate tasks efficiently among more agents. Therefore, it is conjectured that the positive effects of social learning on the team performance may not be as significant in flat teams as in the teams organized into task-based sub-teams. In task-based sub-teams, the co-
ordination is required in smaller groups that can learn about each other more quickly and more comprehensively.

With more modes of social learning, flat teams in which all the members can observe each other’s activities should show higher improvements in team performance compared to the flat teams with social cliques. While the task coordination requirements, in terms of the number of agents, remain the same for either case, flat teams distributed into social cliques reduce the social learning opportunities across the social groups. Based on these conjectures, it can be hypothesized that:

*The increase in team performance, with the increase in the number of modes of social learning, is highest when the team is organized into task-based sub-teams, lower when the team is flat, and lowest when the team is flat but grouped into social cliques.*

**Hypothesis 5**

Figure 3.6 shows a graph to illustrate this hypothesis. When the teams are organized into task-based sub-teams, the difference between the performance of teams with “all learning modes” and the teams with “only personal interaction” is much higher than that for flat teams or the flat teams with social cliques with a similar difference.

If the overall team sizes are comparable, then social learning should lead to higher levels of TMM formation in flat teams as compared to the teams organized as sub-teams. Compared to the teams organized as sub-teams, in the flat teams, agents may interact with more agents. Thus, there is less likelihood of repeated observations of the same agent. In terms of the level of TMM formation, flat teams with social cliques have an advantage over the teams organized into task-based sub-teams because in flat teams with social cliques, the agents interact with and observe more agents than the teams with task-based sub-teams. Thus, it can be hypothesized that:

*The increase in levels of TMM formation, with the increase in the number of modes of social learning, is highest when the team is flat, lower when the team is flat but*
grouped into social cliques, and lowest when the team is organized into task-based sub-teams.

**Hypothesis 6**

Figure 3.7 shows a graph to illustrate this hypothesis. When the teams are organized into task-based sub-teams, the difference between the levels of TMM formation for teams with “all learning modes” and the teams with “only personal interaction” is much higher than that for the flat teams or the flat teams with social cliques with a similar difference.

**Figure 3.7: Hypothesized correlation of team structure and modes of learning in terms of TMM formation**

TMM formation can be measured in terms of the amount (density) of TMM, accuracy of the TMM and the importance of what is learnt (Badke-Schaub et al., 2007; Klimoski & Mohammed, 1994; Mohammed et al., 2000). Hypothesis 6 specifically considers the amount of TMM formation. This need not necessarily mean that the other measures of TMM are better in flat teams as compared to the other teams. Accuracy is not measured because, in these simulations, all that the agents learn is accurate. In fact, in terms of the importance of TMM, i.e., learning the information that is relevant and important, it is conjectured that the teams organized into sub-teams may perform better than the flat teams because all the task related information is generally required within the task groups. Flat teams with social cliques should be worse than the flat teams because they do not allow learning from observations across the social groups, while the agents with the relevant task competence might be part of some other social group. Therefore, a notion of efficiency of the TMM is introduced.

The efficiency of TMM is the ratio of the amount of relevant information in the TMM to the total amount of TMM formed. In terms of the overall TMM, this is indirectly measured as the ratio of the team performance to the total amount of TMM formed in a given simulation, i.e.,

\[
\text{Efficiency of TMM} = \frac{\text{Team performance}}{\text{Level of TMM formed}}
\]

Based on these conjectures, it can be hypothesized that:

*When all modes of social learning are available to the agents, the increase in the efficiency of TMM formation is highest when the team is organized into task-based*
sub-teams, lower when the team is flat, and lowest when the team is flat but grouped into social cliques.

Hypothesis 7

As discussed earlier, busyness reduces the amount of social learning. Thus, if the increase in the team performance with social learning is highest in the teams organized as task-based sub-teams then the decrease in the team performance with the increase in busyness level should also be highest in the task-based sub-teams. Based on the same argument, the decrease in the team performance with the increase in the busyness level should be higher for the flat teams when compared to the flat teams with social cliques. Thus, it can be hypothesized that:

The decrease in team performance, with the increase in busyness levels, is highest when the team is organized as task-based sub-teams, lower when the team is flat, and lowest when the team is flat but grouped into social cliques.

Hypothesis 8

Figure 3.8 shows a graph to illustrate this hypothesis. When the teams are organized into task-based sub-teams, the difference between the team performance at higher and lower busyness levels is greatest, i.e., the slope for “busyness levels vs. team performance” is steepest when the teams are organized as sub teams. The slope for the “busyness levels vs. team performance” is least when the teams are flat but divided into social cliques.

![Figure 3.8: Hypothesized correlation of team structure and busyness in terms of team performance](image)

Similarly, it can be argued that:

The decrease in the amount of TMM formation, with the increase in busyness levels, is highest when the team is flat, lower when the team is flat but grouped into social cliques, and lowest when the team is organized into task-based sub-teams.

Hypothesis 9

Figure 3.9 shows a graph to illustrate this hypothesis. When the teams are flat, the difference between the TMM formation at higher and lower busyness levels is greatest, i.e., the slope for “busyness levels vs. levels of TMM formation” is steepest when the teams are flat. The slope for
the “busyness levels vs. levels of TMM formation” is least when the teams are organized as task-based sub-teams.

![Figure 3.9: Hypothesized correlation of team structure and busyness in terms of TMM formation](image)

It is conjectured that in smaller groups, with fewer agents to interact with and observe, the likelihood that team familiarity would lead to more agents developing mental models of each other is greater. Thus, in terms of the team performance, which is directly influenced by the efficiency of the TMM formed, it can be expected that the teams organized as task-based sub-groups are better for formation of relevant task related TMMs. Hence, it can be hypothesized that:

\[
\text{The increase in team performance, with the increase in team familiarity, is highest when the team is organized into task-based sub-teams, lower when the team is flat, and lowest when the team is flat but grouped into social cliques.}
\]

**Hypothesis 10**

Figure 3.10 shows a graph to illustrate the hypothesis. When the teams are organized into task-based sub-teams, the difference between the team performance at higher and lower levels of team familiarity is greatest, i.e., the slope for “team familiarity vs. team performance” is steepest when the teams are organized into task-based sub-groups. The slope for the “team familiarity vs. team performance” is least when the teams are flat but grouped into social cliques.

![Figure 3.10: Hypothesized correlation between team familiarity and team structure in terms of team performance](image)

### 3.2.4 Correlation between social learning and task types:

It is likely that the amount of increase in the team performance with social learning may vary with the task type (section 2.2.2.2). Social learning may have a greater role to play when the teams are working on non-routine tasks because (1) the agents require more detailed TMM while working on non-routine tasks, and (2) the non-routine tasks also require integration and coordination.
Social learning should not only allow the agents to identify the experts-to-tasks mapping, but social learning should also allow the development of mental models for the capability range for each of the experts in the tasks they can perform. However, since there is more to learn about other agents in case of the non-routine tasks, the positive effects of social learning on team performance will only be higher if the team members have spent more time with each other.

On the other hand, the teams working on routine tasks should be able to improve their performance with social learning much faster. The details of the task performed are available from the task observation, but these details are not known from the interaction observations. While details of the task performed are important when the task is non-routine, for routine tasks, such details are not required. Hence, the decrease in the team performance resulting from the lower number of learning modes (partial or only personal interactions) should be greater for the teams working on routine tasks than that for the teams working on non-routine tasks. Thus, it is hypothesized that:

*The decrease in team performance, with the reduction in the number of learning modes, is greater for the teams are working on routine tasks as compared to the teams working on non-routine tasks.*

**Hypothesis 11**

Figure 3.11 shows a graph to illustrate the hypothesis. The difference in the performance of the teams with “all learning modes” and the teams with “only personal interaction” is higher when the teams are working on routine tasks, as compared to the teams working on non-routine tasks.

The teams working on non-routine tasks are likely to take longer to finish a project (i.e., more number of messages are exchanged between the agents). Hence, by the end of the project the teams working on non-routine tasks should have higher amounts of TMM formed as compared to the teams working on routine tasks.

**Figure 3.11: Hypothesized correlation of task types and learning modes in terms of team performance**

The task related interactions between any two agents can generally be expected to be longer if the team is working on non-routine tasks. Hence, even at higher busyness levels, an agent has a
fair chance to observe the interaction between two other agents to learn their competences. However, for the teams working on routine tasks, such interactions are usually very brief (i.e., fewer messages are exchanged). Hence, if an agent working on routine tasks misses the opportunity to observe the interaction because of higher busyness level then no expert-task mapping is formed for that observable interaction. Based on these conjectures, it is hypothesized that:

\[ \text{The decrease in team performance, with the increase in busyness levels, is greater for the teams working on routine tasks as compared to the teams working on non-routine tasks.} \]

**Hypothesis 12**

Figure 3.12 shows a graph to illustrate the hypothesis. When the teams are working on routine tasks, the difference between the team performance at higher and lower busyness levels is greater, i.e., the slope for “busyness levels vs. team performance” is steeper, as compared to the “busyness level vs. team performance” slope for the non-routine tasks.

![Figure 3.12: Hypothesized correlation of busyness levels and team performance for different task types](image)

Similarly, there are fewer numbers of interactions and fewer details about a TMM to learn in the case of routine tasks. Therefore, in relative terms, the decrease in TMM formation with the increase in busyness levels can be expected to be higher for the teams working on routine tasks as compared to the teams working on non-routine tasks. Members of the teams working on non-routine tasks should have more opportunity to observe the other team members because the duration of the interactions between any two members can be expected to be longer. For teams working on non-routine tasks, the project duration should also be longer. Thus, it is hypothesized that:

\[ \text{The decrease in levels of TMM formation, with the increase in busyness levels, is greater for the teams working on routine tasks as compared to the teams working on non-routine tasks.} \]

**Hypothesis 13**
Figure 3.13 shows a graph to illustrate the hypothesis. When the teams are working on the routine tasks, the difference between the TMM formation at higher and lower busyness levels is greater, i.e., the slope for “busyness levels vs. TMM formation” is steeper, as compared to the “busyness level vs. TMM formation” slope for the teams working on the non-routine tasks.

![Figure 3.13: Hypothesized correlation of busyness levels and TMM formation for different task types](image)

The team performance should increase with the team familiarity irrespective of whether the team is working on routine or non-routine tasks. However, TMM should have greater role to play for teams working on non-routine tasks. If more agents have pre-developed mental models of each other by having worked together in a similar project, the teams working on non-routine tasks should have significantly higher improvement in their performance because (1) the teams working on non-routine tasks take longer to perform the task, and (2) in such team’s agents need additional details about the other agents in terms of their capabilities for task solutions. Hence, it is hypothesized that:

*The rate of increase in team performance, with the increase in team familiarity, is higher for the teams working on non-routine tasks than that for the teams working on routine tasks.*

**Hypothesis 14**

Figure 3.14 shows a graph to illustrate the hypothesis. When the teams are working on the non-routine tasks, the difference between the team performance at higher and lower levels of team familiarity is greater, i.e., the slope for “team familiarity vs. team performance” is steeper, as compared to the “team familiarity vs. team performance” slope for the teams working on the routine tasks.

![Figure 3.14: Hypothesized correlation of team familiarity and team performance for different task types](image)

---

7 Agents have more to learn when the task is non-routine. This includes learning about the capability range of the other agents in addition to the expert-task mapping.
Figure 3.14: Hypothesized correlation of team familiarity and team performance for different task types

In flat teams, any of the team members may have a competence in any task, unlike the teams organized into task-based sub-teams, which, by structure, narrows down the search space for the relevant experts to the members within a task-group. Thus, team performance is likely to vary with the team structure for either task type. This difference in performance can be expected to vary with the task types as well. Since there is more to learn when the teams are working on the non-routine tasks, it is hypothesized that:

*The relative difference in team performance across the different team structures is higher for the teams working on non-routine tasks as compared to the teams working on routine tasks.*

**Hypothesis 15**

Figure 3.15 shows a graph to illustrate the hypothesis. The difference in the performance between the teams organized as sub-teams and flat teams is higher when the teams are working on the non-routine tasks, as compared to the teams are working on the routine tasks.

![Figure 3.15: Hypothesized correlation of team structure and team performance for different task types](image)

**Figure 3.15: Hypothesized correlation of team structure and team performance for different task types**

On the other hand, any reduction in social learning opportunities can be expected to have a higher relative effect on the TMM formation in the teams working on routine tasks as compared to the teams working on the non-routine tasks because, for routine tasks (1) there are fewer opportunities for social learning due to fewer and shorter interactions, and (2) there are fewer details to be learnt about the TMM. Based on this argument, it is hypothesized that:

*The relative difference in levels of TMM formation across the different team structures is higher for the teams working on routine tasks as compared to the teams working on non-routine tasks.*

**Hypothesis 16**

Figure 3.16 shows a graph to illustrate the hypothesis. The difference in the levels of TMM formation between the teams organized as sub-teams and the flat teams is higher when the teams are working on non-routine tasks as compared to the teams working on routine tasks.
Thus, this research investigates the sixteen research hypotheses that state the different correlations between the five independent variables (i.e., learning modes, busyness level, level of team familiarity, task type, and team structures) and the two dependent variables (i.e., TMM formation and team performance). Table 3.1 summarizes the hypotheses being investigated in this research.
Table 3.1: Matrix of hypotheses being investigated

<table>
<thead>
<tr>
<th>Learning Modes (LM)</th>
<th>Busyness levels (BL)</th>
<th>Team Familiarity (TF)</th>
<th>Team Structure (TS)</th>
<th>Task Type (T)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>H1:</strong> When compared to the teams that have all modes of learning available to the agents, the decrease in team performance, with the increase in busyness levels, is lower in the teams that have partial modes of learning available to the agents. The decrease in team performance, with the increase in busyness levels, is lowest for the teams in which the agents learn only from personal interactions.</td>
<td><strong>H3:</strong> When compared to the teams that have all modes of learning available to the agents, the increase in team performance, with the increase in levels of team familiarity, is lower in the teams that have partial modes of learning available to the agents. The increase in team performance, with the increase in levels of team familiarity, is lowest for the teams in which the agents learn only from personal interactions.</td>
<td><strong>H5:</strong> The increase in team performance, with the increase in the number of modes of social learning, is highest when the team is organized into task-based sub-teams, lower when the team is flat and lowest when the team is flat but grouped into social cliques.</td>
<td><strong>H6:</strong> The increase in levels of TMM formation, with the increase in the number of modes of social learning, is highest when the team is flat, lower when the team is flat but grouped into social cliques, and lowest when the team is organized into task-based sub-teams.</td>
<td><strong>H7:</strong> When all modes of social learning are available to the agents, the increase in the efficiency of</td>
</tr>
</tbody>
</table>
lower in the teams that have partial modes of learning available to the agents. The decrease in levels of TMM formation, with the increase in busyness levels, is lowest for the teams in which the agents learn only from personal interactions.

<table>
<thead>
<tr>
<th>BL</th>
<th><strong>H4:</strong> The increase in team performance, with the increase in team familiarity, is higher when busyness levels are lower.</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>TMM formation is highest when the team is organized into task-based sub-teams, lower when the team is flat, and lowest when the team is flat but grouped into social cliques.</td>
</tr>
<tr>
<td></td>
<td><strong>H8:</strong> The decrease in team performance, with the increase in busyness levels, is highest when the team is organized as task-based sub-teams, lower when the team is flat, and lowest when the team is flat but grouped into social cliques.</td>
</tr>
<tr>
<td></td>
<td><strong>H9:</strong> The decrease in the amount of TMM formation, with the increase in busyness levels, is highest when the team is flat, lower when the team is flat but grouped into social cliques, and lowest when the team</td>
</tr>
<tr>
<td></td>
<td><strong>H12:</strong> The decrease in team performance, with the increase in busyness levels, is greater for the teams working on routine tasks as compared to the teams working on non-routine tasks.</td>
</tr>
<tr>
<td></td>
<td><strong>H13:</strong> The decrease in levels of TMM formation, with the increase in busyness levels, is greater for the teams working on routine tasks as compared to the teams working on non-routine tasks.</td>
</tr>
</tbody>
</table>
is organized into task-based sub-teams.

**H10:** The increase in team performance, with the increase in team familiarity, is highest when the team is organized into task-based sub-teams, lower when the team is flat, and lowest when the team is flat but grouped into social cliques.

**H14:** The rate of increase in team performance, with the increase in team familiarity, is higher for the teams working on non-routine tasks than that for the teams working on routine tasks.

**H15:** The relative difference in team performance across the different team structures is higher for the teams working on non-routine tasks as compared to the teams working on routine tasks.

**H16:** The relative difference in levels of TMM formation across the different team structures is higher for the teams working on routine tasks as compared to the teams working on non-routine tasks.
Chapter 4

Conceptual Framework and Computational Modelling

This chapter presents the details of the conceptual framework and the computational model that serve as the basis to study the correlation between social learning modes, TMM formation, and team expertise in project-teams with varying team structures, levels of team familiarity, busyness levels of agents, and task types, as discussed in Chapter 3.

4.1 Modelling decisions

The characteristics of the cases to study (section 3.2) and the resulting modelling decisions for computational implementation are discussed in this section.

4.1.1 Team

The research hypotheses, proposed in Chapter 3, are based on the characteristics of small teams. Hence, as discussed in section (section 2.2), agents are assumed to have the ability to maintain a mental model of all the other agents in the team.

This research primarily focuses on project-based teams. In such teams, members need not necessarily have worked with each other earlier (Laubacher & Malone, 2002). Thus, familiarity in such teams may range from 0 to 100 %. Even if agents may have been part of the same team earlier, it does not necessarily mean that the agents have a pre-developed mental model of each other at the start of the next project. Hence, team familiarity is defined as the percentage of agents that were part of the same team earlier rather than the percentage of agents that have a pre-developed mental model of each other. At the start of the new project (simulation round), agents with team-familiarity will have varying levels of pre-developed TMMs, depending on their experiences from the previous project.
Project-based teams provide flexibility and opportunities for employees to assume new roles (Devine et al., 1999; Guzzo & Dickson, 1996; Hackman, 1987; Laubacher & Malone, 2002; Lundin, 1995; Mohrman et al., 1995; Packendorff, 1995). Even the project leaders may be nominated on an as-needed basis, to meet the specific requirements of the project (Laubacher & Malone, 2002). Such scenarios may occur in design firms and organizations (Perkins, 2005).

The implemented computational model adopts a similar approach. The team leader is not designated by the experimenter but emerges during the simulation, as appointed by the Client Agent. This involves an initial bidding process in which the Client Agent invites all the team members for the initial proposal to lead the project. Details of how the bidding process is computationally implemented are discussed in section 5.7.1, where architecture of the Client Agent is discussed.

### 4.1.1.1 Team structure and social learning

Social learning opportunities may vary with the team structure, as influenced by the social interaction and observation opportunities available to the team members. Three types of team structures (section 2.2.1, and Figure 2.1) are modelled: flat teams, flat teams with social cliques, and teams organized into task-based sub-teams.

Flat teams allow unrestricted access to all the agents in the team for task allocations as well observations. In flat teams with social cliques, agents can allocate tasks to any other agent in the team, but their ability to observe the other agents is limited to the members within their social cliques. In teams organized as task-based sub-teams, not only is the agents’ ability to observe the other agents limited within the sub-team, but even most of the task-allocation interactions are within the sub-team.

In all the simulations, the leader is chosen by the Client Agent. The three types of team structures implemented are summarized in Table 4.1.

<table>
<thead>
<tr>
<th>Team type</th>
<th>Task allocation</th>
<th>Scope of observation</th>
</tr>
</thead>
<tbody>
<tr>
<td>Flat</td>
<td>Any member of the team</td>
<td>Any member of the team</td>
</tr>
<tr>
<td>Flat with social clique</td>
<td>Any member of the team</td>
<td>Only members of the social group</td>
</tr>
<tr>
<td>Task-based sub-teams</td>
<td>Only members of the task group</td>
<td>Only members of the task group</td>
</tr>
</tbody>
</table>

At the start of the simulation, an affiliation is assigned to each of the agents in the team. This affiliation identifies the agents’ social group or the task group, depending on how the affiliation is defined (social or task-based) for that simulation. An agent that may have expertise in multiple tasks,
relating to more than one task group, is still part of only one group. In that scenario, if the team is divided into task-based groups, the non-group related expertise of that agent may not be used by the team.

Therefore, the opportunities for social learning may vary with the team structure. The limitations to social learning modes may also arise in teams working on face-to-face interactions, if such limitations are related to the cognitive abilities of the team members.

4.1.2 Social learning in team environments

Various forms of social learning opportunities exist in a team environment. The team members may learn about each other through personal interaction with each other; they may learn by observing the other members perform a task; they may learn about the other agents by observing the interaction between the other agents.

For example, in Figure 4.1, if \(A_1\) allocates a task \(T_1\) to \(A_2\) and asks \(A_2\) to pass on the resulting next task \(T_2\) to \(A_3\), then \(A_2\) may assume something about what \(A_1\) might think of \(A_3\)'s capability in \(T_2\). If another agent \(A_4\) observes \(A_1\) allocating the task \(T_1\) to \(A_2\), then \(A_4\) may assume that since \(A_1\) is allocating the task \(T_1\) to \(A_2\), \(A_1\) itself does not have the competence to perform \(T_1\), or else it would have done the task by itself. At the same time, \(A_3\) may also assume that it is likely that \(A_2\) knows how to perform \(T_1\) because it is being allocated that task by \(A_1\).

Now, if \(A_1\) gets feedback from \(A_2\) on whether \(A_2\) can perform the given task \(T_1\) or not, then \(A_1\) will know something about the competence of \(A_2\) in the task \(T_1\). In another instance, if \(A_3\) observes \(A_5\) performing a task \(T_4\), then \(A_4\) knows that \(A_5\) can perform \(T_4\). \(A_4\) may be able to use this knowledge later, if at some other stage, it is looking for someone to perform \(T_4\).

![Figure 4.1: Social learning opportunities in a team environment](image-url)
Such scenarios are common when members are working in a team, and such assumptions (attributions) allow team members to build the mental models of each other (Wallace, 2009; Iso-Ahola, 1977; Olivera, 2004; Kelley, 1973). In making such assumptions, team members inherently attribute intentional and unintentional behaviours to the actions of other team members. The ability of humans to identify with other humans as intentional agents, similar to themselves, allows them to make such assumptions, and learn for social interactions\(^8\) (Malle, 1997; Tomasello, 1999; Knobe, 2002). Details of the social learning modes (i.e., learning from personal interactions, learning from task observations, and learning from interaction observations) and their implementation are presented in section 5.5, where the agent details are discussed.

4.1.3 TMM

As team members interact, they build mental models of themselves and the other members of the team. This allows the team to efficiently coordinate and allocate tasks. The mental model for individual agents is termed as the AMM, and collectively, the AMMs for all the agents within the team forms a TMM in an agent (section 5.3.2, section 5.4.2). Knowing which agent has the expertise in a given task, i.e., knowing “who knows what” is an important part of the TMM. Thus, the development of TMM involves learning about the competence of each agent in each of the different tasks the team needs to perform. The TMM formed by each agent may be different to the TMM of the other agents because all the agents will not have the same interactions and observations.

A well developed TMM allows generalization and audience design. Generalization is the ability of the agent to identify patterns, and learn the causal history of relationships between the enabling factors and the actions of a typical agent (Malle, 2005). Audience design (Bell, 1984) allows the agents to adapt their actions and responses to suit the expected behaviours of the specific agents, or groups, they may need to interact with.

In the simulations, the causal relationships between the enabling factors and actions are pre-defined for the agents, Table 4.2. Hence, the agents are not required to learn this mapping from their experience. However, agents need to learn the details of the knowledge (enabling factor) of each agent. Thus, this research primarily explores the advantages of the audience design benefits of the TMM.

Learning “who knows what” is sufficient for agents’ working on routine tasks. However, when the task is non-routine, agents also need to learn about the capability range of the other agents (section

---

\(^8\) The forms of social learning resulting from personal interactions and observations can extend beyond the examples discussed above to include: explicit information seeking and queries about other team members; instructing team members on how to perform the task, or whom to allocate the task; recommending one member to another; and so on. In the simulations these additional forms of social learning are not considered.
This knowledge of others’ solution capability range allows agents to propose solutions that will be acceptable to the agent evaluating the solution.

Table 4.2: Causal relationships between agents’ enabling factors and actions

<table>
<thead>
<tr>
<th>Enabling factor</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Competence</td>
<td>Competence in a given task determines whether an agent can perform a given task or not.</td>
</tr>
<tr>
<td>Capability range</td>
<td>For agents working on non-routine tasks, the capability range determines what solutions the agent may provide for a given task.</td>
</tr>
</tbody>
</table>

Therefore, if the agents have a well developed TMM, then audience design should allow: (a) allocating the task to an agent that can perform it (applicable to both routine and non-routine tasks), and; (b) proposing a solution that the task allocator will accept because it would conform to the task allocator’s acceptable solution range (needed in case of non-routine tasks).

Details of the computational implementation of AMM and TMM are provided in section 5.3.2 and section 5.4.2.

4.1.4 Busyness

Agents learn from the observations they make. This observable sense data includes agent-task interactions and agent-agent interactions. But this learning is subject to their attention. If an agent is busy when the observable data is available, then the observation is not made in that instance. A “Busyness” factor is introduced for agent’s attention to the observable data.

Busyness is defined at a parametric level rather than at the process level. For example, statements such as, “Half of the time, I am too busy to observe what the others are doing”, are common in team environments. Thus, busyness can be defined as the likelihood that an agent is not able to observe an environmental event or stimuli that occurred at that instance, which the agent could have sensed, if it was attending to it. Mathematically, the above example can be represented by a busyness factor=$\frac{1}{2}$.

*Implementation:* Busyness is implemented as, the probability that an agent is not able to sense the observable data (interactions among other agents, and task-performance by some other agent), available at that instance.

4.1.5 Team familiarity

In a newly formed project team, it is possible that some of the team members have a prior acquaintance. Agents that have prior-acquaintance may have a pre-developed mental model of each other. This may allow the team to develop the team expertise much faster because part of what the
agents need to know is already known to some of them. However, even if the agents may have previously been part of the same team (training project), it does not necessarily mean that the agents have a pre-developed mental model of each other. It is possible that the agents may not have had the opportunity to interact with or observe each other in the training project. Hence, team familiarity is defined as the percentage of agents that were part of the same team earlier (training project) rather than the percentage of agents that have a pre-developed mental model of each other.

For example, statements such as, “Nearly a quarter of the team had worked together in the previous project”, are commonly used. This can be mathematically represented as, Team Familiarity=$\frac{1}{4}$.

At the start of the test project, the agents with team-familiarity may have varying levels of pre-developed TMMs, depending on their experiences from the training project.

*Implementation:* Team familiarity is implemented as the percentage of team members that are retained from the training project onto the test project.

### 4.1.6 Task
The effects of TMM formation may vary with the task. Two types of task are differentiated:

#### 4.1.6.1 Routine tasks
Routine tasks are defined as the tasks that can be performed or executed with certainty if the task performer has all the requisite information and knowledge about the task, independent of the task allocator.

In the design domain such tasks are common in parametric design including parametric re-design of standardized-housing, generating working drawings, and so on.

The outcomes of routine tasks are independent of the task performer, i.e., two or more agents assigned the same task will provide the same solution. These tasks can be computationally modelled as a look-up process. For example, such mappings can be encoded by statements such as, “if the task is $T^1$, the solution is $S^1$.” Thus, depending on whether an agent knows the requisite mapping (e.g. $T^1$-$S^1$) or not, it can either perform the task ($T^1$) with certainty or cannot perform the task at all.

Once an agent has performed the current task, it can access its knowledge base to identify the next task to be performed. This is also implemented as a look-up table. For example, “if the last task performed is $T^1$, then the next task to be performed is $T^2$.”

#### 4.1.6.2 Non-routine tasks
Non-routine tasks are defined as tasks that may have more than one solution (non-unique solutions). For non-routine tasks, a solution cannot be provided with certainty of being accepted, even if the task
performer has all the requisite information and knowledge about the task because the performer may not have enough knowledge about the evaluator.

Non-routine tasks may be further divided into creative and non-creative tasks. For creative tasks, the solution space may not be defined (Gero, 2001). However, only non-creative tasks are modelled in this thesis. Non-routine tasks that are non-creative have a defined solution space. Such non-routine tasks can be modelled as combinatorial search problems (Campbell et al., 1999; Mitchell, 2001; Siddique & Rosen, 2001) such that the task performance requires finding one possible combination of a discrete set of sub-solutions that satisfy the specified requirements. If the solution space is sufficiently large, the number of possible combinations to explore can grow exponentially. For a given non-routine task, multiple solutions may exist. The agents may only know some of the solutions that lie within their capability range. This means that solutions for non-routine tasks are dependent on the task performer, i.e., two agents assigned the same task may provide different solutions because they may have a different capability range (i.e., they may know different solutions for the same task). Thus, the task performers need to identify the solution space acceptable to the evaluator, i.e., solutions must lie within the capability range of the evaluator. Hence, dealing with the non-routine tasks in a team environment will require agents to develop a mental model of the capability range of the other agents.

The following vector-based representation is used for a compatibility-based model of non-routine design tasks with non-dominated solutions.

1. The main task T can be divided into \( \eta \) sub-tasks represented as \( T_1, T_2, \ldots, T_\eta \).
2. For each sub-task let there be \( \mu \) acceptable solutions, e.g., for \( T_1 \) the solutions could be \( T_1(2), T_1(2), \ldots, T_1(\mu) \).
3. A complete solution is a combination of the sub-solutions, and can be represented as an \( \eta \) dimensional vector ConceptSol, shown here as
   \[
   ConceptSol = \sum_{i=1}^{\eta} T_i(j), \text{ where } 1 \leq j \leq \mu \text{ such that there is a solution for each of the } \eta \text{ sub-tasks } (i), \text{ which may be one } (j) \text{ of the } \mu \text{ possible solutions for that sub-task.}
   \]
   For example, the set \{\( T_1(3), T_2(2), T_3(3), \ldots, T_\eta(2) \)\} is a possible solution.
4. Conceptually, the \( \mu \) acceptable sub-solutions can be assumed to represent specific attributes of the sub-solution, for example, the quality of the sub-solution, performance of the sub-solution, and so on. Thus, a value of \( j=1 \) may represent the lowest quality, while the maximum value of \( j=\mu \) represents the highest quality. Hence, if the sub-solutions are added together, the average quality of the overall solution (ConceptSol) can be calculated as
Value of overall solution, \( V_s = \frac{1}{\eta} \sum_{i=1}^{\eta} j(i) \), where \( 1 \leq j \leq \mu \)

5. The overall design space in this scenario is defined by a \( \mu \times \eta \) matrix, Figure 4.2. There is an acceptable range of values for the overall solution depending on the capability of the agent receiving the solution. Thus, a range of acceptable values, “\( 1 \leq V_s \leq \mu \)” means all the solutions are acceptable, while a range, “\( 1 \leq V_s \leq 2 \)” means the range of acceptable solutions is reduced. Within the acceptable range, none of the solutions are dominant.

For example, let us say that there are \( \mu = 3 \) sub-tasks, and \( \eta = 3 \) possible solutions for each of the sub-tasks such that the \( i \)th solution has a value \( i \). Further, let the acceptable value of the overall solution be given by \( V_s \) such that \( 0 \leq V_s \leq 2 \).

Based on the specified requirements, i.e., \( V_s \leq 2 \), either of the following solutions, \( \{T^1(1), T^2(1), T^3(3)\} \), with \( V_s = (1+1+3)/3 = X \), and \( \{T^1(2), T^2(2), T^3(2)\} \), with \( V_s = (2+2+2)/3 = X \), are equally acceptable.

For each of these solutions, the calculated value of \( V_s \) is either less than or equal to 2, which is the acceptable range. However, based on the same conditions, the following solution, \( \{T^1(2), T^2(2), T^3(3)\} \), with \( V_s = (2+2+3)/2 = X \), is not acceptable.

![Figure 4.2: Matrix of solution space for a decomposable task with \( \eta \) sub-tasks each with \( \mu \) possible solutions](image)

Given these conditions,

6. Assuming that all the sub-solutions are compatible, the maximum number of solutions possible is \( \mu^\eta \).
7. The range of values of the overall solution is $\mu$. However, if the Client Agent’s acceptable range for the overall solution is $1/z$ of $\mu$, the acceptable number of solutions reduces to $z (\mu^{\eta^1})$. However, the number of acceptable solutions will reduce at a faster rate if the levels of solution decomposition increase, i.e., whether a sub-solution can be decomposed further. The acceptable solution range will reduce because narrowing the range of solution range at higher levels will narrow the acceptable solution range at each sub-solution level.

In the discussion above, it is assumed that all the sub-solutions are compatible and have equal weight$^9$. Similarly, it is assumed that each sub-task has the same number of alternative solutions ($\mu$). The Client Agent has a pre-coded range for the overall solutions. Similarly, all the agents in the team have their own capability range for the tasks they have expertise in. Thus, a simulation with non-routine tasks requires the team members to collectively generate a solution that falls within the Client Agent’s acceptable range. The teamwork involves coordination$^{10}$ and evaluation of the sub-solutions such that the solutions are compatible in terms of their overall value ($V_s$).

**Solution evaluation strategy:**

In the simulations, the solutions are evaluated at the integration stage following a top down approach, i.e., the solutions for higher-level tasks are completed first. Initially, the Client Agent approves the overall acceptable solution range. The team leader considers this approved range as the boundary limit when evaluating the solutions for the corresponding sub-tasks. Once the team leader approves sub-tasks provided by other team members, those team members consider the approved solution as a benchmark to refine the acceptable solution range for the solutions to be coordinated at the next lower level. This cycle continues until all the tasks are decomposed into the lowest levels, Figure 4.3(b).

If the integrated solution exceeds the upper limit, the evaluator chooses one of the sub-solutions to be reworked. For example, let there be a task $T$, with three sub-tasks $T^1$, $T^2$ and $T^3$ that need to be evaluated at integration level. Let the desired solution range at the integration level be $V_s \leq 3$. Let the sub-solutions proposed by the different agents, working separately on each of the sub-tasks, be $T^1(3)$, $T^2(5)$, and $T^3(9)$. The overall solution, $\{T^1(3), T^2(5), T^3(9)\}$ does not meet the Client Agent’s requirements because the calculated $V_s$ is greater than 3. Therefore, one of the sub-tasks is sent for rework. In this case, $T^3(9)$ is selected for rework because the value for this sub-solution (9) is farthest from the mean (1.5) of the desired overall solution value (0-3). The sub-task to rework is chosen by the

---

$^9$ With this representation it is possible to consider weights and constraints, where some of the sub-solutions may never be compatible with some other solutions. Range of solutions (quality) can also be a constraint, i.e., for a given solution, the difference between the worst quality and highest quality part should be within a specified range. In the simulations, weights and specific constraints are not considered.

$^{10}$ Coordination and evaluation is required for non-routine tasks only.
agent coordinating and evaluating the integrated solution. This cycle of task evaluation and rework continues until the sub-solutions are compatible at each level.

4.1.6.3 Task handling approaches
In general, the task handling approaches vary with the task types. Some tasks need to be allocated in sequence, where sub-tasks can only be performed if the preceding sub-task has been completed. In other cases, sub-tasks can be allocated in parallel (Malone & Crowston, 1994; Malone & Herman, 2003), Figure 4.3. For tasks that can be allocated in parallel, they either need to be independent of each other, or their solutions need to be integrated and validated for compatibility, at a later point in time, during the task handling process. Both kinds of task handling approaches are possible.

Figure 4.3: Sequential and parallel task allocations (a) Purely sequential task allocation (b) Combination of sequential and parallel task allocation

By definition, routine tasks can be completed following a sequential task allocation (Figure 4.3(a)). Non-routine tasks require a combination of sequential and parallel task allocations, Figure 4.3(b). In non-routine tasks, sequential task allocation follows the hierarchy of the sub-tasks, i.e., higher level solutions are to be approved first, before lower level sub-tasks are generated. When a higher level solution is decomposed into multiple lower level tasks for further detailing, the lower level sub-tasks can be allocated in parallel. The top-down approach to task handling is common in project teams, especially in design industry, where the conceptual solutions are approved first, and the detail design follows.
4.1.6.4 Task allocation and team knowledge

The knowledge required to perform the task given by the Client Agent is distributed across the agents that are part of the team such that no agent can perform all the tasks by itself.

In order to achieve higher team performance and efficient utilization of the expertise and knowledge distributed across the team members, it is desirable that sub-tasks are allocated to the agents that have the highest competence in performing the sub-tasks. That is, team performance is likely to improve if the team members are aware of each other’s competence and expertise. In some simulations, it is likely that the team members may not have prior-acquaintance with each other. In such teams, members may make errors in initial task allocation because there is no pre-developed mental model of other agents. On the other hand, if some members of the team have prior-acquaintance at the beginning of the project, they use their pre-developed mental models for task allocation\(^\text{11}\). All agents allocate tasks to the agents that they believe have the highest competence to perform the given task. The TMM developed and maintained by each agent allows that agent to identify the agent with the highest competence for a given task. Details of TMM and its usage in task allocation will be discussed in section 5.3 and section 5.4.

\(^{11}\) In real world scenario, this task allocation may also be affected by other factors, such as trust, strength of social ties, power relationships, mentorship, and so on. These other factors are not considered in this thesis, and are considered for future research. Thus, only competence based task allocation is considered.
Chapter 5
Model Implementation

The computational model is implemented as a multi-agent-system in Java Agent Development Environment (JADE). JADE is a Java based software platform that provides middleware functionalities that facilitate implementation of multi-agent systems and distributed applications (Bellifemine et al., 2007).

5.1 Agent overview

All agents are simple reactive agents. Agents learn about each other based on assumptions of intentionality of actions (section 5.5). In terms of their implementation, the agents have subsumption architecture (Brooks, 1991), with layers of finite state machines, characterized by defined solution space and plausible states for TMM.

Apart from the default JADE agents that are part of the Agent Management System, the developed computational model includes:

1. *Agents working on routine tasks (R-Agent)*: The R-agents perform or refuse the given task, depending on their ability to perform the task. These agents are not required to evaluate solutions provided by others, and nor are they required to coordinate solution integration, because routine tasks are purely sequential (section 4.1.6.3).

2. *Agents working on non-routine tasks (NR-Agent)*: The NR-agents perform or refuse the given task, depending on their ability to perform the task. However, the NR-agents need to choose one among many possible solutions that they can provide, and which they expect to be acceptable to the task allocator (section 4.1.6.2). Thus, the NR-agents are required to build expectations about the capability range of other agents into the TMM. These agents are
required to evaluate solutions provided by others, and they also need to coordinate solution integration, because non-routine tasks are allocated sequentially as well as in parallel (section 4.1.6.3).

3. **Client Agent:** A reactive agent that is not a part of the team, but interacts with the team to call for the initial project bid, nominate the team leader, and approve the overall solution.

4. **Simulation Controller:** A reactive agent that is required to: start and monitor the simulations; check the number of simulation runs; switch between training rounds and test rounds of the simulation; and, shut down the simulations based on the parameters set by the experimenter.

Both the R-Agents and the NR-Agents possess basic social and cognitive abilities that include:

1. Recognition and categorization of messages and their semantics.
2. Decision making and choice relating to (1) whom to allocate the task, (2) which solution to choose or what sub-task to choose for rework (applicable to NR-Agent), and (3) what messages to send.
3. Perception of actions and observations, and interpretation of social interactions and observations based on assumptions of intentionality.
4. Prediction and monitoring related to audience design (applicable to NR-Agent). That is, selecting a solution based on the predicted capability range of the task allocator, with the expectation that, the chosen solution is acceptable to the task allocator. Feedbacks from the task allocator are monitored to update the perceived capability range.
5. Reasoning about themselves and the other agents, in terms of actions and observations. Accordingly, the agents maintain the states of the TMM.
7. Interaction and communication; and,
8. Learning, based on generic rules pre-coded into the agents.

For both the kinds of agents, the plans are pre-coded into the agents. Hence, planning is not required.

### 5.2 Overview of the simulation environment:

A schematic representation of the simulation environment is given in Figure 5.1. Each agent in the team has a unique ID. All the agents must register with the Simulation Controller. At the time of registering with the Simulation Controller, each agent registers its task expertise (tasks that it can perform) and affiliations (task groups / social groups). A single agent may have expertise in multiple
tasks such that multiple agents may have expertise in the same task. Agents must also register with the DF (Directory Facilitator) agent, predefined in the JADE environment to provide “yellow page” services to other agents. In these simulations, the “yellow page” services are used selectively. Team members access the DF agent only to identify group members, but not the details of their expertise. Simulation Controller has access to all the details that may be required to manage the team composition and the simulation runs. If a member leaves the team, or a new member joins the team, the member must deregister or register with the DF agent. Thus, the DF agent maintains a list of the current team members. This list is accessed by the Simulation Controller, which uses the registered profiles of each agent to maintain the team composition, and to ensure that the team has the requisite expertise in each simulation round. At the start of the simulation round, the Client Agent calls for a bid for the first task from all agents in the team. Once the lead agent is chosen by the Client Agent, the team members interact among themselves to complete the task before informing the Client Agent about the completion of the task. Agents in JADE interact through message-based communications.

Figure 5.1: Simulation environment implemented in JADE

A single simulation run consists of two simulation rounds. The first round of simulation is the training round in which agents start with default (experimenter-defined) values. None of the agents have any TMM formed at this stage. Once the training round is completed, a second round of simulation is re-run as the test round. At this stage, depending on the level of team familiarity required in the second round (experimenter-defined), some or all of the agents carry over the TMM formed from the training round to the test round (see section 5.3.5).

The results from the training round are used to measure the levels of TMM formed for different levels of busyness. Measurement of team performance for all the cases (busyness or team familiarity) is based on the results from the test round. The Simulation Controller is responsible for managing the
simulation rounds and the number of simulation runs. Once the test round is complete, the Simulation Controller checks the number of simulation runs completed. If more simulation runs are required, all agents are reset to their default (user-defined) values and the next simulation run is activated. If the required number of simulations is completed, the simulation platform is shut down. Details of the Simulation Controller and simulation lifecycle are provided in section 5.8.

5.3 Implementing the R-Agent (Agent working on routine tasks)

The R-Agents either have complete or no knowledge of the allocated task. Hence, if given a task, the R-agents either perform the task with certainty or show failure to perform the task. Since routine tasks can be performed with certainty, no evaluation is required for a task performed by another agent. Similarly, as discussed in section 4.1.6.1, no compatibility issues need to be checked with the routine tasks.

Figure 5.2 shows the activity diagram for the R-Agents. The R-Agents can sense / receive three kinds of data: (1) a task to perform; (2) feedback / reply for the task allocated earlier by this agent, and; (3) observe interaction between two other agents or another agent and some task.

1. If the received message requires this agent to perform a task, the R-agent looks-up its knowledge base. If the agent does not have the requisite expertise (knowledge), it sends a refusal message, showing failure to perform the given task. While doing so, it also updates its TMM about the contents of its own competence in the given task (i.e., negative update because it cannot perform the task), and the competence of the task allocator in the given task (i.e., negative update because it cannot perform the task). If the agent has the requisite expertise, it performs the task, and updates its TMM about the contents of its own competence in the given task (i.e., positive update because it can perform the task), and the competence of the task allocator in the given task (i.e., negative update because it cannot perform the task). At the same time, the agent also informs the task allocator that the task is done.

Once it has performed the given task, the agent looks-up its knowledge base to check the next task to be performed. If there are no more tasks to be performed, the agent informs the Client Agent that all the tasks have been performed. However, if there is another task to be performed, the agent checks if it has the expertise to perform the new task. If yes, it performs the task, and the same cycle of activities is repeated12. If the agent does not have the expertise

---

12 These activities include updating its TMM about the contents about its own competence in the task, identifying the next resulting task and choosing an agent to allocate the identified next task.
(once again it updates its TMM), it looks up its TMM for the agent with the highest competence in the task, and allocates the task to the target agent.

2. If an agent has allocated a task to a target agent, it always receives a feedback. The feedback is either a refusal (showing failure to perform the task) or confirmation that the task is done. On receipt of the feedback, the agent updates its TMM with the content about the competence of the target agent in the given task.

3. Agents that are not busy at a given time may be able to observe the activities of the other agents in the team. Such observations allow the agent to update the TMM about the competence of the agents that are observed, and the tasks involved in those observations. For a routine task, the update of each agent’s TMM involves either a positive or negative change in the value corresponding to the competence level in the task.

![Activity diagram for the R-Agents](image)

**Figure 5.2: Activity diagram for the R-Agents**

### 5.3.1 Knowledge required for the R-Agents

All the task related knowledge is pre-coded in to the agents. Task related message exchanges are also defined, and based on FIPA (Foundation for Intelligent Physical Agents) protocol. Details of the messages are discussed in section 5.6.
When the team is initially formed, the TMMs, maintained separately by each agent, are not developed. Once the simulation is started, agents develop the separate TMMs as they interact with and observe the other agents. For a specific input and required task, agents either have full or no knowledge of the solution. The task-solution mapping is implemented as a look-up table. All agents know the set of tasks to be performed, and their dependencies (section 4.1.6.1).

The protocol for task handling is known to all the agents. The task handling protocol includes the steps for selection of agent to allocate tasks. This is implemented as “IF-THEN” rules to switch between the different conditions that the agents’ may encounter. That is, whether the agent can perform the next task by itself, or if it has already identified an expert for the task, or if multiple experts have been identified for the same task, as shown in Figure 5.2.

5.3.2 Implementation of TMM for the R-Agent:

When an agent is initialized, it has a default AMM for all the agents in the team. This AMM consists of: (a) task identifier; (b) the number of times the agent has performed the assigned task, \( P_T \); and, (c) the number of times a task has been allocated to the agent, \( G_T \).

The ratio \( \frac{P_T}{G_T} \) defines an agent’s view of another agent’s competence value for a given task. For each agent, there are as many competence values as the total number of tasks. With no prior information, there is an equal chance that an agent may or may not have competence in the given task. Hence, at time \( t=0 \), the ratio \( \frac{P_T}{G_T} \) is taken as \( \frac{1}{2} \) for all the tasks, which is the value ascribed to the agent and is not the intrinsic value. When the agent performs a given task, both \( P_T \) and \( G_T \) are incremented by 1. If an agent cannot perform a given task, only \( G_T \) is incremented by 1.

The AMM is represented as an \( m \)-dimensional vector of the competence values of the \( m \) tasks within the team (Grey column in Figure 5.3). The TMM is represented as an \( m \times n \) matrix (Figure 5.3), where \( n \) is the total number of agents. Each element \( T_{rs} \) represents the competence of the \( s \)th agent for the \( r \)th task (\( T_r \)), such that \( 0 \leq T_{rs} \leq 1 \).

![Figure 5.3: Matrix representing the TMM of the R-Agents](image-url)
5.3.3 Using the TMM for task allocation and handling:

Agents allocate the task to the agent that has the highest competence value in the given task. When the simulation starts (t=0), all the agents have the same default value for the competence in each task. In such a scenario, agents allocate the task to a random agent. If the team has no task-based sub-teams, the task allocation is open to all the agents within the team. If the team is divided into task-based sub-groups, then agents allocate the task to a random agent from the group that relates to the given task.

Once the team members have gained experience working with each other, there will be differences in known competence of agents in a given task. In this scenario, it is possible that more than one agent has the highest competence value. In that case, the agent creates a shortlist of all the agents with the highest competence value, and the task is allocated to an agent randomly selected from this shortlist.

5.3.4 Observing the change in TMM

The TMM formation can be measured in terms of the amount, accuracy, and the importance of the TMM formed (Badke-Schaub et al., 2007; Langan-Fox et al., 2000; Mohammed et al., 2000). In this case, the level of TMM formed is measured as a ratio of the number of matrix elements whose values are different from the initial values by the end of the simulation. Since each agent starts with a default value for each element in the matrix, the values in each element will change only if the agent has learnt it through social interactions and observations.

For example, let there be 10 agents in the team and altogether 10 tasks to be performed by the team. In that case, the TMM is represented as a 10×10 matrix such that there are 100 elements in the TMM. When the simulation starts, all the elements have a default value=1/2. As the agents learn about each others’ capabilities in the different tasks, they update the values of the corresponding elements in the TMM. By the end of the simulation, let us assume that 60 of these values were updated such that the value of each of these 60 elements is different from 1/2. Thus, TMM formation in this case is 60%.

Each agent in the team maintains a separate TMM, which it updates based on its own interactions and observations. Therefore, by the end of the simulation, it is expected that each agent’s TMM will be different. However, overlap and similarities across the TMM of the agents is likely. The overall TMM formation for the team is calculated as an average of the TMM formation for each agent in the team. For example, in a team of 10 agents, if 4 agents have 60% TMM formation, 4 agents have 40% TMM formation, and 2 agents have 50% TMM formation, then the overall TMM formation for the team is 50%.

In these simulations, the kinds of assumptions about what is learned by the agents ensure that what the agents learn is accurate. Hence, the accuracy of the mental model (Badke-Schaub et al., 2007; Rentsch & Hall, 1994) is not measured.
It can be argued that if the agents are only learning the competence and expertise of the other agents, everything that they learn is important. However, even in this scenario, some information is more important than the others. Not all the agents have competence in all the tasks, and the tasks need to be performed in a sequential order. Thus, it is more important for an agent to identify the agents that have competence in the tasks that immediately follow the task performed by this agent. Knowing the competence of agents in tasks that are not immediately related to this agent may not be useful for task allocation, and, hence, not critical to the team performance.

The important TMM is measured directly in some of the experiments with the routine tasks. In general, the efficiency of TMM, as discussed in section 3.2 (hypothesis 7), is used as an indirect measure for evaluating the importance of TMM formed at team levels.

In summary, Efficiency of TMM = Team performance / Level of TMM formation.

5.3.5 Reset TMM

Apart from the messages related to task handling and the project, agents also receive messages from the Simulation Controller that are related to the team composition and the simulation status. If the messages received are related to the changes in team composition, or the start of a new project, agents may need to reset some or all of their TMM to default values. The two possible cases are:

**Case 1:** Starting of training round: if the next round of simulations is a training round, it means it is also the start of a new simulation run. Hence, for all the agents, all the values in the TMM are reset to default values.

**Case 2:** Starting of test round: if the next round of simulations is a test round, it means some or all of the agents may have worked together in the training round. If all the agents are the same in the training round and test round, i.e., if the team familiarity is 100%, all the agents retain their TMM. If the team familiarity is less than 100%, new agents are introduced into the team such that each new agent acquired in the team replaces an agent that was part of the training round. The access to the team list from the DF agent allows each agent to identify the agents that are new in the team. For example, let there be 10 agents, A^1 to A^{10} that were part of the team in the training round. Now, if the desired team familiarity in the test round is 80%, then the new team has 8 agents retained from the training round, and 2 new agents, A^{3'} and A^{7'} such that they replace the other 2 agents, A^3 and A^7, that were not retained from the training round.

While all new agents (i.e., A^{3'} and A^{7'}) start with a default TMM, the agents retained from the training round (i.e., A^1, A^2, A^4, A^5, A^6, A^8, A^9, and A^{10}) reset their AMM of the agents that have been replaced (A^3, A^7) while retaining the AMM of the rest of the agents (i.e., A^1, A^2, A^4, A^5, A^6, A^8, A^9,
and $A^{10}$. That is, the retained agents retain part of their TMM, while the other part that may not be useful (i.e., related to $A^3$ and $A^7$) is reset to default values (to be used for AMM of $A^3'$ and $A^7'$).

5.4 Implementing the NR-Agents (Agents working on non-routine task)

The NR-Agents have complete or no knowledge of the allocated task. Hence, if given a task, the NR-agents either provide a valid solution or show failure to perform the task. However, even if the agent provides a valid solution, the task allocator may not necessarily accept the solution if (1) the solution does not conform to the solution range acceptable to the task allocator, or (2) the solution is not compatible with the solutions for the dependent (parallel) sub-tasks. The NR-Agents need a more detailed architecture than the R-Agents such that they can reason about the acceptable solution range of the task allocator. In addition, each NR-Agent needs to have the ability to evaluate solutions for compatibility at integration level, if needed.

Figure 5.4 shows the activity diagram for the NR-Agents. The NR-Agents can sense / receive four kinds of data: (1) a task to perform; (2) feedback / reply for the task performed earlier by this agent; (3) solution for the task allocated earlier by this agent to another agent, and; (4) observe interaction between two other agents or another agent and some task.

1. If the received message requires this agent to perform a task, the agent looks-up its knowledge base. If the agent does not have the expertise, it sends a refusal message. While doing so, it updates its TMM about the contents of its own expertise in the task (negative update) and the contents of the expertise of the task allocator in that task (negative update).

   If the agent has the expertise, it looks up the range of solutions it can provide. At the same time, the agent checks its TMM\(^{13}\) to get the task allocator’s acceptable range of solutions for the given task. This allows the agents to create a shortlist of the solutions that it expects is acceptable to the task allocator. If there are no solutions within the shortlist, it means this agent cannot provide any of the solutions that it expects to be acceptable to the task allocator. Hence, it refuses to perform the task, and updates its TMM about its own competence. If there are solutions that it can provide, and that it expects to be acceptable to the task allocator, it selects one of the solutions, to perform the task. Once the task is performed, the agent awaits a feedback from the task allocator on the acceptance of the solution.

2. The feedback received by an agent for the task it has previously performed either requires the agent to rework the task, which means the solution provided earlier was not acceptable, or

\(^{13}\) The TMM of NR-Agents store additional details about the competence of agents in the various tasks, as discussed in section 5.4.2. (Figure 5.6)
confirms that the solution was accepted. In either case, the agent learns something about the acceptable solution range of the task allocator for the given task, and updates its TMM about the capability range of the task allocator.

If the agent is required to rework the task, then the cycle of solution selection and waiting for feedback continues until the task is either approved or the agent exhausts all its optional solutions that it could expect to be accepted. If the agent exhausts all its optional solutions, it shows failure to perform the task.

If the task is accepted, the agent checks if the accepted task has to be decomposed into sub-tasks for further detailing. If the current task does not require further decomposition, the agent identifies the next task in the sequence. If there are no more tasks left to be performed, the agent awaits further data to be sensed. Otherwise, it looks up the TMM for a target agent that can perform the next task, and allocates the task to the target agent. The target agent can be the agent itself.

If the current task needs to be decomposed into sub-tasks for further detailing, the agent breaks up the task into sub-tasks. For each of the sub-tasks, the agent looks up the TMM for a target agent that can perform the task, and allocates the task to the target agent.

Once the agent has allocated a task or a sub-task to other agent or agents, it awaits the solutions to be received.

3. When an agent receives a solution for a task from an agent, it learns something about the capability range of the task performer in the given task, and it updates its TMM about the contents of the competence of the task performer. Following that, the agent checks if all the solutions that need to be evaluated together, for integration and compatibility, have been received or not. These solutions correspond to the sub-tasks that were decomposed from the same, upper level task. If all the solutions have not yet been received, the agent waits for other solutions to be received.

Once the solutions for all the sub-tasks have been received, the agent evaluates the solutions for their compatibility. If the solutions are compatible, the agent approves all the solutions, and sends approval messages to all the agents that provided the solutions. At the same time, the agent sends a message to the Client Agent, informing that the partial solution is complete.

If the solutions are not compatible, the agent identifies a sub-task for rework. Details about choosing a task for rework is discussed in section 4.1.6.2 (also see 5.4.4, Figure 5.10). Once an
agent has chosen a sub-task for rework, it looks up its memory\textsuperscript{14} (section 5.4.1) to identify the agent that had performed the task earlier. If the agent that had proposed a solution earlier has not exceeded the maximum number of attempts allowed, then the task is re-allocated to the same agent. Otherwise, in addition to the re-allocation of the task to this agent, the task is also re-allocated to some other agent.

4. Agents that are not busy at any given time may be able to observe the activities of the other agents in the team. For a non-routine task, the update of an agent’s TMM through observations may also include changes to the values for capability ranges, in addition to the changes in the competence levels.

Other messages received by the agent are related to the control of the simulation. These messages are not shown in the activity diagram. Such messages are same for all the agents, whether they are working on routine tasks or non-routine tasks.

\textsuperscript{14} NR-Agents maintain a working memory of the task allocation and number of attempts taken by task performers, which helps them coordinate task integration and task allocation. This working memory is implemented as a look-up table (section 5.4.1).
5.4.1 Knowledge required for the NR-Agents

For both the R-agents and the NR-agents, the task related knowledge is pre-coded. These include: details for task handling and task related message exchange; details of tasks to be performed and their dependencies; and, protocols for task handling. While the task handling processes for the NR-agents
serve similar purposes as for the R-agents, the non-routine tasks need to be decomposed and the solutions need to be integrated. This decomposition and integration requires additional knowledge for task coordination and evaluation. All the knowledge of task coordination and assessment of the compatibility of the integrated solution are pre-coded into the agent’s knowledge base. The solution evaluation strategy was discussed in section 4.1.6.2. Pseudo code for implementing the solution evaluation strategy is given in Figure 5.5.

Given:
Number of sub-tasks for task T is \( \eta \)
Number of possible solutions for each sub-task is \( \mu \)
Acceptable solution range for T is between \( L_R \) to \( U_R \)
Received List=list of sub-solutions received

If received a solution \( T_i(j) \), add to Received List
If Size (Received List) \( \neq \eta \), wait
Else {  // Get overall solution value at integration stage \( V_S \), using

\[
V_s = \frac{1}{\eta} \sum_{i=1}^{\eta} j(i) \quad \text{// where } 1 \leq j \leq \mu
\]

If \( L_R \leq V_S \leq U_R \) { approve all sub-solutions}
Else {  Select a task for rework (for details see Figure 5.10)

Reallocate the task to target agent(s) (for details see Figure 5.9)
}

}

Figure 5.5: Pseudo codes for selecting task for rework (non-routine tasks)

**Maintaining a working memory**

The NR-Agents need to maintain a working memory that stores the information about the number of attempts taken by the task performers for the tasks that are active, i.e., tasks for which the solutions have not yet been accepted.

This working memory is implemented as a look-up table. For example, Let \( A^2 \) be the task allocator that needs to maintain the details about the task \( T^1 \), which is one of the tasks that it has allocated. Thus, if the task \( T^1 \) is active, the working memory of \( A^2 \) for the allocated tasks contains the identity of the agent \( A^1 \) that has proposed a solution for \( T^1 \), and the number of attempts that \( A^1 \) has taken so far, i.e., the number of non-accepted solutions \( A^1 \) has proposed thus far for \( T^1 \), in the current project.

Once the solution for the task \( T^1 \) is accepted, the number of attempts for \( T^1 \) is not useful anymore. Thus, this information is erased from the working memory. The working memory is independent of the TMM. Thus, the reset and update of working memory does not affect the reset and update of TMM.
5.4.2 Implementation of TMM for the NR-Agent:

The AMM for the NR-agents consists of: (a) task identifier; (b) the number of times the agent has performed the task assigned, \( P^T \); (c) the number of times a task has been allocated to the agent, \( G^T \); (d) perceived lower range of solution for the task, \( L_R \) and; (e) perceived upper range of solution for the task, \( U_R \).

Thus, the AMM and TMM for the NR-agents are similar to the AMM and the TMM for the R-agents, but with additional details for the capability range. For the NR-Agents, the AMM is represented as an \( m \)-dimensional vector of vectors showing the competence values, the lower range, and the upper range of the \( m \) possible tasks (Grey column in Figure 5.6). The TMM is represented as an \( m \times n \) matrix (Figure 5.6), where \( n \) is the total number of agents. Each element \([T^r, L_{Rr}, U_{Rr}]\) is a vector that holds the values for the competence, the lower range and the upper range of the \( s \)th agent for the \( r \)th task.

Figure 5.6: Matrix representing the TMM of an agent working on non-routine tasks

As is the case with the R-agents, the default value of \( P^T/G^T \) at time \( t=0 \) is \( \frac{1}{2} \). At \( t=0 \), the default values of \( L_R \) and \( U_R \) for each of the tasks is set to \( L_{Rmin} \) and \( U_{Rmax} \) respectively, where \( L_{Rmin} \) is the minimum possible lower range, and \( U_{Rmax} \) is maximum possible upper range for any solution. Thus, unless the details about an agent’s capability range are known, an agent can be expected to provide any of the possible solutions.

Besides the \( L_R \) and \( U_R \), agents also maintain the temporary values for the lower and upper range, \( L_T \) and \( U_T \) respectively. \( L_T \) and \( U_T \) are calculated as the range acceptable to an agent, based on the solution that it has already accepted in a given project. For example, for an agent \( A^0 \) let the perceived range of acceptable solutions for task \( T^0 \) be from \( L_R=2 \) to \( U_R=9 \). In a given project, let that agent \( A^0 \) accept a solution for \( T^0 \) from agent \( A^1 \) such that the value of the accepted solution is 8. In that case, agent \( A^1 \)
needs to coordinate the sub-tasks for \( T^0 \) such that the overall solution for \( T^0 \) is close to the value 8, within a limit acceptable to \( A^0 \). Let us say, in this case, this acceptable limit is from 6 to 9. In that scenario, \( L_T=6 \) and \( U_T=9 \). The temporary range, defined by \( (U_T - L_T) \) is always a sub-set of the overall perceived range \( (U_R - L_R) \) for any agent. The temporary values of lower and upper range are specific to the project (simulation round), and are erased once a project is over. These values are not carried over even if the agents retain their acquired TMM to the next project.

5.4.3 Updating AMM and TMM:

When an agent receives a positive feedback on another agent’s competence, both \( P^T \) and \( G^T \) are incremented by one. If a negative feedback is received, only the \( G^T \) value is incremented by one.

For the NR-Agents, merely updating the competence values is not enough. Agents check the solutions provided or rejected by a target agent to update the capability range of the target agent, in the given task. If an agent provides a solution or accepts a solution, it means that the solution lies within the specified range of solutions for that agent, in the given task. If an agent rejects a solution provided by someone else, it can be assumed that the solution is outside the range of solutions acceptable to that agent, in the given task. Pseudo code for the update of the required solution range is provided in Figure 5.7.

```plaintext
Given Agent A, Task T, Solution received T(N), MaxWindow, MinWindow
CurL_R = current lower range in TMM for agent A in task T
CurU_R = current upper range in TMM for agent A in task T
CurL_O = lowest range observed and registered (not hypothesized) for agent A in task T
CurU_O = highest range observed and registered (not hypothesized) for agent A in task T

Case 1: Solution proposed by agent A
currentRange = CurU_R - N
If (CurU_R ≥ N) and (CurU_R ≤ N))
{   If (currentRange = MaxWindow) { do nothing }
   Else
   {     U_buffer = CurU_R - N
          If (U_buffer > (MaxWindow - 1)) { CurU_R = N + MaxWindow - 1}
          Else if (N - (MaxWindow - 1) ≤ 0) { CurL_R = 0 // assuming 0 is lowest possible value}
          Else { CurL_R = N - (MaxWindow - 1)}
   }
}
Else if (CurU_R < N) { CurU_R = N; CurU_O = N; }
Else if (CurL_R > N) { CurL_R = N; CurL_O = N; }
```
**Case 2:** Solution rejected by agent A

If \((N \leq \text{CurUR})\) and \((N \geq \text{CurLR})\)

\[
\begin{cases}
\text{If } ((\text{CurUR} - N) \leq \text{MinWindow}) & \text{\{ If } ((N - 1) \leq \text{CurUO}) \text{\{ CurUR=N-1 \}} \text{ Else } \{ \text{CurUR=CurUO} \} \} \\
\text{If } ((N - \text{CurLR}) \leq \text{MinWindow}) & \text{\{ If } ((N + 1) \leq \text{CurLO}) \text{\{ CurLR=N+1 \}} \text{ Else } \{ \text{CurLR=CurLO} \} \} \\
\end{cases}
\]

*Figure 5.7: Pseudo code for update of acceptable solution range*

As part of the common knowledge about the capability range of a typical agent in the team, agents assume that the difference between the upper range and lower range of any agent’s capability, for a given task, are similar. Conceptually this means that if an agent provides high quality solutions, the solutions provided by this agent are very unlikely to be lower than a particular range. Similarly, an agent that provides low quality solutions may not be able to provide a solution higher than its capability range. In Figure 5.7, the span of solution range for a typical agent is discussed in terms of the MaxWindow and the MinWindow. MaxWindow refers to the maximum possible span, while MinWindow refers to the minimum possible span.

For example, Let us assume that for any task, the upper and lower range of valid solutions has a value 9 and 0 respectively. Let MaxWindow=4 and MinWindow=2. Now, if an agent provides a solution with value 9, and, if the agent has the maximum capability span (i.e., span=MaxWindow) in this task, then this agent provides a solution between 9 and 6 (9-4+1). However, if the agent has minimum capability span (i.e., span=MinWindow) in this task, then the agent only provides solutions between 9 and 8 (9-2+1). Thus, the solution span refers to the number of optional solutions that an agent provides for a given task. It is assumed that the solution options that an agent can provide fall in a continuous range, within a limited span\(^{15}\).

Figure 5.8 schematically represents the concept of the solution capability span of an agent. Thus, agents A\(^1\) and A\(^2\) have a solution span of 3 each, while the agent A\(^3\) has a solution span of 4. Conceptually, agents A\(^1\) and A\(^2\) provide a solution in mid-range, while agent A\(^3\) provides the solutions in higher range. Given the capability span of the three agents in this case, it can be inferred that MaxWindow=4 (largest span, as seen in case of A\(^3\)), and MinWindow=3 (smallest span, as seen in case of A\(^1\) and A\(^2\)) for a typical agent.

In the simulations, MaxWindow and MinWindow values are pre-coded into the agents as part of their common knowledge about the typical agents of the team.

---

\(^{15}\) The assumptions relating to the solution span is important to the simulation because they provide a reference for agents to build expectations on likely capability range of the other agents. Without the solution span, the narrowing of the solution span will not be expectation based, rather it will be absolute, i.e., only those solutions that have not been accepted will be removed from the capability range. In that case, teams may take longer to converge to solutions at each integration level. This may reduce the differences in team performance across the different simulations, and hence, a pattern in results may be difficult to observe.
When an agent observes another agent either performing or rejecting a task, the observer agent uses the known values of the typical MaxWindow and MinWindow to calculate and update the likely span (lower and upper capability range) of the observed agent for the observed task, Figure 5.7.

5.4.4 Using the TMM for task allocation and handling:

For the NR-Agents, the use of TMM for task allocation and handling is similar to the R-Agents. However, the NR-Agents also need to consider the number of attempts taken by an agent, which has already been allocated the task earlier. Beyond a threshold value, given by the maximum number of attempts (experimenter defined), an agent may not be the only one to be assigned the rework. In that case, besides assigning the rework to that agent, one more agent is chosen for task-allocation. Figure 5.9 shows the pseudo code for the selection of an agent for task allocation.

---

**Figure 5.8: Capability of each agent is defined by a typical solution span**

---

Given: Task T  
Create lists AlreadyTried, Available  
AgForRework=getCurrentActorStatus (T) // agent to allocate the task to  
// Function getCurrentActorStatus checks if there is an agent that has already been assigned this task earlier. // If no, it returns “ No Current Actor” else it returns Agent name and N^At =current number of attempts  
If (AgForRework ≠ “ No Current Actor”)  
{  
Get N^At  
If (N^At < MaximumAttempts)  
{  
Allocate rework to current actor  
Increment N^At for current actor }  
Else { Add current actor to AlreadyTried list  
Remove current actor from available list  
Look-up TMM for newActor with highest competence for T such that newActor is in Available list  
Allocate task to last agent added to Already Tried list, and to the newActor  
Set newActor as current actor }  
}  
Else  
{  
Look-up TMM for newActor with highest competence for T
Allocate task to newActor
Set newActor as current actor

Figure 5.9: Pseudo code for selection of agent for task allocation (non-routine task)

For non-routine tasks, the selection of the task for rework is also dependent on the TMM. The task allocator needs to evaluate the sub-solutions together at the integration level, and choose a sub-task for rework so that the reworked solution is most likely to ensure that the overall solution lies within the acceptable solution range. The need to select a task for rework arises only if the value of the overall solution does not fall within the desired range. This process of selecting a task for rework involves a distance measure for the solution values. As discussed in section 4.1.6.2, the sub-solution farthest from the mean of the acceptable solution range is chosen for rework. Pseudo code for the selection of the task for rework is provided in Figure 5.10.

Given: TaskToCoordinate, i.e., a set of sub-solutions to evaluate for compatibility at integration level
Look-up records to identify agent $A^0$ that approved higher level solution for TaskToCoordinate
Get temporary lower ($L_I$) and temporary upper ($U_I$) range of solutions acceptable by $A^0$ for TaskToCoordinate
Desired mean ($M$)=($L_I$ + $U_I$); // mean of the desired range of solutions
Q=number of sub-tasks for TaskToCoordinate // Thus Q sub-solutions are to be evaluated
Let $d_{max}$=0; // to be used to identify the solution with value furthest from $M$
For (e=0; e < Q ; e++)
{
    $S_e$=Value of the ith sub-solution
    $d_e$=$S_e$ – $M$; // distance of eth solution from desired mean of overall solution range
    If ($d_e$ > $d_{max}$)
    {
        $d_{max}$ = $d_e$;
        taskForRework=TasktoCoordinate (e); // eth sub-task for TaskToCoordinate
    }
Return taskForRework

Figure 5.10: Pseudo code for selection of task for rework

For the non-routine tasks, the solution selection is based on the capability range of the task performer and the acceptability range of the task allocator. The task performer looks up its TMM for the capability range of the task allocator corresponding to the given task. For the selected solution (which is one of the solutions in the task performer’s capability range) to be accepted, the solution must also overlap with the solution range acceptable to the task allocator. Once the agent has identified a shortlist of solutions that it can provide and that are also acceptable to the task allocator, it can choose any of the solutions from the shortlist, provided the chosen solution has not already been proposed in the same project. The pseudo code for solution selection is provided in Figure 5.11.
Since the agent constantly updates the task allocator’s acceptable solution range, the task performer is able to adapt the solution to suit the task allocator, showing the characteristics of audience design.

Given: Task T, Task Allocator A
Look-up knowledge base for expertise on T
If (expertise not found) { show failure}
Else
{
    Create a list called shortlist
    Look-up knowledge base for Capability range (self) for T, say CapList
    Look-up TMM to get Task Allocator A’s acceptable solution range for T, say AccList
    For each solution $T^S$ in acceptable range CapList
    {    For each solution $T^T$ in acceptable range AccList
        {  If ($T^S = T^T$)
            { add $T^S$ to shortlist
                Break } }
    }
    If (shortlist ≠ null)
    {    For each solution $S^S$ in shortlist
        {    For each solution $S^A$ in the list AlreadyTried; // list of solutions proposed earlier
            {  If ($S^S = S^A$)
                { remove $S^S$ from shortlist
                    Break; } } }
    }
    If (shortlist ≠ null) { Select random solution from shortlist}
    Else { show failure }
}

Figure 5.11: Pseudo code for selection of solution

5.4.5 Observing the change in TMM for the NR-Agents
The TMM formation for the NR-Agents is measured the same way as the TMM formation is measured for the R-Agents. Since the TMM of the NR-Agents is more detailed than the TMM of R-Agents, in order to keep the two measures similar in every way to enable comparisons, only the changes in the competence values are considered for assessing the TMM formation of the NR-Agents. The changes in the values of lower and upper capability ranges are considered separately. It is likely that the rate at which the agents learn about the capability range of other agents will be different to the rate at which they identify the other agents’ expertise areas, i.e., tasks for which other agents can provide at least one
solution. The capability range values will be updated (i.e., learnt) only if in the given task related interaction or observation, there are changes required\textsuperscript{16} to the default values for narrowing down the capability range.

Thus, for the NR-Agents, TMM formation is measured in two parts. The first part matches with the TMM formation measures used for the R-Agents. The second part measures the ratio of the number of default capability range values that have changed to the total number of default capability range values at the start of the simulation round.

The conditions for reset of the TMM of the NR-Agents are the same as that of the R-Agents.

5.5 Implementing learning in agents

Agents learn as they interact with their environment, which includes the task and the other agents. This interaction with the environment includes the observations that the agents make. For the R-Agents, learning is primarily limited to knowing “who knows what”. The NR-Agents are also capable of learning individual agent’s capability range for the solutions. Agents learn from their personal interaction with the other agents and through the observations, Figure 5.12. Only the task-related interactions in the team are considered.

In terms of task handling, all the agents in the model consider other agents to be similar to themselves in their intentions and goals. This means: (a) if an agent has the competence to perform a task, it will; (b) agents always intend to allocate a task to an agent that they expect to have the highest competence to do the task; and, (c) agents will refuse to do a task only if they do not have the competence to do it. These assumptions about others’ intentions and goals allow agents to learn about each other’s mental states as they interact with their environment. Learning is rule-based, as given in Table 5.1.

![Figure 5.12: Learning opportunities in a team environment (symbols defined in Table 5.1)](image)

\textsuperscript{16} For example, the default capability range is taken as 0-9. If the agent proposes a solution with value 2, and the solution is accepted, it means the same capability range is retained because there is no further information. However, if the solution is not accepted, it means 2 is outside the capability range of the task allocator. Since the MinWindow of the task allocator=3 and MaxWindow=5, and the solution range is assumed to be continuous, 0 and 1 are also ruled out. Therefore, the capability range is narrowed down to 3-9.
Table 5.1: Learning assumptions corresponding to learning opportunities shown in Figure 5.12

<table>
<thead>
<tr>
<th>Condition (IF)</th>
<th>Deduction (THEN)</th>
</tr>
</thead>
<tbody>
<tr>
<td>If an agent $A^1$ allocates a task $T^1$ to another agent $A^2$</td>
<td>then $A^2$ knows that $A^1$ does not have competence to perform task $T^1$</td>
</tr>
<tr>
<td>If an agent $A^2$ gives a feedback to another agent $A^1$ that had allocated task $T^1$ to $A^2$</td>
<td>then $A^1$ knows about $A^2$’s capability for $T^1$</td>
</tr>
<tr>
<td>If an agent $A^3$ receives a task $T^2$ from another agent $A^2$</td>
<td>then $A^3$ knows that $A^3$ has the competence to perform the task preceding $T^2$ (i.e., $T^1$) as per the task dependencies</td>
</tr>
<tr>
<td>If an agent $A^1$ observes another agent $A^3$ allocating task $T^3$ to a third agent $A^4$</td>
<td>then $A^1$ knows that: $A^3$ does not have the competence to perform task $T^3$.</td>
</tr>
<tr>
<td>If an agent $A^1$ observes another agent $A^5$ performing Task $T^4$</td>
<td>then $A^1$ knows that $A^5$ has the competence to perform the task $T^4$</td>
</tr>
</tbody>
</table>

Three types of learning capabilities are considered:

1. Learning from personal interaction (PI): agent learns when it is directly interacting with another agent. This includes task allocation and response to an allocated task.
2. Learning from task observation (TO): when an agent is dealing with a task, i.e., either performing the task or failing in the process of performing a task, the observer can learn about the agent’s expertise in the given task.
3. Learning from interaction observation (IO): when two agents are exchanging information, i.e., allocating task from one to another, replying back on an allocated task, or proposing a solution to the other, the observer can learn about the sender’s as well the as the receiver’s expertise in the given task.

The typical interaction between any two agents dealing with the non-routine task involves exchange of more messages than the agents working on the routine tasks, Figure 5.13. Hence, the scope of learning from observations is expected to be higher in case of the non-routine task.
Implementing agent interactions and observations

All interaction and observations are through message exchange. All messages sent from one agent to the other are wrapped in a message envelope based on FIPA-ACL message protocol. Table 5.2 lists the typical parameters in the FIPA-ACL message envelope. The parameters marked in grey shaded zone are either pre-defined, default, or not required for the messages exchanged in the simulations. Rather than defining the ontology separately, the knowledge for parsing the message has been encoded into each agent.

The performative “CFP” stands for Call for Proposals, and is used when an agent either calls for bids, or assigns a task in the hope of receiving a solution from the target agent. Similarly, “INFORM” is used to tag a message that provides some information on the allocated or performed task. The “INFORM” message may relate to a task performed (Done), or a solution accepted or approved.

Table 5.2: Parameters in a typical FIPA-ACL message envelope

<table>
<thead>
<tr>
<th>Parameters</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sender</td>
<td>Identity of the sender of the message</td>
</tr>
<tr>
<td>Receiver</td>
<td>Identity of the intended recipient(s) of the message(s)</td>
</tr>
<tr>
<td>Reply to</td>
<td>Which agent to direct subsequent messages to within a conversation thread</td>
</tr>
<tr>
<td>Performative</td>
<td>Type of communicative act of the message e.g. inform, refuse etc</td>
</tr>
<tr>
<td>Content</td>
<td>Content (main body) of the message</td>
</tr>
<tr>
<td>Language</td>
<td>Language in which the current parameter is expressed</td>
</tr>
<tr>
<td>Encoding</td>
<td>Specific encoding of the message content</td>
</tr>
</tbody>
</table>
The “Sender” and the “Receiver” fields are used by the agent management system (AMS) to deliver the message. The primary information for the interacting agents is contained in the “Content” and the “In-Reply-To” fields. When a task is refused, the In-Reply-To field is accessed to identify which task has been refused. Similarly, a solution proposed in the content field is in reply to the task listed in the In-Reply-To field. Details of the different message types used in the reported simulations are provided in Table 5.3. Three types of messages are listed.

1. Task handling messages are the messages exchanged between the agents in the simulation environment for exchange of information about the task.

2. Simulation control messages involve the Simulation Controller as one of the interacting agents. These messages are used explicitly for managing the simulation.

3. The messages in the AMS are default messages that are sent when either the simulation platform or an agent is launched or closed down.
## Table 5.3: Types of messages used and their description

<table>
<thead>
<tr>
<th>Message semantic</th>
<th>Performative used</th>
<th>Sender/Receiver(s)</th>
<th>Content field</th>
<th>In-Reply-To field</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Task handling messages</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>First task</td>
<td>CFP</td>
<td>Client/ All</td>
<td>“First task”</td>
<td>-</td>
<td>Call for bid from the Client Agent to all the team members to lead (first task) the project</td>
</tr>
<tr>
<td>Task</td>
<td>CFP</td>
<td>Source/ Target</td>
<td>“Task”</td>
<td>[ Task label ]</td>
<td>Task assigned by one agent (source) to another agent (target)</td>
</tr>
<tr>
<td>Accepted</td>
<td>INFORM</td>
<td>-do-</td>
<td>“Accepted”</td>
<td>[ Task label, Solution ]</td>
<td>Inform the target agent that the solution it proposed is accepted</td>
</tr>
<tr>
<td>Approved</td>
<td>INFORM</td>
<td>Source/ Client</td>
<td>“Approved”</td>
<td>[ Task label, Solution ]</td>
<td>To inform the Client Agent that the partial solution coordinated by this agent is complete</td>
</tr>
<tr>
<td>Show Failure</td>
<td>REFUSE</td>
<td>Target/ Source</td>
<td>“Failure”</td>
<td>Task label</td>
<td>A target agent shows inability to perform a given task</td>
</tr>
<tr>
<td>Done</td>
<td>INFORM</td>
<td>-do-</td>
<td>“Done”</td>
<td>Task label</td>
<td>For routine tasks: the target agent informs the source agent that the given task is completed</td>
</tr>
<tr>
<td>Done</td>
<td>INFORM</td>
<td>-do-</td>
<td>[ “Done”, Solution ]</td>
<td>Task label</td>
<td>For non-routine tasks: the target agent proposes a solution to the source agent for the given task</td>
</tr>
<tr>
<td>Observation</td>
<td>INFORM-IF</td>
<td>Actor/Observer(s)</td>
<td>Ex-Msg-Copy</td>
<td>[ Original Sender, Original Content, Original In-Reply-To, Original Receiver, Original Performative ]</td>
<td>Whenever an agent (actor) sends a message to another agent, a duplicate message is sent to all the other agents that can observe the interaction or action of the actor (sender)</td>
</tr>
<tr>
<td><strong>Simulation Control messages</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Go-Register</td>
<td>INFORM-IF</td>
<td>Controller/Agent(s)</td>
<td>“Go-Register”</td>
<td>-</td>
<td>The Simulation Controller asking the agent to register with the DF (only the agents in the current team are registered in a given simulation round)</td>
</tr>
</tbody>
</table>
Go-De-register INFORM-IF -do- “Go-De-register” -

The Simulation Controller asking the agent to de-register with the DF in case of Attrition

Next round INFORM-IF -do- “Next round” -

Message to the team members to start the next project (On receipt of this message, for all agents with prior acquaintance, members retain the TMM formed from previous projects. However, the temporary values related to the tasks in the specific project are erased.)

Next run INFORM-IF -do- “Next run” -

Message to all the agents after completion of one simulation run (Upon receipt of this message, the TMM and all the values are reset to the default experimenter-given values to start another simulation run.)

| Default JADE Agent management System (AMS) messages |
|-----------------|-----------------|-----------------|-----------------|-----------------|
| Request         | -               | -               | -               | -               |
| Register        | -               | -               | -               | -               |
| Deregister      | -               | -               | -               | -               |
Based on the task knowledge and the task handling protocols, agents are able to assign values to the required parameters while sending the message. For example, if an agent $A^i$ receives a routine task $T^i$ from agent $A^s$ to perform, the message envelope contains the details such as: Performative=CFP; Sender=$A^s$; Receiver=$A^i$; Content=$T^i$, and so on.

After receiving this message, $A^i$ checks the performative. Since the performative is CFP, it knows that it has received a task to perform. $A^i$ then checks the details of the task to perform. It looks-up its knowledge base to check if it can perform the task $T^i$. If $A^i$ cannot perform the task $T^i$, it needs to send a refusal message to $A^s$. In order to do that, $A^i$ creates a message for delivery with the details such as: Performative=REFUSE; Sender=$A^i$; Receiver=$A^s$; Content=REFUSE; In-reply-To=$T^i$, and so on. However, if $A^i$ could perform the task $T^i$, it needs to inform $A^s$ that the task $T^i$ has been done. In that case, the message sent by $A^i$ to $A^s$ have details such as: Performative=INFORM; Sender=$A^i$; Receiver=$A^s$; Content="Done"; In-reply-To=$T^i$, and so on.

On receipt of this feedback message, $A^s$ checks the performative and the message details to process it, as done by $A^i$.

Since the interaction in JADE is entirely based on message exchange, the observation of the interaction between two agents by other agent(s) or the observation of the task performed by some other agent is also implemented using message transfer. Thus, when an agent sends a message to another agent or performs a task, a duplicate message is sent to all the agents that are not busy at that instance. The content parameter of the duplicate message is marked distinctly as “Ex-msg-copy” (in Table 5.3), and contains the details of the interacting agents and the contents of the interaction in the In-Reply-To parameter. The details of the duplicate message (as provided in the In-Reply-To parameter) meant for observation is sent as a vector of five elements to include:

[Original Sender, Original Content, Original In-Reply-To, Original Receiver, Original Performative]

Thus, upon parsing the message, the observer can identify the original sender (Original Sender) and receivers (Original Receiver) of the message, and what the message conveyed (Original Content), in response to what (Original In-Reply-To).

For example, if $A^i$ performs a task $T^i$, it needs to inform the task allocator $A^s$ that the task $T^i$ has been done. In that case, the message sent by $A^i$ to $A^s$ has details such as: Performative=INFORM; Sender=$A^i$; Receiver=$A^s$; Content="Done"; In-reply-To=$T^i$, and so on. At that point of time in the simulation, for each of the other agents in the team, it is checked if the agent is busy or is it able to observe $A^i$ informing $A^s$ about the completion of task $T^i$. To each agent $A^y$ that is not busy, a duplicate message is sent with the details such as: Performative=INFORM; Sender=$A^i$; Receiver=$A^y$; Content="Ex-msg-copy"; In-Reply-To={$A^i$, "Done", $T^i$, $A^s$, CFP}, and so on. $A^y$, the recipient of the
duplication message, ignores the details of the message sender, but parses the details provided in the 
In-Reply-To field. Parsing these details, A^y knows that A^1 can perform the task T^1. If only task 
observation is modelled, only the details of who performed the task are extracted. If an interaction 
observation is to be modelled, then A^y only extracts the details of the interaction, i.e., A^w allocated a 
task to A^1, which allows A^y to know who interacted with whom, and about what.

Therefore, all the messages for observation have the same representation. Task observation and 
interaction observation are differentiated in the way the agent perceives the data. Details of how the 
different learning capabilities are implemented are provided in Table 5.4. The first column in Table 5.4 
shows the conditions, i.e., which message is being observed. Actions show how the messages are 
perceived, and how the TMM is updated. The last four columns show the applicable TMM updates, 
based on the learning modes available to the agent.

1. When an agent learns only from personal interaction (PI), it does not perceive any of the 
duplicate messages. The agent has no opportunity to observe the social interactions.
2. When the agent learns from task observations (TO), in addition to PI, i.e., (PI+TO), then it 
either observes an agent perform a task or not able to perform the task. But in this case, the 
agent does not know the other details such as, who allocated the task to this agent.
3. When the agent learns from interaction observations (IO), in addition to PI, i.e., (PI+IO), it 
may not know the details of the task performed but it does observe who allocated the task, and 
who was assigned the task. Similarly, if an agent is informing the other agent about the 
completion or refusal of a task, the observer knows something about the task allocator as well 
as the task assignee. When an agent allocates the next task to itself, i.e., if sender=receiver, 
there are no interactions to observe.
4. When the agent learns from both TO as well as well IO, in addition to PI, the agent not only 
observes an agent performing or refusing a task, but it also observes the task allocator.
Table 5.4: Implementing observations: conditions and updates

<table>
<thead>
<tr>
<th>Condition</th>
<th>Action</th>
<th>PI</th>
<th>PI+ IO</th>
<th>PI+ TO</th>
<th>PI+ TO+IO</th>
</tr>
</thead>
<tbody>
<tr>
<td>OriginalContent == [ “done”, Solution ]</td>
<td>UpdateLandULimit (OriginalSender, OriginalInReplyTo)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UpdatePositiveAMM (OriginalSender, OriginalInReplyTo)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1. sender == receiver</td>
<td>-</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2. sender # receiver</td>
<td>UpdateNegativeAMM (OriginalReceiver, OriginalInReplyTo)</td>
<td>X</td>
<td>\checkmark</td>
<td></td>
<td></td>
</tr>
<tr>
<td>OriginalContent == &lt; “Task” &gt; &amp; OriginalPerformative== = CFP</td>
<td>No Interaction, hence nothing to observe</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3. sender == receiver</td>
<td>UpdateNegativeAMM (OriginalSender, relatedInfo)</td>
<td>X</td>
<td>\checkmark</td>
<td></td>
<td></td>
</tr>
<tr>
<td>4. sender # receiver</td>
<td>UpdatePositive AMM (OriginalSender, getPreceedingTask (relatedInfo))</td>
<td>X</td>
<td>\checkmark</td>
<td></td>
<td></td>
</tr>
<tr>
<td>(sender # receiver) &amp;&amp; rework</td>
<td>UpdateLandUNegative (OriginalSender, relatedInfo, rejectedSolValue)</td>
<td>X</td>
<td></td>
<td>\checkmark</td>
<td>\checkmark</td>
</tr>
<tr>
<td>OriginalContent == [ “REFUSE” ]</td>
<td>UpdateNegativeAMM (OriginalSender, OriginalInReplyTo)</td>
<td>X</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>UpdatePositive AMM (OriginalSender, getPreceedingTask (OriginalInReplyTo))</td>
<td>X</td>
<td></td>
<td>\checkmark</td>
<td>\checkmark</td>
</tr>
<tr>
<td></td>
<td>UpdateNegativeAMM (OriginalReceiver, OriginalInReplyTo)</td>
<td>X</td>
<td>\checkmark</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
5.7 Implementing Client Agent

The Client Agent is not a part of the team but interacts with the team to call for the initial project bid, nominate the team leader, allocate the task, and approve the overall solution. These are the broad activities of the Client Agent, irrespective of whether the teams are working on routine tasks or non-routine tasks. The primary differences in the Client Agent’s activities in the two scenarios (routine tasks vs. non-routine tasks) are related to:

5.7.1 Bid selection process

The Client Agent receives the proposed bids from the agents. In case of the routine task, any of the bids can be selected at random because no matter which agent bids, the proposals of all the agents that can perform the task will be the same. However, if the task is non-routine, the proposals from different agents are likely to be different because each agent can propose different range of solutions for the same task. Thus, the Client Agent needs to select a bid that is the best match to its own acceptable range of solutions. Figure 5.14 provides the pseudo code for the selection of bids for non-routine tasks.

```
Bidlist = list of bids received by the Client Agent
CLR=Acceptable lower range of solution for Client Agent
CUR=Acceptable upper range of solution for Client Agent
iLR=Acceptable lower range of proposed solution in ith bid in the bidlist
iUR=Acceptable upper range of proposed solution in ith bid in the bidlist
Create bidlistOverlaplist; // list of Overlap of each bid in the bidlist
For ( i=0; i < bidlist.size() ; i++ )
{
    Let, BidOverlap=0; // overlap between solution and Client Agent’s acceptable range
    For (k=iLR ; k < iUR +1 ; k++)
        { For (p=CLR; p < CUR+ 1; p++)
            { if (k=p) { MaxOverlap++;} }
        } bidlistOverlaplist[i]= BidOverlap ;
}
MaxOverlap=0 ; // to get highest value of BidOverlap across all bids
For (i=0; i < bidlist.size() ; i++ )
{
    If (bidlistOverlaplist[i] ≥ MaxOverlap) { MaxOverlap=bidlistOverlaplist[i];} }
// Now with MaxOverlap value known create a shortlist of all bids with MaxOverlap
For (i=0; i < bidlist.size() ; i++ )
{
    If (bidlistOverlaplist[i]=MaxOverlap) { Shortlist.add (bidlist[i]) ; }
```

102
For each bid in the shortlist

\[
\text{If } (cL_R \geq l_R) \{ \text{Lbuffer}=cL_R - l_R; \} \text{ Else } (\text{Lbuffer}=l_R - cL_R;)
\]

\[
\text{If } (u_R \geq cU_R) \{ \text{Ubuffer}=u_R - cU_R; \} \text{ Else } (\text{Ubuffer}=cU_R - u_R;)
\]

create list called FinalShortList

LeastTotalBuffer=Maximum range (default) // to get margin of solution space

For each bid in shortlist \{ currentBuffer =Lbuffer + Ubuffer ;\}

If (currentBuffer \leq LeastTotalBuffer){ LeastTotalBuffer=currentBuffer}

// Now to get a FinalShortlist of all bids that have the minimum buffer i.e., LeastTotalBuffer

For each bid in shortlist \{ If (currentBuffer=LeastTotalBuffer) { Add bid to FinalShortlist } \}

Select a random bid from FinalShortlist

**Figure 5.14: Pseudo code for bid selection in non-routine tasks**

The first step in evaluating the bids is to shortlist all the bids that have maximum overlap of the proposed solution space with the solution space defined by the Client Agent’s acceptable range of solutions. In the best-case scenario, there would be at least one bid that proposes the same upper and lower range of solutions as that of the Client Agent’s acceptable upper and lower range. The next best scenario would be to choose a bid from the short-listed bids so that the difference in solution space of the chosen bid and the solution space defined by the Client Agent’s acceptable range is least. Thus, in Figure 5.15, the bid with a solution space defined by the dark blue region is better than a bid with a solution space defined by the light blue region. Either of these bids is better than the bid represented by the small grey region, which has least overlap with the Client Agent’s acceptable range of solutions, represented by orange region.

**Figure 5.15: Bids received by Client Agent compared against the desired range**

For example, let the Client Agent’s acceptable range of solutions be defined by a lower value \(cL_R=2\) and an upper value \(cU_R=4\). For the best case scenario, it is desirable to have a bid \{\(bL_R=2, bU_R=4\}\}, where \(bL_R\) is the lower range, and \(bU_R\) is the upper range of the best bid. However, let two sample proposals received by the Client Agent be given by \{\(L_R=2, U_R=6\}\} and \{\(L_R=1, U_R=7\}\}. In this case, the proposal \{\(L_R=2, U_R=6\}\}, is a better match to Client Agent’s acceptable solution range.
than the proposal \( \{y_L = 1, y_U = 7\} \). While both the proposals overlap completely with the Client Agent’s acceptable range \( \{c_L = 2, c_U = 4\} \), there are fewer redundant solutions in the bid \( \{x_L = 2, x_U = 6\} \). In the Figure 5.14, LeastTotalBuffer is used to assess the redundancy. Once the final shortlist is created, a bid is chosen at random from the final shortlist.

### 5.7.2 Receipt of task completion information

Once the agent has accepted a bid, it allocates the task to the bidder, who becomes the lead agent. The team members coordinate the task and the sub-tasks among themselves. In case of the routine task, once the entire set of tasks is done, the Client Agent is informed of the task completion. In case of the non-routine tasks, the agents coordinating partial solutions directly report to the Client Agent about the completion of the partial solutions. Hence, the Client Agent has to ensure that all the partial solutions are received before informing the Simulation Controller that the project has been successfully completed.

Conceptually, this means that in case of the routine tasks, the Client Agent only has to allocate the task and wait for the confirmation that the task has been completed. However, when the task is non-routine, the Client Agent also initially approves a solution at the highest level. Once this higher level solution has been approved, it is the responsibility of the team leader to ensure that the resulting sub-tasks at lower levels are compatible. Similarly, the responsibility for coordinating solution integration applies to the other agents that coordinate the sub-solutions at lower levels. Thus, the Client Agent does not have to check the compatibility of each of the sub-solutions at lower levels. However, the Client Agent has to ensure that all the sub-solutions that are required at each level of task decomposition have been received. Once all the sub-solutions have been received, the Client Agent informs the Simulation Controller that the project has been successfully completed.

Figure 5.16 and Figure 5.17 show the activity diagrams of the Client Agent for the routine tasks and the non routine tasks, respectively.

---

![Activity diagram for the Client Agent (routine task)](image)

**Figure 5.16: Activity diagram for the Client Agent (routine task)**
5.8 Implementing the Simulation Controller

The Simulation Controller is a reactive agent that is required to: (1) start and monitor the simulations; (2) check the number of simulation runs; (3) switch between training rounds and test rounds of the simulation; and, (4) shut down the simulations based on the parameters set by the experimenter. Figure 5.18 shows the activity diagram of the Simulation Controller.
Figure 5.18: Activity diagram for simulation controller

5.9 Description of simulation lifecycle

Figure 5.19 shows the interaction protocol across the different agent types in the team during the entire simulation lifecycle. The simulation cycle has been described earlier in section 5.2. This section details the two types of simulations resulting from the two task types:

Case 1: routine task
The team members that can perform the “firstTask” bid to lead the task. Once the deadline for the task bid is over, the Client Agent chooses a random agent from the set of bidders as the lead agent and allocates the first task to the lead agent. In this case, all the task handling is sequential. Thus, after performing the first task, the lead agent allocates the resulting task to another agent that it expects can perform the next task. Based on whether it can perform the given task or not, the target agent (task receiver) either informs the source agent (task allocator) that the task is done, or communicates failure to perform the task by sending a refusal message. If the target agent refuses to perform the given task,
the source agent allocates the task to another agent, and the cycle continues until the task is performed. If the target agent is able to perform the task, it looks-up the next task in the sequence, and becomes a source agent for the next task, which it allocates to some other agent in the team. This cycle of task allocation continues until the entire set of tasks has been performed. The agent that performs the last task informs the Client Agent about the task completion.

**Case 2: non-routine task**

The team members that can perform the “firstTask” bid to lead the task. This bid proposal includes the range of solutions that the bidder can provide. Once the deadline for task bid is over, the Client Agent evaluates each of the proposals and shortlists the bids that are closest to its acceptable range of solutions. If more than one bid is shortlisted, a random proposal is selected and the task is allocated to the lead agent. The non-routine task needs to be decomposed and is performed top-down. Once the lead agent receives the task, it decomposes the task into sub-tasks, which it allocates to the other agents that it expects to be able to perform those tasks. Since the solutions for the decomposed tasks must be compatible, non-routine tasks require the source agents to coordinate the solutions. Target agents that cannot perform the given task send refusal messages, while agents that can perform the task communicate a proposed solution. Once the source agent has received the solutions for all the related sub-tasks, it checks the solutions for compatibility. The sub-tasks for which the solutions may not be compatible are sent for rework. The cycle of rework and task allocation continues unless the solutions for all the sub-tasks are approved. Once the solution for a sub-task is approved, the agent that performed the sub-task checks if the given sub-task needs to be decomposed further. If no decomposition is required, it informs the Client Agent that the sub-task is performed. If the task needs to be decomposed further, the same cycle of task allocation, coordination, and rework continues until all the sub-tasks are performed and the compatibility is ascertained.

In both the cases (routine tasks and non-routine tasks), once the Client Agent receives the notification of task completion, it passes on the information to the Simulation Controller. Depending upon the simulation status, the Simulation Controller either activates the next round (test round), or next run (new training round), or sends a request to the Agent Management System (default in JADE) to shut down the simulation platform.
Figure 5.19: Interaction protocol among all agent types during the simulation lifecycle
5.10 Computational model as the simulation environment

The proposed computational model incorporates the various parameters and variables related to the research hypotheses. The developed computational model is non-deterministic because the results from the simulation are not an artefact of the model. If the results were deterministic, then results from all similar simulations should be the same in each run. Thus, if the model were deterministic, it will not be a useful simulation model.

The developed simulation model is non-deterministic, which means the analysis of the results is based on means (average) of the results obtained from multiple simulation runs. The number of simulation runs required to obtain the results with an acceptable confidence level is achieved by conducting a split-half paired t-test on the results, as reported in section 6.1.2 and section 6.2.1.1. The simulation environment behaves as a non-deterministic system because of the following factors:

Likelihood of task allocation to an agent:
Agents allocate tasks to the agents that they identify (based on values in the TMM) to have the highest competence in the task. When more than one agent has the same (highest) competence value, agents select an agent based on random distribution to allocate the task. This scenario, i.e., when multiple agents have the same (highest) value for a given task, is common in the initial phases of the simulation, because the agents have no pre-developed TMM. The number of attempts an agent takes to allocate the task to the relevant expert determines the team performance. For example, let us take a scenario as shown in Figure 5.20. The team of twelve agents A\textsubscript{1} to A\textsubscript{12} need to complete a set of tasks that include tasks T\textsubscript{1} to T\textsubscript{3}, starting with T\textsubscript{1} in sequential order. Let the agent A\textsubscript{1} be the lead agent in the training round. Since A\textsubscript{1} has no pre-developed TMM, it allocates the task T\textsubscript{1} to a randomly selected agent in the hope of identifying a corresponding expert. A successful task allocation can take as few as 1 or as many as 11 attempts. This task allocation continues until A\textsubscript{1} successfully identifies the relevant expert, which is A\textsubscript{4} in this case. Now, A\textsubscript{4} needs to allocate the task T\textsubscript{2} to another agent, and it adopts a similar task allocation approach. This cycle of task allocation continues until all the tasks, i.e., T\textsubscript{1}, T\textsubscript{2} and T\textsubscript{3} have been successfully allocated. Once this has been done and the same team is retained for the test round such that A\textsubscript{1} is once again the lead agent, then all the unsuccessful task allocation attempts may be eliminated. In the test round, A\textsubscript{1} knows that A\textsubscript{4} can perform T\textsubscript{1}. Similarly, A\textsubscript{4} and A\textsubscript{9} know who to allocate the resulting next task. Thus, in this case, A\textsubscript{1}, A\textsubscript{4}, A\textsubscript{9} and A\textsubscript{12}, connected by solid arrows in Figure 5.20, form what we can define as the critical task network.

Thus, when the agents already have prior-acquaintance working with each other, the randomness is reduced. However, in such a scenario, the team performance depends upon whom the Client Agent
chooses as the lead agent. If the same agent is chosen as the lead agent in the test round as well the training round, then the lead agent is already a part of the critical task network. However, if $A^5$ is chosen as the lead agent in the test round, the advantages of the critical network developed in the training round are reduced, unless $A^5$ had observed $A^4$ performing $T^1$ in the training round.

![Critical network formed because of prior-acquaintance](image)

**Figure 5.20: Critical network formed because of prior-acquaintance**

**Selection of a sub-set of agents with prior acquaintance:**
Prior acquaintance reduces randomness in the task allocation by developing a critical network of agents that know who to pass the resulting next task. However, the efficiency of task allocation depends on the level of team familiarity. If team familiarity is low, and some of the agents who were part of the critical task network developed in the training round are no longer a part of the team in the test round, then the critical task network may break.

Depending on what node of the critical network is broken, the team performance will be affected differently. For example, in Figure 5.20, if agent $A^1$ is missing in the test round, the decrease in team performance is likely to be higher than the case where agent $A^1$ remains the lead agent and only the agent $A^{12}$ is missing from the critical network. Within the critical task network, the loss of an agent earlier in the sequence\(^\text{17}\) would mean that all the tasks from that node onwards might require multiple attempts to be allocated successfully.

The team composition is non-deterministic because the team familiarity is a percentage of the total team size, and it is possible that a different set of agents are chosen for each simulation.

\(^{17}\) For example, if the critical task network has 5 agents $A^1$ to $A^5$ in the same order of task allocation, then $A^1$ is earlier in the sequence (node position=1), compared to $A^3$ (node position=3) or $A^5$ (node position=5).
Likelihood that an agent is busy in given simulation cycle

Even if an agent is not a part of the critical network, it may have observed all the task allocations and interactions during the training round. This can significantly improve the team performance despite the broken critical networks. For example, in Figure 5.20, it is likely that in the training round, A\(^5\) observed A\(^4\) performing the task T\(^1\). Thus, in the test round, if A\(^5\) was chosen as the lead agent, then A\(^5\) could still allocate the task T\(^1\) to A\(^4\) because it already knows about A\(^4\)'s competence in T\(^1\). However, it is possible that A\(^5\) was busy and failed to observe A\(^4\) when it performed T\(^1\) or when A\(^1\) allocated T\(^1\) to A\(^4\). If that happens, A\(^5\) would need to search for an expert on T\(^1\) through random selection and allocation. The likelihood of busyness is defined by a probability factor, making it a non-deterministic event.

Selection of solution for non-routine tasks:

When the team is working on non-routine tasks, the agents have to propose solutions from a set of non-dominant alternatives. The choice of the solution is important because the solution should conform to the task allocator’s acceptable range. Agents look up their TMM and check the range of solutions that they expect to be acceptable to the task allocator. However, if the agents have no prior-experience of working with each other, they do not have the TMM developed to help in solution selection. In such a scenario, agents select a solution alternative at random. It may not be possible to determine the number of attempts at solutions proposed by an agent before it is accepted by the task allocator. Thus, in case of the non-routine tasks, the non-deterministic nature of solution selection and acceptance adds to the un-predictability of the team performance in a given simulation.
Chapter 6
Simulation Details and Results

Two kinds of experiments are conducted. The first set of experiments is conducted for model validation. The second set of experiments is conducted to test the research hypotheses discussed in Chapter 3.

6.1 Experiments to validate the computational model:

Experiments for model validation are designed such that:

1. The observed measures for team performance can be compared against the theoretically calculated measures. For these simulations, scenarios are chosen so that the values can be theoretically determined. If the observed values conform to the theoretical values, the consistency of the model is validated but not necessarily the simulation of a social behaviour.

2. The observed behaviour for the social (group) and individual learning cases can be compared against similar studies reported elsewhere (Moreland et al., 1998; Ren et al., 2006). These two studies compare the performance of the teams where members were trained individually against the teams where members were trained together as a group. While Moreland et al. (1998) report lab based studies with a team size of 3 members, Ren et al. (2001; 2006) conduct similar studies using a computational model, ORGMEM, with team sizes ranging from 3 to 35 members. If the social behaviours observed in the validation simulations resemble the social behaviours reported in the two case studies, then this model meets the criteria for Social Turing Test (Carley & Newell, 1994) (section 2.3).
### 6.1.1 Simulation set-up:

A routine design task with \(N^T\) discrete sub-tasks is introduced to a team of \(N^A\) agents such that \(N^A = gN^A \times g\), where \(N^A\) is the number of agents in the team, \(g\) is the number of equal-sized task-groups, each with \(gN^A\) agents. The number of tasks is represented as \(N^T = 1N^T + 2N^T + \ldots + nN^T\), where \(N^T\) is the total number of tasks in the team, and \(kN^T\) is the number of tasks to be performed by \(k\)th group. Expertise distribution is represented as \(N^T_N^p (N^A_N^p)\) such that there are \(N^T_N^p\) tasks for which there are \(N^A_N^p\) agents that can perform each of those tasks. For all the cases, \(N^T = \sum N^T_N^p\). However, \(N^A\) need not equal \(\sum (N^T_N^p \times N^A_N^p)\), i.e., there may be more agents than the number of tasks. For example, in Table 6.1, for a flat team with \(N^A = 15\) agents and \(N^T = 10\) tasks, the expertise distribution is given as \(9(1)1(5)\). In this case, for 9 of the 10 tasks, there is exactly 1 agent that can perform each of those tasks. However, for 1 of the 10 tasks, there are 5 agents that can perform the task.

For a team where the agents start with no prior knowledge of each other’s competence, the initial sub-task allocations are random, until an agent with the corresponding task expertise is identified. Each time a sub-task is allocated, two messages (“call for proposal” and feedback) are exchanged. Thus, for a team with \(N^A\) agents, \(N^T\) tasks, and \(N^P\) agents per task, the theoretical upper limit (\(L_{\text{max}}\)) of the number of messages exchanged before the task is complete is \((N^A - N^P + 1) \times N^T \times 2\).

Hence, the calculated upper limit, \(L_{\text{max-cal}}\) for a team with given expertise distribution is

\[
L_{\text{max-cal}} = 2 \times \sum kN^T \left(kN^A - kN^P + 1\right).
\]

The simulations reported in Table 6.2 are conducted with two kinds of R-agents. In any given simulation, only one kind of agents is used at a time. The difference in the agents is entirely based on their learning capabilities. The agents of type \(A^{R1}\) learn only from their personal interactions. These agents are not able to observe the other data from the environment. The agents of type \(A^{R2}\) learn from personal interactions, task observations, and interaction observations.

Table 6.1 summarizes the number of messages exchanged in the simulations. \(O\) is the maximum value for the number of messages observed. Table 6.2 summarizes the simulation results for the level of TMM formation. Two types of team structures are used: (1) flat teams; and (2) teams organized into task-based sub-groups.

### 6.1.2 Calculating the value of TMM formed

All the agents start with a default value of the TMM. As the agents learn about the competence of themselves and the other agents, the values for corresponding elements in the TMM are updated. The value of the level of TMM formation is the percentage of the default values that have been replaced by the learnt competence values at the end of the simulation. The overall value of the level of TMM formation is the average of the values of TMM formation for all the individual team members (section
5.3.4). SD is the standard deviation across the value of TMM formation among the agents, evaluated across the entire team.

### Table 6.1: Summary of the number of messages exchanged in training set

<table>
<thead>
<tr>
<th>Agent</th>
<th>N^A</th>
<th>N^T</th>
<th>N^T1 (N^P1) N^T2 (N^P2)</th>
<th>Runs</th>
<th>L_{max-cal}</th>
<th>O</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR^1</td>
<td>6</td>
<td>4</td>
<td>4(1)</td>
<td>30</td>
<td>48</td>
<td>26</td>
</tr>
<tr>
<td>AR^1</td>
<td>6</td>
<td>4</td>
<td>1(3) 3(2)</td>
<td>30</td>
<td>38</td>
<td>20</td>
</tr>
<tr>
<td>AR^1</td>
<td>15</td>
<td>10</td>
<td>9(1)1(5)</td>
<td>30</td>
<td>292</td>
<td>229</td>
</tr>
<tr>
<td>AR^1</td>
<td>15=5×3</td>
<td>10 =4+3+3</td>
<td>4(1)3(1)3(1)</td>
<td>30</td>
<td>100</td>
<td>73</td>
</tr>
</tbody>
</table>

### Table 6.2: Summary of the TMM formation after training (60 runs)^18

<table>
<thead>
<tr>
<th>Agent</th>
<th>Type</th>
<th>N^A</th>
<th>N^T</th>
<th>N^T1 (N^P1) N^T2 (N^P2)</th>
<th>TMM (%)</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR^1</td>
<td>Flat</td>
<td>6</td>
<td>4</td>
<td>4(1)</td>
<td>21.56</td>
<td>4.90</td>
</tr>
<tr>
<td>AR^2</td>
<td>Flat</td>
<td>6</td>
<td>4</td>
<td>4(1)</td>
<td>56.69</td>
<td>9.05</td>
</tr>
<tr>
<td>AR^1</td>
<td>Flat</td>
<td>15</td>
<td>10</td>
<td>9(1)1(5)</td>
<td>8.97</td>
<td>1.76</td>
</tr>
<tr>
<td>AR^2</td>
<td>Flat</td>
<td>15</td>
<td>10</td>
<td>9(1)1(5)</td>
<td>50.98</td>
<td>8.17</td>
</tr>
</tbody>
</table>

### Table 6.3: Summary of the number of messages exchanged in test set (60 runs)\(^19\)

<table>
<thead>
<tr>
<th>Agent</th>
<th>Type</th>
<th>N^A</th>
<th>N^T</th>
<th>N^T1 (N^P1) N^T2 (N^P2)</th>
<th>Avg. no. of messages</th>
<th>SD</th>
</tr>
</thead>
<tbody>
<tr>
<td>AR^1</td>
<td>Flat</td>
<td>6</td>
<td>4</td>
<td>4(1)</td>
<td>12.60</td>
<td>2.80</td>
</tr>
<tr>
<td>AR^2</td>
<td>Flat</td>
<td>6</td>
<td>4</td>
<td>4(1)</td>
<td>11</td>
<td>0</td>
</tr>
<tr>
<td>AR^1</td>
<td>Flat</td>
<td>12</td>
<td>7</td>
<td>6(2) 1(1)</td>
<td>20.70</td>
<td>5.23</td>
</tr>
<tr>
<td>AR^2</td>
<td>Flat</td>
<td>12</td>
<td>7</td>
<td>6(2) 1(1)</td>
<td>16</td>
<td>0</td>
</tr>
</tbody>
</table>

### 6.1.3 Discussion of simulation results:

The number of messages observed (O) in all the test cases is below the L_{max-cal} (Table 6.1), providing the preliminary validation of the consistency of the implemented model.

---

18 A split-half t-test for two sample mean on all data shows a confidence level > 99% (Alpha=0.01 for all cases). Individual results: TMM=21.56 [t-value=-0.943, P(T<=t)=0.353]; TMM=56.69 [t-value=-0.881, P(T<=t)=0.386]; TMM=8.97 [t-value=0.558, P(T<=t)=0.581]; TMM=50.98 [t-value=0.981, P(T<=t)=0.334].

19 A split-half t-test for two sample mean on all data shows a confidence level > 99% (Alpha=0.01 for all cases). Individual results: Messages=12.60 [t-value=-0.779, P(T<=t)=0.442]; Messages=11 [SD=0]; Messages=20.70 [t-value=0.810, P(T<=t)=0.424]; Messages=16 [SD =0]. The null hypothesis in these t-tests assumed that the split-half samples belong to same data sets. Results support the null hypothesis in each case, suggesting that 60 simulation runs are enough to get a confidence level > 99% in the results.

In Table 6.3, SD=0 for AR^2 because the task is routine, TF=100% and BL=0%. Thus, while the task can be performed with certainty, social learning allows all the agents to identify the relevant experts in the training rounds. The simulations where task=routine, LM=PI+TO+IO, TF=100% and BL=0% is a special case in which even though TMM varies, the team performance appears deterministic. Team performance in all other experiment conditions is non-deterministic, as discussed in section 5.10 and observed in Table 6.12 and Table 6.13.
The simulations with the two types of agents, namely \( A^{R1} \) and \( A^{R2} \), correspond to the studies on individual training and group training of the team members, as reported by Ren et al. (2006) and Moreland et al. (1998). Group training involves personal interactions, communication and observations. This matches the case where the agents have all learning modes available to them (\( A^{R2} \)). The simulations where the agents can only learn from personal interaction (\( A^{R1} \)) are similar to the individual training case.

The measure of TMM formation adopted for these simulations are similar to the measures reported in the two studies, i.e., Ren et al. (2006) and Moreland et al. (1998). Both these studies (Moreland et al., 1998; Ren et al., 2006) calculate the density and accuracy of TMM formation. Density measures “how much of the TMM is learnt”. Accuracy measures “how much of what is learnt is correct”. In this thesis, accuracy need not be measured because whatever agents learn is accurate. Hence, density (amount) is the only measurement required.

The measures for team performance used in the two studies include the time taken to perform the task, and the quality of output. The quality of output is not assessed in this thesis, because none of the acceptable solutions are dominant. The team performance is measured in terms of ‘time’ i.e., the amount of communication required, calculated as the number of messages exchanged.

The teams in which the agents can learn from social observations, in addition to their personal interactions, have significantly higher level of TMM formation, Table 6.4\(^{20}\). These results conform to the findings reported in the two cases studies. The two cases studies (Moreland et al., 1998; Ren et al., 2006) also reported positive effects of group training on the team performance. The findings from the validation simulations are similar (Table 6.3).

Experiments measuring the team performance were conducted with different team sizes (6 and 12 members). In both the cases, the teams of \( A^{R2} \) agents performed significantly better than the teams of \( A^{R1} \) agents, Table 6.4. The difference in the performance of the differently trained teams increases with the increase in team size, Table 6.4, i.e., social learning has greater effect on team performance as the team size increases.

The increase in the team size significantly reduces the level of TMM formation, Table 6.5. The effects of team size on TMM formation is higher in the teams where only individual learning is available to the agents (t-value=−19.76). The effects of team size on TMM formation is lower if social learning opportunities are available to the agents (t-value=−3.92).

---

\(^{20}\) Results obtained from experiments with \( A^{R1} \) agents are compared against the results obtained from experiments with \( A^{R2} \) agents. The two sample t-tests reject the null hypothesis that there is no difference in the mean of the results from the experiments with the \( A^{R1} \) agents and the \( A^{R2} \) agents. This shows that social learning enhances TMM formation. When the team size=6, the difference in means of results obtained from experiments with the \( A^{R1} \) agents and the \( A^{R2} \) agents shows a t-value=27.48. However, when team size=15, the corresponding t-value increase to 39.20, suggesting that the team size also has an effect on TMM formation.
Moreland et al. (1998) and Ren et al. (2006) report that when the team size is 3, the difference in the performance between the teams with group training and individual training is significant in terms of the quality but not in terms of time. However, a similar study by Ren et al. (2001), with larger team sizes, shows significant effect on the team performance, even in terms of time. Thus, the earlier studies validate the observed effects of team size and group training on TMM formation and the team performance.

Table 6.4: Difference in effects of social learning ($A^{R2}$) and individual learning ($A^{R1}$)

<table>
<thead>
<tr>
<th>Variable</th>
<th>Team Size</th>
<th>N</th>
<th>df</th>
<th>Mean ($A^{R2} / A^{R1}$)</th>
<th>t-value</th>
<th>P-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TMM (%)</td>
<td>6</td>
<td>60</td>
<td>59</td>
<td>56.69 / 21.56</td>
<td>27.48</td>
<td>&lt;0.001</td>
</tr>
<tr>
<td>Messages</td>
<td>6</td>
<td>60</td>
<td>59</td>
<td>11.00 / 12.60</td>
<td>-4.43</td>
<td>&lt;0.001</td>
</tr>
<tr>
<td>TMM (%)</td>
<td>15</td>
<td>60</td>
<td>59</td>
<td>50.98 / 8.97</td>
<td>39.20</td>
<td>&lt;0.001</td>
</tr>
<tr>
<td>Messages</td>
<td>12</td>
<td>60</td>
<td>59</td>
<td>16.00 / 20.70</td>
<td>-6.96</td>
<td>&lt;0.001</td>
</tr>
</tbody>
</table>

Table 6.5: Difference in effects of team size across agents with social ($A^{R2}$) and individual ($A^{R1}$) learning

<table>
<thead>
<tr>
<th>Variable</th>
<th>Agent type</th>
<th>Team sizes (X/ Y)</th>
<th>N</th>
<th>df</th>
<th>Mean (X/ Y)</th>
<th>t-value</th>
<th>P-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>TMM (%)</td>
<td>$A^{R1}$</td>
<td>(15 / 6)</td>
<td>60</td>
<td>59</td>
<td>8.97 / 21.56</td>
<td>-19.76</td>
<td>&lt;0.001</td>
</tr>
<tr>
<td>Messages</td>
<td>$A^{R1}$</td>
<td>(12 / 6)</td>
<td>60</td>
<td>59</td>
<td>20.70 / 12.60</td>
<td>11.64</td>
<td>&lt;0.001</td>
</tr>
<tr>
<td>TMM (%)</td>
<td>$A^{R2}$</td>
<td>(15 / 6)</td>
<td>60</td>
<td>59</td>
<td>50.98 / 56.69</td>
<td>-3.92</td>
<td>&lt;0.001</td>
</tr>
<tr>
<td>Messages</td>
<td>$A^{R2}$</td>
<td>(12 / 6)</td>
<td>60</td>
<td>59</td>
<td>16.00 / 11.00</td>
<td>-</td>
<td>-</td>
</tr>
</tbody>
</table>

Compared to ORGMEM, the computational model used by Ren et al. (2001; 2006), the computational model developed in this thesis has fewer assumptions and simpler agents. However, the similarities in the observed social behaviour validate the usability of this model for the nature of study being conducted in this thesis. This equivalency test of the results from the developed model with the results from another computational model (ORGMEM) also provides docking (Axtell et al., 1996) for the developed model.

6.2 Experiments designed to test the research hypotheses

The experiment scenarios required to test the research hypotheses are generated by the combination of the different social learning modes, busyness levels, levels of team familiarity, team structures, and the task types. Table 6.6 summarizes the list of experiments.
Agent’s social learning capabilities can be set to four different levels:
1. Learning only from personal interactions (PI)
2. Learning by personal interaction as well by observing the interaction among the agents in the team, i.e., (PI+IO)
3. Learning by personal interaction as well as by observing the other agents perform a task, i.e., (PI+TO), and
4. Learning by personal interaction as well as by observing the task performances and the interactions of the other agents, i.e., (PI+IO+TO)

Three different types of team structures are used:
1. Flat teams
2. Flat teams with social cliques
3. Task-based sub-teams

Two types of task types are used:
1. Routine tasks
2. Non-routine tasks

Computationally, busyness level can have any value between 0 and 100%. In the reported experiments, up to six different values of busyness level are used (0, 25, 33, 50, 66, and 75%).

The team familiarity values also range between 0 and 100%. However, these values are inversely related to the team size, i.e., for a team of size \( n \), the level of team familiarity must be a multiple of \( \frac{1}{n} \).

Six different values are used for the level of team familiarity (17, 33, 50, 66, 83 and 100%). All the experiments for team familiarity are conducted with the team size=12.

Thus, there are 288 different experiments conducted using the different learning modes, levels of team familiarity, busyness levels, team structures and the task types, Table 6.6.

Table 6.6: Experiment matrix showing the combination of parameters used in different simulations

<table>
<thead>
<tr>
<th>Simulations</th>
<th>BL</th>
<th>TF</th>
<th>Team structure (TS)</th>
<th>Task type</th>
<th>No. of experiments</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Flat</td>
<td>Social cliques</td>
<td>Task-based sub-teams</td>
</tr>
</tbody>
</table>

Experiments with routine tasks and busyness

<table>
<thead>
<tr>
<th>Simulations</th>
<th>BL</th>
<th>TF</th>
<th>Team structure (TS)</th>
<th>Task type</th>
<th>No. of experiments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Personal Interaction (PI)</td>
<td></td>
<td></td>
<td>Flat</td>
<td>Social cliques</td>
<td>Task-based sub-teams</td>
</tr>
<tr>
<td>PI + Interaction</td>
<td></td>
<td></td>
<td>Flat</td>
<td>Social cliques</td>
<td>Task-based sub-teams</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Simulations</th>
<th>BL</th>
<th>TF</th>
<th>Team structure (TS)</th>
<th>Task type</th>
<th>No. of experiments</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Flat</td>
<td>Social cliques</td>
<td>Task-based sub-teams</td>
</tr>
</tbody>
</table>

Using the table, it can be seen that the experiments conducted with routine tasks and busyness are: 6 × 3 = 18 experiments.

Thus, the total number of experiments conducted with different parameters is: BL × TS = 288.
<table>
<thead>
<tr>
<th></th>
<th>PI</th>
<th>PI+IO</th>
<th>PI+TO</th>
<th>PI+IO+TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>observation (IO)</td>
<td>√</td>
<td>-</td>
<td>-</td>
<td>√</td>
</tr>
<tr>
<td>PI + Task observation (TO)</td>
<td>-</td>
<td>√</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PI+IO+TO</td>
<td>√</td>
<td>-</td>
<td>√</td>
<td>-</td>
</tr>
</tbody>
</table>

**Experiments with routine tasks and team familiarity**

<table>
<thead>
<tr>
<th>PI</th>
<th>PI+IO</th>
<th>PI+TO</th>
<th>PI+IO+TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>√</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>√</td>
<td>-</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>√</td>
</tr>
<tr>
<td>6×3=18</td>
<td>TF×TS</td>
<td>6×3=18</td>
<td>TF×TS</td>
</tr>
</tbody>
</table>

**Experiments with non-routine tasks and busyness**

<table>
<thead>
<tr>
<th>PI</th>
<th>PI+IO</th>
<th>PI+TO</th>
<th>PI+IO+TO</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>√</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>√</td>
<td>-</td>
</tr>
<tr>
<td>-</td>
<td>-</td>
<td>-</td>
<td>√</td>
</tr>
<tr>
<td>6×3=18</td>
<td>BL×TS</td>
<td>6×3=18</td>
<td>BL×TS</td>
</tr>
</tbody>
</table>

**Experiments with non-routine tasks, team familiarity and busyness**

<table>
<thead>
<tr>
<th>PI</th>
<th>PI+IO</th>
</tr>
</thead>
<tbody>
<tr>
<td>-</td>
<td>√</td>
</tr>
<tr>
<td>-</td>
<td>√</td>
</tr>
<tr>
<td>6×3=18</td>
<td>TF×TS</td>
</tr>
</tbody>
</table>
Experiments with routine tasks, busyness and team familiarity

In general, each experiment is conducted with 60 simulation runs. A split-half t-test for two sample mean conducted for all the simulation data gives a confidence level greater than 95% (Alpha=0.05, t-value < t-critical, P(T<=t) > 0.25), supporting the null hypothesis that the two split-half samples are from the same data sets. Some of the experiments are conducted with 120 simulation runs, but the t-values and P-values obtained from 60 simulations and 120 simulation runs are similar. Some of the experiments with busyness levels gave similar confidence levels with 30 simulation runs.

6.2.1 Details of experiments conducted:

6.2.1.1 Experiments with routine tasks and busyness
The experiments with the routine tasks are conducted with two different team sizes, Table 6.7. All the experiments with routine tasks are conducted with the R-Agents. The experiments with routine tasks and busyness are conducted with the team size=15 (Set 1, Table 6.7). Only the level of TMM formation is measured in these experiments. Accordingly, only the training rounds are required in these simulations, and team familiarity is not a variable. These experiments are used to establish the different correlations between the learning modes, busyness level, team structures and the level of TMM formation. Each agent has competence in only one task. However, for some tasks, more than one agent has competence in the same task. In any simulation, each agent is affiliated to only one group. If the team is flat in the given simulation, all the agents in the team are part of the same group (Grp_1). Within their own group, agents can observe the task performance or the interactions of any other agent. In flat teams, the agents can also allocate the task to any other agent in the team.

Table 6.7: Team compositions used for simulations with the routine tasks

<table>
<thead>
<tr>
<th>Agent ID</th>
<th>Known task</th>
<th>Sub-teams/ Social</th>
<th>Flat Teams</th>
<th>Agent ID</th>
<th>Known Task</th>
<th>Sub-teams/ Social</th>
<th>Flat Teams</th>
</tr>
</thead>
<tbody>
<tr>
<td>Set 1</td>
<td></td>
<td></td>
<td></td>
<td>Set 2</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
If the team is flat but divided into sub-groups, the affiliation is assigned to the agent as shown in the third column of Set 1, Table 6.7. Within their own group, agents can observe the task performance or interactions of any other agent. Since the team is flat for the purpose of task-allocation, agents can allocate the task to any other agent in the team.

If the simulated team is organized into task-based sub-groups, then the same group distribution is used for the agent affiliation, i.e., as listed in the third column of Set 1, Table 6.7. However, in these simulations the agent affiliation constrains the task allocation. For example, if the task 1_c is to be performed, then the agents search for an expert only within Grp_1. Even though there are agents in Grp_2 and Grp_3 that can perform the task 1_c, the task is never allocated to them. Hence, if an agent’s expertise is not relevant to the task-group it is affiliated to, then that expertise may be redundant for the team.
6.2.1.2 Experiments with routine tasks and team familiarity

The experiments with the routine tasks and team familiarity are conducted with team size=12 (Set 2, Table 6.7). In these experiments, both the team performance and the level of TMM formation are measured. These simulations involve both the training rounds as well as the test rounds. The task used in the training round is repeated in the test round. However, differences in performance are expected because the team leader is selected by the Client Agent in the test round, independent of the selection made in the training round. Accordingly, more than one agent has competence in the first task (1_a). In addition, in the experiments with team familiarity, some of the agents from the training round are replaced by new agents. This increases the likelihood of breaking the critical task network, discussed in section 5.10.

6.2.1.3 Experiments with non-routine tasks and busyness

The experiments with the non-routine tasks and team familiarity are conducted with team size=12 (Table 6.8). All the agents used in these experiments are NR-Agents. Each agent has expertise in more than one task. For each task, each agent has a pre-defined range of solutions that it can provide. Similar to the experiments with the routine tasks, each agent is affiliated to only one group.

In the simulations with non-routine tasks, the tasks used are quasi-repetitive, i.e., while the task remains the same, the specifications for the desired solution range are different in the training rounds and the test rounds. In the training rounds, the desired range for overall solution is [3 4] and in the test rounds the desired solution range is [4 5]. For the non-routine tasks, the sub-solutions proposed by agents should be such that the overall solution falls within the desired range. Hence, the change in the task specifications are expected to result in differences in the team performance, besides the factors such as level of team familiarity and the selection of team leader, as discussed in section 6.2.1.1.

Table 6.8: Team compositions used for simulations with non-routine tasks

<table>
<thead>
<tr>
<th>Agent ID</th>
<th>Known task and range</th>
<th>Sub-teams/ Social Clique</th>
<th>Flat Teams</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bs0</td>
<td>Tm [3 4], Tmb, b [3 8]</td>
<td>Grp_b</td>
<td>Grp_b</td>
</tr>
<tr>
<td>Bs1</td>
<td>Tm [3 5], Tmb, b [3 8]</td>
<td>Grp_b</td>
<td>Grp_b</td>
</tr>
<tr>
<td>Bs2</td>
<td>Tm [3 5], Tmb, b [3 8]</td>
<td>Grp_b</td>
<td>Grp_b</td>
</tr>
<tr>
<td>Bs3</td>
<td>Tm, a [3 7], Tm, c [3 8]</td>
<td>Grp_c</td>
<td>Grp_b</td>
</tr>
<tr>
<td>Bs4</td>
<td>Tm, a [3 7], Tm, c [3 8]</td>
<td>Grp_a</td>
<td>Grp_b</td>
</tr>
<tr>
<td>Bs5</td>
<td>Tma, a [2 6], Tma, c [2 9]</td>
<td>Grp_a</td>
<td>Grp_b</td>
</tr>
<tr>
<td>Bs6</td>
<td>Tma, a [2 6], Tma, c [2 9]</td>
<td>Grp_a</td>
<td>Grp_b</td>
</tr>
<tr>
<td>Bs7</td>
<td>Tmb, a [2 6], Tmb, c [2 9]</td>
<td>Grp_b</td>
<td>Grp_b</td>
</tr>
</tbody>
</table>
Bs8 $T^{mb,a}$ [2 6], $T^{mb,c}$ [2 9], Grp_b Grp_b
Bs9 $T^{mc,a}$ [2 6], $T^{mc,c}$ [2 9], Grp_c Grp_b
Bs10 $T^{ma,b}$ [2 9], $T^{mb,b}$ [2 9], Grp_a Grp_b
Bs11 $T^{ma,b}$ [2 9], $T^{mc,b}$ [2 9], Grp_c Grp_b
Client [3 4] in training round
[4 5] in test round

<table>
<thead>
<tr>
<th>Summary of experiments:</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of agents</td>
</tr>
<tr>
<td>Total number of tasks</td>
</tr>
<tr>
<td>Tasks needing coordination</td>
</tr>
<tr>
<td>Expertise distribution</td>
</tr>
<tr>
<td>Groups</td>
</tr>
</tbody>
</table>

Figure 6.1 shows the hierarchy of the tasks listed in Table 6.8. The task assigned to the agent needs to be decomposed and integrated. Thus, four coordination tasks (grey nodes in Figure 6.1) are generated to manage the solution integration. In simulations with the non-routine tasks, the agents are required to evaluate the received solutions, as discussed in section 4.1.6.2.

6.2.1.4 Experiments with team familiarity and busyness
The experiments with team familiarity and busyness are conducted with routine tasks and team size=12 (set 2, Table 6.7). The R-Agents are used in these simulations, and the agents have all the modes of social learning available to them.
6.2.2 Simulation results

This section summarizes the data obtained from the simulations. A brief discussion of the results is presented. Detailed discussion on the experiment results, with respect to the research hypotheses, is presented in Chapter 7.

6.2.2.1 Experiments with routine tasks and busyness level

Table 6.9 and Table 6.10 summarize the data obtained from the simulations with 15 R-Agents (Set 1, Table 6.7) and 12 R-Agents (Set 2, Table 6.7), respectively. Since busyness is defined in terms of the agents’ attention to social observations (task or interaction), busyness has no influence on the agents that can learn only from personal interactions. Hence, only one set of experiments are conducted with these agents to determine the contribution of personal learning in the amount of TMM formation. This also corresponds to the cases with 100% busyness level for other agents that can also learn from social observations. Even if an agent can learn from the social observations, at 100% busyness level, it does not attend to any of the observable data. There is no difference in flat teams and flat teams with social cliques for the agents that can learn only from personal interactions because these two team types are differentiated only in terms of the opportunities for social observations.

<table>
<thead>
<tr>
<th>Learning modes</th>
<th>Busyness level %</th>
<th>Flat Teams</th>
<th>Social Cliques</th>
<th>Task-based sub-teams</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>% TMM formation</td>
<td>Std Dev</td>
<td>% TMM formation</td>
</tr>
<tr>
<td>PI</td>
<td></td>
<td>9.08</td>
<td>1.66</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>17.36</td>
<td>0.96</td>
<td>16.20</td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>16.62</td>
<td>0.80</td>
<td>15.40</td>
</tr>
<tr>
<td></td>
<td>33</td>
<td>17.01</td>
<td>0.96</td>
<td>14.92</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>16.61</td>
<td>0.95</td>
<td>15.44</td>
</tr>
<tr>
<td></td>
<td>66</td>
<td>16.36</td>
<td>1.51</td>
<td>15.04</td>
</tr>
<tr>
<td></td>
<td>75</td>
<td>16.56</td>
<td>1.25</td>
<td>14.77</td>
</tr>
<tr>
<td>PI+IO</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>0</td>
<td>46.00</td>
<td>3.60</td>
<td>18.86</td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>41.23</td>
<td>4.27</td>
<td>19.66</td>
</tr>
<tr>
<td></td>
<td>33</td>
<td>38.89</td>
<td>3.87</td>
<td>17.97</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>34.21</td>
<td>4.46</td>
<td>16.92</td>
</tr>
<tr>
<td></td>
<td>66</td>
<td>32.59</td>
<td>4.61</td>
<td>17.27</td>
</tr>
<tr>
<td>Learning modes</td>
<td>Busyness level %</td>
<td>Flat Teams</td>
<td>Social Cliques</td>
<td>Task-based sub-teams</td>
</tr>
<tr>
<td>----------------</td>
<td>-----------------</td>
<td>------------</td>
<td>----------------</td>
<td>---------------------</td>
</tr>
<tr>
<td></td>
<td>% TMM formation</td>
<td>Std Dev</td>
<td>% TMM formation</td>
<td>Std Dev</td>
</tr>
<tr>
<td>PI</td>
<td>7.73</td>
<td>1.98</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>PI+IO</td>
<td>0</td>
<td>18.42</td>
<td>1.20</td>
<td>15.41</td>
</tr>
<tr>
<td>PI+TO</td>
<td>25</td>
<td>18.52</td>
<td>1.20</td>
<td>16.23</td>
</tr>
<tr>
<td></td>
<td>33</td>
<td>18.17</td>
<td>1.37</td>
<td>15.02</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>17.95</td>
<td>1.62</td>
<td>15.45</td>
</tr>
<tr>
<td></td>
<td>66</td>
<td>17.76</td>
<td>1.76</td>
<td>14.36</td>
</tr>
<tr>
<td></td>
<td>75</td>
<td>13.74</td>
<td>2.34</td>
<td>13.84</td>
</tr>
<tr>
<td>PI+IO+TO</td>
<td>0</td>
<td>37.42</td>
<td>7.59</td>
<td>17.84</td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>33.00</td>
<td>9.01</td>
<td>15.88</td>
</tr>
<tr>
<td></td>
<td>33</td>
<td>31.96</td>
<td>8.51</td>
<td>15.95</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>29.24</td>
<td>6.53</td>
<td>14.30</td>
</tr>
<tr>
<td></td>
<td>66</td>
<td>29.48</td>
<td>5.16</td>
<td>13.90</td>
</tr>
<tr>
<td></td>
<td>75</td>
<td>27.28</td>
<td>6.86</td>
<td>13.99</td>
</tr>
<tr>
<td>PI+IO+TO</td>
<td>0</td>
<td>41.33</td>
<td>6.78</td>
<td>21.33</td>
</tr>
<tr>
<td></td>
<td>25</td>
<td>36.53</td>
<td>7.04</td>
<td>18.77</td>
</tr>
<tr>
<td></td>
<td>33</td>
<td>35.63</td>
<td>6.54</td>
<td>19.29</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>35.77</td>
<td>6.89</td>
<td>18.35</td>
</tr>
<tr>
<td></td>
<td>66</td>
<td>31.72</td>
<td>5.78</td>
<td>16.64</td>
</tr>
<tr>
<td></td>
<td>75</td>
<td>30.03</td>
<td>4.81</td>
<td>14.25</td>
</tr>
</tbody>
</table>
Learning modes and level of TMM formation

Learning from social observations improves the amount of TMM formation. For example, in flat teams when the agents learn only from personal interactions, TMM formation is 9.08% (SD=1.66). If the agents have all modes of learning available to them, TMM formation increases to 50.98% (SD=8.33).

A comparison of the level of TMM formation in the flat teams at busyness level =0 shows that TMM formation reduces when the agents are not able to observe task performance. The level of TMM formation also reduces when the agents are not able to observe interactions. However, when the task observation is not available, i.e., PI+IO, the reduction in TMM formation is higher as compared to the case when the interaction observation is not available, i.e., PI+TO, (Table 6.9 and Table 6.10). The difference in the contribution of the task observations and the interaction observations can be explained in terms of the modelling assumptions. When an agent learns from interaction observations, it only observes the details of the task allocator. The details of whether the task receiver performed the task or refused to perform the task are not available (see Section 5.6, Table 5.4). In this case, the agent (observer) infers that the task allocator cannot perform the task that it is allocating. The agent also infers that the task allocator can perform the task that precedes the allocated task (see Table 5.1). Hence, in interaction observations, only two values per task are updated in the TMM, both related to the task allocator. The interactions in this model correspond to the task allocation but not to the response to the allocated task. The later is considered as part of task observations. When an agent observes task performance, but not the interaction (or task allocation), it knows whether an agent has performed the given task or not. Hence, for each agent that has been allocated the task, the related values in the TMM are updated. Half of the values that are updated based on the interaction observation (i.e., related to the performance of preceding task) are always updated with the task observations at some previous simulation cycle, provided the observer agent was not busy at that instance. The contribution of the task observations is higher than the interaction observations because only formal, task-related, interactions are considered in these simulations.

The levels of TMM formation at the different busyness levels for each case of learning modes are compared. The results show that the reduction in TMM formation with the increase in busyness is higher for the task observations as compared to the interaction observations. The differences in the TMM formation, at higher and lower busyness levels, is higher for the agents learning from personal interactions and task observations (PI+TO) than that for the agents learning from personal interactions and interaction observations (PI+IO), Table 6.9 and Table 6.10.
**Team structure and level of TMM formation**

For all the simulations, irrespective of the agents’ learning abilities, TMM formation is highest for flat teams, and lowest for the teams organized into task-based sub-teams (Table 6.9 and Table 6.10). This result is expected. In flat teams, each agent has more agents to interact with and observe than either flat team with social cliques, or the teams organized into task-based sub-teams. The different team structures can be argued to result in differences in the effective team size. For the same number of agents, the flat teams have an effectively larger team size than the teams organized into sub-teams. Based on these explanations, the results adhere to the previously reported findings on the effects of team size on the level of TMM formation (Ren et al., 2001). However, based on the same arguments, the differences in the level of TMM formation across the different team structures should increase as the team size increases. The simulation results support this conjecture. For example, when the team size=15 (Table 6.9) and the learning mode=PI+TO+IO, the values of TMM formation for flat teams, flat teams with social cliques, and the teams organized as sub-teams are 50.98% (SD=8.33), 24.30% (SD=3.56) and 7.96% (SD=0.70), respectively. When the team size=12 (Table 6.10), the corresponding values are 41.33% (SD=6.78), 21.33% (SD=3.83) and 8.11% (SD=0.73), respectively.

Therefore, with the increase in team size, the differences in TMM formation across the different team structures increase. TMM formation is highest for flat teams, lower in flat teams distributed into social cliques, and lowest when the teams are organized as task-based sub-teams.

**6.2.2.2 Experiments with non-routine tasks and busyness level**

Table 6.11 summarizes the data obtained from the simulations with 12 NR-Agents (Table 6.8). These results show similar patterns to the experiments with the routine tasks. The TMM formation is higher in flat teams, lower in flat teams distributed into social cliques, and lowest in the teams organized as sub-teams. When task observations is absent, i.e., PI+IO, the TMM formation is lower compared to the TMM formation when interaction observations is absent, i.e., PI+TO, Table 6.11.

<table>
<thead>
<tr>
<th>Learning modes</th>
<th>Flat Teams</th>
<th>Social Cliques</th>
<th>Task-based sub-teams</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>% TMM formation</td>
<td>Std Dev</td>
</tr>
<tr>
<td>Busyness level %</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Personal Interaction</td>
<td>11.95</td>
<td>1.96</td>
<td>-</td>
</tr>
<tr>
<td>0</td>
<td>15.52</td>
<td>0.93</td>
<td>14.38</td>
</tr>
<tr>
<td>25</td>
<td>15.14</td>
<td>0.80</td>
<td>13.97</td>
</tr>
<tr>
<td>33</td>
<td>15.56</td>
<td>1.68</td>
<td>13.94</td>
</tr>
</tbody>
</table>
Table 6.12 summarizes the results of the simulations with the routine tasks and 12 R-Agents. Team familiarity=0% means all the agents in the test round are new. Hence, the team performance in that case is not affected by the agents’ learning modes.

Higher team performance is indicated by lower number of messages. The results show that the team performance increases with increase in team familiarity. When the agents learn from all modes of social learning (PI+IO+TO), in all the simulations with team familiarity=100%, the team shows optimal performance, as reflected in standard deviation=0. Optimal performance is also achieved when the agents learn from task observation, in addition to personal interaction (PI+TO). However, when the agents do not learn from task observation (PI+IO), the team does not consistently achieve optimal performance, even at team familiarity=100%. The pattern of increase in the team performance with increase in team familiarity varies with the learning modes.

6.2.2.3 Experiments with routine tasks and team familiarity

Table 6.12: Experiments with routine tasks and team familiarity (Set 2, Table 6.7)
<table>
<thead>
<tr>
<th></th>
<th>17</th>
<th>33</th>
<th>50</th>
<th>66</th>
<th>83</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>Personal</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interaction</td>
<td>53.63</td>
<td>53.20</td>
<td>51.47</td>
<td>48.53</td>
<td>32.80</td>
<td>20.67</td>
</tr>
<tr>
<td></td>
<td>15.21</td>
<td>12.38</td>
<td>11.35</td>
<td>11.48</td>
<td>13.25</td>
<td>5.20</td>
</tr>
<tr>
<td></td>
<td>24.53</td>
<td>24.80</td>
<td>22.67</td>
<td>20.93</td>
<td>19.33</td>
<td>16.67</td>
</tr>
<tr>
<td></td>
<td>3.58</td>
<td>3.36</td>
<td>3.83</td>
<td>3.77</td>
<td>3.18</td>
<td>1.63</td>
</tr>
<tr>
<td>Personal</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interaction +</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interaction +</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Observation</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Team structure, team familiarity and team performance</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Independent of team familiarity, the team performance is highest in the teams organized as task-based sub-teams, and comparable in the flat teams and the flat teams with social cliques, though marginally higher in the flat teams, in general, Table 6.12.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
In the teams organized as task-based sub-teams, the difference between the best (16, SD=0) and the worst (25.33, SD=4.76) team performance is lower than the difference between the best (16, SD=0) and the worst (55.41, SD=15.37) team performance in the flat teams, or the flat teams with social cliques [best=16 (SD=0) and worst=56.27 (SD=17.32)]. Therefore, team familiarity plays a bigger role in the flat teams (F=249.55, P-value<0.001)\(^{21}\) and the flat teams with social cliques (F=262.63, P-value<0.001) as compared to the teams organized into task-based sub-groups (F=75.99, P-value<0.001). Dividing the teams into task-based sub-groups enhances the team’s performance by narrowing down the exploration space.

### 6.2.2.4 Experiments with non-routine tasks and team familiarity

Table 6.13 summarizes the results of the simulations with the non-routine tasks and 12 NR-Agents. As with the routine tasks, at team familiarity=0%, the team performance is not affected by the agents’ learning modes. In simulations with the non-routine tasks, even at team familiarity=100% and with all learning modes available to the agents, the team does not attain optimal performance consistently because of the variations in solution selection.

#### Table 6.13: Experiments with non-routine tasks and team familiarity

<table>
<thead>
<tr>
<th>Learning modes</th>
<th>% Team familiarity</th>
<th>Flat Teams</th>
<th>Social Cliques</th>
<th>Task-based sub-teams</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>No. of messages</td>
<td>Std Dev</td>
<td>No. of messages</td>
<td>Std Dev</td>
</tr>
<tr>
<td>Personal Interaction</td>
<td>0</td>
<td>153.52</td>
<td>21.91</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>17</td>
<td>143.53</td>
<td>20.86</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>33</td>
<td>142.07</td>
<td>22.58</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>143.67</td>
<td>23.11</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>66</td>
<td>144.87</td>
<td>25.51</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>83</td>
<td>139.80</td>
<td>19.85</td>
<td>-</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>78.47</td>
<td>17.66</td>
<td>-</td>
</tr>
<tr>
<td>Personal Interaction +</td>
<td>0</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Interaction observation</td>
<td>17</td>
<td>150.27</td>
<td>16.59</td>
<td>147.47</td>
</tr>
<tr>
<td></td>
<td>33</td>
<td>145.73</td>
<td>22.55</td>
<td>147.07</td>
</tr>
<tr>
<td></td>
<td>50</td>
<td>145.67</td>
<td>17.94</td>
<td>142.47</td>
</tr>
<tr>
<td></td>
<td>66</td>
<td>144.13</td>
<td>19.72</td>
<td>142.53</td>
</tr>
<tr>
<td></td>
<td>83</td>
<td>142.93</td>
<td>15.85</td>
<td>139.67</td>
</tr>
</tbody>
</table>

\(^{21}\) Significance values, F obtained from ANOVA tests that compare results from experiments with different levels of team familiarity
In simulations with the non-routine tasks as well, the team performance increases with the increase in team familiarity. However, for the teams working on the non-routine tasks, the increase in the team performance with the increase in team familiarity is not significant at lower levels of team familiarity. In contrast, at levels close to 100%, there is significant increase in the team performance with the increase in the team familiarity. For the teams working on the routine tasks, the increase in the team performance, with the increase in the team familiarity, is relatively gradual across the various levels of team familiarity. A plausible explanation for the difference in the correlation of team familiarity and the team performance across the task types is that the critical task network (section 5.10) has greater significance for the teams working on the non-routine tasks than for the teams working on the routine tasks. If the non-routine task is to be allocated to a new agent, then not only does the task allocator have to identify the relevant expert, but the task performer may also need to learn the task allocator’s acceptable solution range during the test round, which negatively affects the team performance. In the non-routine tasks, the sub-solutions need to be compatible at integration level. Hence, even if an agent

\[ R^2 = 0.9986 \text{, and in teams organized as sub-teams, } R^2 = 0.9927. \]

\[ R^2 = 0.7368 \text{, and in teams organized as sub-teams, } R^2 = 0.7882. \]
has observed the details of the range of sub-solutions accepted by another agent during the training round, the same sub-solution may not be acceptable in the test round because of the changes in the sub-solutions proposed by another agent. In addition, even a small change in the task specifications at higher level may have a cascading effect on the acceptable range of solutions at the lower levels, requiring the agents to explore more solutions before they are accepted. When the team familiarity is close to 100%, then the critical task network is completely retained. Thus, even if the task leader chosen in the test round is different from the task leader in the training round, the likelihood that most of the agents that performed the tasks in the training round will again get the same task in the test round is higher. These agents would already have identified the agent in the related node, and also partially learnt about their desired solution range (based on what solutions were accepted in the training round). Retention of the agents that perform the coordination tasks should be more critical to the team performance, since non-routine tasks need coordination. Because of the coordination tasks, only a few agents in the team exchange most number of messages, Figure 6.2. These agents have personal interactions with more number of agents. Hence, if these agents are retained in the training round, the team performance should be much higher. In the teams with team familiarity = 100%, these agents are certainly retained, but that may not be happening in the teams with lower levels of team familiarity.

![Pattern of message exchange across team members](image)

**Figure 6.2: Pattern of message exchange across team members in teams working on non-routine tasks**

On the other hand, in the teams working on the routine tasks, the number of messages exchanged across the team is more uniform, Figure 6.3. Hence, it is likely that the effects of reduction in the team familiarity are more gradual in such teams because the replacement of any agent from the team may have a comparable effect on the team performance.

---

24 Results shown for 15 simulation runs
For the teams working on the routine tasks as well as for the teams working on the non-routine tasks, the team performance is highest in the teams organized as task-based sub-teams, and comparable in the flat teams and the flat teams distributed into social cliques.

### 6.2.2.5 Experiments with busyness and team familiarity

Table 6.14 summarizes the results of the simulations with the routine tasks and 12 R-Agents. The simulations for assessing the correlation between busyness and team familiarity, with respect to the team performance, involve the training rounds as well as the test rounds. During the training rounds, busyness determines how much the agents learnt about each other. Once the training round is over, some of the agents are retained in the test round. If the busyness level is lower during the training round, and the team familiarity level is higher in the test rounds, then the team performance should be higher, i.e., either the reduction in team familiarity, or the increase in busyness, should result in lower team performance because in either case, the agents should have lower level of TMM formation.

<table>
<thead>
<tr>
<th>Flat Teams, Routine tasks, All learning modes (i.e., PI+IO+TO)</th>
<th>TF%</th>
<th>0</th>
<th>17</th>
<th>33</th>
<th>55</th>
<th>66</th>
<th>83</th>
<th>100</th>
</tr>
</thead>
<tbody>
<tr>
<td>BL%</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of messages</td>
<td>59.08</td>
<td>55.41</td>
<td>51.80</td>
<td>43.63</td>
<td>36.60</td>
<td>26.40</td>
<td>16.00</td>
<td></td>
</tr>
<tr>
<td>Std Dev</td>
<td>14.27</td>
<td>15.37</td>
<td>10.19</td>
<td>13.26</td>
<td>10.47</td>
<td>9.05</td>
<td>0.00</td>
<td></td>
</tr>
<tr>
<td>25</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>No. of messages</td>
<td>-</td>
<td>56.27</td>
<td>54.53</td>
<td>47.6</td>
<td>41.47</td>
<td>26.40</td>
<td>16.00</td>
<td></td>
</tr>
</tbody>
</table>

Results shown for 15 simulation runs
<table>
<thead>
<tr>
<th>No. of messages</th>
<th>Std Dev</th>
<th>10.77</th>
<th>12.25</th>
<th>11.86</th>
<th>12.95</th>
<th>7.18</th>
<th>0.00</th>
</tr>
</thead>
<tbody>
<tr>
<td>33</td>
<td>No. of messages</td>
<td>61.60</td>
<td>55.47</td>
<td>53.20</td>
<td>47.33</td>
<td>27.20</td>
<td>16.53</td>
</tr>
<tr>
<td></td>
<td>Std Dev</td>
<td>13.42</td>
<td>13.02</td>
<td>17.58</td>
<td>15.27</td>
<td>9.97</td>
<td>1.60</td>
</tr>
<tr>
<td>50</td>
<td>No. of messages</td>
<td>60.13</td>
<td>52.80</td>
<td>41.87</td>
<td>36.53</td>
<td>28.67</td>
<td>16.53</td>
</tr>
<tr>
<td></td>
<td>Std Dev</td>
<td>11.22</td>
<td>13.15</td>
<td>10.10</td>
<td>9.78</td>
<td>10.02</td>
<td>1.40</td>
</tr>
<tr>
<td>66</td>
<td>No. of messages</td>
<td>54.93</td>
<td>55.33</td>
<td>48.27</td>
<td>48.13</td>
<td>29.87</td>
<td>16.93</td>
</tr>
<tr>
<td></td>
<td>Std Dev</td>
<td>11.70</td>
<td>11.70</td>
<td>16.73</td>
<td>8.83</td>
<td>7.87</td>
<td>2.49</td>
</tr>
<tr>
<td>75</td>
<td>No. of messages</td>
<td>60.40</td>
<td>55.20</td>
<td>48.80</td>
<td>42.67</td>
<td>33.33</td>
<td>17.33</td>
</tr>
<tr>
<td></td>
<td>Std Dev</td>
<td>17.92</td>
<td>13.69</td>
<td>10.30</td>
<td>15.27</td>
<td>13.62</td>
<td>3.60</td>
</tr>
</tbody>
</table>

The results show that irrespective of the busyness level of the agents during the training rounds, the team performance increases with the increase in team familiarity during the test rounds. Hence, for improved team performance, the team familiarity is a more significant factor than the busyness level. A sensitivity analysis of the results from the experiments with the routine tasks ranked team familiarity (Error ratio=5.314)\textsuperscript{26} higher than busyness level (Error ratio=1.257).

However, busyness level does influence the amount of increase in the team performance with the increase in team familiarity. At team familiarity=100%, busyness level has marginal influence on the team performance because all the agents are retained, and the critical task network developed during the training rounds is intact for the test rounds. Thus, excluding the case of team familiarity=100%, the differences in the team performance across the lower (17%) and higher level (83%) of team familiarity are compared at different busyness levels. The results show that the increase in the team performance with the increase in team familiarity is higher at lower busyness levels. For example, at BL=0%, the corresponding values of the team performance are 55.41 (SD=15.37) and 26.40 (SD=9.05), respectively. The same values at BL=50%, are 60.13 (SD=11.22) and 28.67 (SD=10.02), respectively.

\textsuperscript{26} Error ratios indicate the predictability of the results if the variable is not available. Hence, higher error ratios indicate greater significance of the variable.
Chapter 7
Research Findings

This chapter discusses the findings in terms of the research hypotheses and the data collected in Chapter 6. The team behaviour patterns observed in the simulations are presented.

7.1 Social learning modes, busyness level, and level of team familiarity:

7.1.1 Learning modes, busyness level and team performance

It was hypothesized (hypothesis 1) that when compared to the teams that have all modes of learning available to the agents, the decrease in team performance, with the increase in busyness levels, is lower in the teams that have partial modes of learning available to the agents. The decrease in team performance, with the increase in busyness levels, is lowest for the teams in which the agents learn only from personal interactions.

Therefore, in these experiments, busyness level and learning modes were the independent variables, and the team performance was measured. For each case of learning modes, simulations were conducted with different busyness levels. The results from the experiments with the routine tasks and flat teams with 100% team familiarity are shown in Figure 7.1(a). Figure 7.1(b) shows a graph for similar experiments conducted with the non-routine tasks and flat teams with 100% team familiarity.

Figure 7.1(a) and Figure 7.1(b) illustrate that in general, the team performance increases with the decrease in busyness level. However, the findings partially reject hypothesis 1 because, in Figure 7.1(a), the slope is steeper for the partial learning modes, which contradicts the hypothesis. In Figure 7.1(b), the slope is same for PI+TO+IO and PI+TO, while the slope for PI+IO is flatter, which partially supports the hypothesis.
Table 7.1: Difference in team performance across busyness levels (0, 25, 33, 50, 66 and 75%)

<table>
<thead>
<tr>
<th>Task</th>
<th>LM</th>
<th>TS</th>
<th>TF %</th>
<th>F</th>
<th>P-value</th>
<th>F-critical</th>
<th>F&gt; F-critical</th>
</tr>
</thead>
<tbody>
<tr>
<td>NR</td>
<td>PI+IO+TO</td>
<td>Flat</td>
<td>100</td>
<td>3.846</td>
<td>0.0025</td>
<td>2.266</td>
<td>Yes</td>
</tr>
<tr>
<td>NR</td>
<td>PI+TO</td>
<td>Flat</td>
<td>100</td>
<td>2.719</td>
<td>0.0215</td>
<td>2.266</td>
<td>Yes</td>
</tr>
<tr>
<td>NR</td>
<td>PI+IO</td>
<td>Flat</td>
<td>100</td>
<td>0.072</td>
<td>0.9963</td>
<td>2.266</td>
<td>No</td>
</tr>
<tr>
<td>NR</td>
<td>PI+IO+TO</td>
<td>Sub-teams</td>
<td>100</td>
<td>1.673</td>
<td>0.1434</td>
<td>2.266</td>
<td>No</td>
</tr>
<tr>
<td>NR</td>
<td>PI+TO</td>
<td>Sub-teams</td>
<td>100</td>
<td>3.134</td>
<td>0.0098</td>
<td>2.266</td>
<td>Yes</td>
</tr>
<tr>
<td>NR</td>
<td>PI+IO</td>
<td>Sub-teams</td>
<td>100</td>
<td>1.328</td>
<td>0.2543</td>
<td>2.266</td>
<td>No</td>
</tr>
<tr>
<td>NR</td>
<td>PI+IO+TO</td>
<td>Social Cliques</td>
<td>100</td>
<td>1.5549</td>
<td>0.1753</td>
<td>2.266</td>
<td>No</td>
</tr>
<tr>
<td>R</td>
<td>PI+IO+TO</td>
<td>Flat</td>
<td>100</td>
<td>2.1494</td>
<td>0.0618</td>
<td>2.266</td>
<td>No</td>
</tr>
<tr>
<td>R</td>
<td>PI+TO</td>
<td>Flat</td>
<td>100</td>
<td>1.8747</td>
<td>0.1011</td>
<td>2.266</td>
<td>No</td>
</tr>
<tr>
<td>R</td>
<td>PI+IO</td>
<td>Flat</td>
<td>100</td>
<td>3.1058</td>
<td>0.0103</td>
<td>2.266</td>
<td>Yes</td>
</tr>
<tr>
<td>R</td>
<td>PI+IO+TO</td>
<td>Sub-teams</td>
<td>100</td>
<td>5.5380</td>
<td>&lt;0.0001</td>
<td>2.266</td>
<td>Yes</td>
</tr>
<tr>
<td>R</td>
<td>PI+TO</td>
<td>Sub-teams</td>
<td>100</td>
<td>6.7893</td>
<td>&lt;0.0001</td>
<td>2.266</td>
<td>Yes</td>
</tr>
</tbody>
</table>
It was hypothesized (hypothesis 2) that when compared to the teams that have all modes of learning available to the agents, the decrease in levels of TMM formation, with the increase in busyness levels, is lower in the teams that have partial modes of learning available to the agents. The decrease in levels of TMM formation, with the increase in busyness levels, is lowest for the teams in which the agents learn only from personal interactions.

Therefore, in these experiments, busyness level and learning modes were the independent variables, and the level of TMM formation was measured. For each case of learning modes, simulations were conducted with the different busyness levels. The results from the experiments with the routine tasks are shown in Figure 7.2(a). Figure 7.2(b) shows a graph for similar experiments conducted with the non-routine tasks and flat teams.

![Figure 7.2: Busyness levels and TMM formation across different learning modes](image)
Figure 7.2(a) and Figure 7.2(b) illustrate that the slope is steepest for PI+IO+TO and least for PI. These findings support hypothesis 2. When the agents learn only from personal interactions, busyness has no influence on TMM formation. In this situation busyness effects observations only (section 4.1.4). The results show that the effects of interaction observations to TMM formation is significantly less when compared to the effects of task observations27.

7.1.3 Learning modes, team familiarity and team performance

It was hypothesized (hypothesis 3) that when compared to the teams that have all modes of learning available to the agents, the increase in team performance, with the increase in levels of team familiarity, is lower in the teams that have partial modes of learning available to the agents. The increase in team performance, with the increase in levels of team familiarity, is lowest for the teams in which the agents learn only from personal interactions.

Therefore, in these experiments, level of team familiarity and learning modes were the independent variables and the team performance was measured. For each case of learning modes, simulations were conducted with the different levels of team familiarity. The results from the experiments with the routine tasks are shown in Figure 7.3(a). Figure 7.3(b) shows a graph for similar experiments conducted with the non-routine tasks and flat team.

Figure 7.3: Team familiarity and team performance across different learning modes

Figure 7.3(b) illustrates that the pattern of increase in the team performance, with the increase in team familiarity, is similar for all cases of learning modes. Figure 7.3(a) also shows the same pattern. However, differences in patterns exist across Figure 7.3(a) and Figure 7.3(b). Some noticeable

27 When the task=routine and TF=100%, an ANOVA test for PI+IO and PI, shows a significance value, F=6.684 (P<0.011). The corresponding value for PI+TO is F=48.432 (P<0.001). When the task=non-routine and TF=100%, the same value for PI+IO is F=3.390 (P<0.068), and for PI+TO, F=29.775 (P<0.001).
differences in the effects of team familiarity across the different learning modes are also observed within Figure 7.3(a). This partially supports hypothesis 3.

In Figure 7.3(b), i.e., non-routine tasks, the increase in team familiarity has no significant effect on the team performance, unless the team familiarity level is close to 100%. The task observations have a greater effect on the team performance than the interaction observations.

The differences observed in Figure 7.3(a) are clearer in Figure 7.4, which plots separate graphs for each learning mode. When the agents learn only from PI, Figure 7.4(a), a threshold exists beyond which the slope increases. In these results, this threshold for personal interactions is observed at around 66% team familiarity. A similar threshold is observed in the cases PI+IO or PI+TO, Figure 7.4(b) and Figure 7.4(c). However, for PI+IO+TO, no noticeable threshold is observed, Figure 7.4(d). Thus, with additional modes of learning, this threshold point moves farther from 100% team familiarity level.

**Figure 7.4: Team familiarity and team performance for agents (Routine task)**

---

28 Comparison of team performances across the different values for TF (0, 17, 33, 50, 66, 83 and 100%), with routine tasks and PI+IO+TO gives F=249.549 (P<0.0001). Corresponding values for PI+IO, PI+TO and PI are F=92.841 (P<0.0001), F=93.302 (P<0.0001), and F=77.356 (P<0.0001), respectively. These ANOVA tests compare the sample means obtained from experiments where only TF is a variable. Thus, for each case of learning modes, the experiments are conducted at different levels of TF. The significance value (F) shows the significance of the effects of TF on team performance for a given learning mode.

29 Comparing performances for TF=17, 33, 50, 66% an ANOVA gives F=0.377, P=0.770.

30 Comparing performances for TF=83 and 100% an ANOVA gives F=677.863, P<0.0001.

31 Comparing performances for PI+TO and PI+IO, the significance value, F=37.122, P<0.0001.
Table 7.2 shows the one-way ANOVA results that compare the performances across different learning modes at different levels of team familiarity. Each row in Table 7.2 summarizes the results from a different ANOVA test. For routine tasks, at TF below 50%, the differences in the performances across the learning modes are not significant.

**Table 7.2: Difference in effects of team familiarity on team performance across the four learning modes, i.e., PI, PI+TO, PI+IO, PI+IO+TO at BL=0**

<table>
<thead>
<tr>
<th>TF %</th>
<th>TS</th>
<th>Task</th>
<th>df</th>
<th>MMS</th>
<th>F</th>
<th>P-value</th>
<th>F-critical</th>
<th>F&gt;F-critical</th>
</tr>
</thead>
<tbody>
<tr>
<td>17</td>
<td>Flat</td>
<td>R</td>
<td>3</td>
<td>442.960</td>
<td>2.3829</td>
<td>0.0701</td>
<td>2.6429</td>
<td>No</td>
</tr>
<tr>
<td>33</td>
<td>Flat</td>
<td>R</td>
<td>3</td>
<td>151.128</td>
<td>0.8751</td>
<td>0.4547</td>
<td>2.6429</td>
<td>No</td>
</tr>
<tr>
<td>50</td>
<td>Flat</td>
<td>R</td>
<td>3</td>
<td>1058.05</td>
<td>5.7821</td>
<td>0.0008</td>
<td>2.6429</td>
<td>Yes</td>
</tr>
<tr>
<td>66</td>
<td>Flat</td>
<td>R</td>
<td>3</td>
<td>1699.782</td>
<td>11.3326</td>
<td>&lt;0.0001</td>
<td>2.6429</td>
<td>Yes</td>
</tr>
<tr>
<td>83</td>
<td>Flat</td>
<td>R</td>
<td>3</td>
<td>429.689</td>
<td>3.8530</td>
<td>0.0102</td>
<td>2.6429</td>
<td>Yes</td>
</tr>
<tr>
<td>100</td>
<td>Flat</td>
<td>R</td>
<td>3</td>
<td>306.683</td>
<td>27.3997</td>
<td>&lt;0.0001</td>
<td>2.6429</td>
<td>Yes</td>
</tr>
<tr>
<td>17</td>
<td>Flat</td>
<td>NR</td>
<td>3</td>
<td>778.556</td>
<td>1.5074</td>
<td>0.2133</td>
<td>2.6429</td>
<td>No</td>
</tr>
<tr>
<td>33</td>
<td>Flat</td>
<td>NR</td>
<td>3</td>
<td>396.356</td>
<td>0.7631</td>
<td>0.5158</td>
<td>2.6429</td>
<td>No</td>
</tr>
<tr>
<td>50</td>
<td>Flat</td>
<td>NR</td>
<td>3</td>
<td>202.061</td>
<td>0.4603</td>
<td>0.7103</td>
<td>2.6429</td>
<td>No</td>
</tr>
<tr>
<td>66</td>
<td>Flat</td>
<td>NR</td>
<td>3</td>
<td>609.844</td>
<td>1.3841</td>
<td>0.2483</td>
<td>2.6429</td>
<td>No</td>
</tr>
<tr>
<td>83</td>
<td>Flat</td>
<td>NR</td>
<td>3</td>
<td>341.5111</td>
<td>0.8179</td>
<td>0.4851</td>
<td>2.6429</td>
<td>No</td>
</tr>
<tr>
<td>100</td>
<td>Flat</td>
<td>NR</td>
<td>3</td>
<td>3728.24</td>
<td>24.1204</td>
<td>&lt;0.0001</td>
<td>2.6429</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table 7.3 shows the results for a correlation analysis between team familiarity and team performance for different cases. Each row summarizes the results of correlations at lower, higher and overall team familiarity levels. The results show that correlation between team familiarity and team performance is weaker at lower levels of team familiarity.

**Table 7.3: Team familiarity and team performance (BL=0%)**

<table>
<thead>
<tr>
<th>LM</th>
<th>Task</th>
<th>TS</th>
<th>TF values (low/ high/ overall)</th>
<th>Correlation</th>
</tr>
</thead>
<tbody>
<tr>
<td>PI+IO+TO</td>
<td>R</td>
<td>Flat</td>
<td>(0, 17, 33, 50) / (50, 66, 83, 100) / all</td>
<td>0.9773 / 0.9973 / 0.9839</td>
</tr>
<tr>
<td>PI+IO+TO</td>
<td>R</td>
<td>Social Clique</td>
<td>(0, 17, 33, 50) / (50, 66, 83, 100) / all</td>
<td>0.9149 / 0.9992 / 0.9694</td>
</tr>
<tr>
<td>PI+IO+TO</td>
<td>R</td>
<td>Sub-team</td>
<td>(0, 17, 33, 50) / (50, 66, 83, 100) / all</td>
<td>0.9242 / 0.9862 / 0.9824</td>
</tr>
</tbody>
</table>
As was observed in Figure 7.3(b), when the task is non-routine, team familiarity has no significant effect on team performance\textsuperscript{32}, unless the team familiarity is close to 100\%\textsuperscript{33}. This suggests that when the task is non-routine, a broken critical task network has greater effect on the team performance. It is likely that when the task is non-routine, the knowledge of the task allocator’s capability range become critical. Any changes made in the selection of a sub-solution may require other sub-solutions to change because the solutions need to be compatible. Hence, the team performance is significantly affected. At 100\% team familiarity, the critical task network is intact and most of the task performers in the test round are likely to be the same as in the training round. Thus, the task performer may already have narrowed down the capability range of the task allocator to a smaller solution span (section 5.4.3), based on the solutions accepted in the test round.

Thus, it is conjectured that as the task complexity increases, there is more information required to enhance the team performance, and, hence, the threshold point shifts closer to 100\% team familiarity. This explains the shift in threshold point to close to 100\% team familiarity in the simulations with the non-routine task.

The results show that the rate of increase in team performance with the increase in team familiarity is contingent upon the task. However, it can generally be stated that:

1. The team performance increases with the increase in team familiarity.
2. The rate of increase in the team performance, with the increase in team familiarity, is faster at higher levels of team familiarity.
3. There exists a threshold beyond which the rate of increase in the team performance, with the increase in team familiarity, is faster.
4. The contributions of social observations (task observation and interaction observation) to the team performance are more evident at intermediate and higher levels of team familiarity.

\textsuperscript{32} Comparing performances for TF=17, 33, 50, 66\% an ANOVA gives F=0.377, P=0.770.
\textsuperscript{33} Comparing performances for TF=83 and 100\% an ANOVA gives F=677.863, P<0.0001.
7.1.4 Team familiarity, busyness level and team performance

It was hypothesized (hypothesis 4) that the increase in the team performance, with the increase in the level of team familiarity, will be greater when the busyness levels are lower.

Therefore, in these experiments, busyness and team familiarity were the independent variables, and the team performance was measured. For each level of team familiarity, simulations were conducted with different busyness levels. The results from the experiments with the routine tasks and flat teams with all learning modes available to the agents are shown in Figure 7.5(a). While Figure 7.5(a) shows the pattern of change in the team performance across the different levels of team familiarity (Team familiarity at X-axis) for different cases of busyness levels, Figure 7.5(b) plots the same results in terms of the change in busyness levels (Busyness level in X-axis) for different cases of levels of team familiarity.

![Figure 7.5: Team familiarity and busyness levels in terms of team performance](image)

In Figure 7.5(a), the overall change in the team performance across the highest and the lowest levels of team familiarity are similar across all busyness levels because busyness level has low or no significant effect on the team performance [Table 7.1, Figure 7.1(a) and Figure 7.1(b)].

Table 7.4 summarizes the results of correlation analysis for team familiarity and team performance. Each row provides results of this correlation at given busyness level (column 1). Team familiarity has significant effect on the team performance irrespective of the busyness levels. The effects of team familiarity do not necessarily decrease with increase in busyness. Thus, the findings partially reject hypothesis 4.

<table>
<thead>
<tr>
<th>BL%</th>
<th>TF values</th>
<th>df</th>
<th>MS</th>
<th>F</th>
<th>P-value</th>
<th>Correlation (TF-TP)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>17, 33, 50, 66, 83, 100</td>
<td>5</td>
<td>6864.036</td>
<td>55.841</td>
<td>&lt;0.0001</td>
<td>0.991</td>
</tr>
</tbody>
</table>
Table 7.5 summarizes the ANOVA results comparing team performance across different busyness levels for given team familiarity level. Each row evaluates the effects of busyness level on team performance at a given level of team familiarity (column 4). As shown in Figure 7.5(b), if the team familiarity is in the intermediate range (50% to 83%), then busyness level has a significant but small effect on the team performance.

Table 7.5: Difference in team performance across busyness levels (0, 25, 33, 50, 66 and 75%) at given team familiarity (17, 33, 50, 66, 83 and 100%)

<table>
<thead>
<tr>
<th>Task</th>
<th>LM</th>
<th>TS</th>
<th>TF %</th>
<th>F</th>
<th>P-value</th>
<th>F-critical</th>
<th>F&gt; F-critical</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>PI+IO+TO</td>
<td>Flat</td>
<td>17</td>
<td>1.337</td>
<td>0.2507</td>
<td>2.266</td>
<td>No</td>
</tr>
<tr>
<td>R</td>
<td>PI+IO+TO</td>
<td>Flat</td>
<td>33</td>
<td>0.7383</td>
<td>0.5957</td>
<td>2.266</td>
<td>No</td>
</tr>
<tr>
<td>R</td>
<td>PI+IO+TO</td>
<td>Flat</td>
<td>50</td>
<td>2.629</td>
<td>0.0255</td>
<td>2.266</td>
<td>Yes</td>
</tr>
<tr>
<td>R</td>
<td>PI+IO+TO</td>
<td>Flat</td>
<td>66</td>
<td>4.542</td>
<td>0.0383</td>
<td>2.266</td>
<td>Yes</td>
</tr>
<tr>
<td>R</td>
<td>PI+IO+TO</td>
<td>Flat</td>
<td>83</td>
<td>2.4111</td>
<td>0.0618</td>
<td>2.266</td>
<td>Yes</td>
</tr>
<tr>
<td>R</td>
<td>PI+IO+TO</td>
<td>Flat</td>
<td>100</td>
<td>2.1494</td>
<td>0.0618</td>
<td>2.266</td>
<td>No</td>
</tr>
</tbody>
</table>

Busyness does not influence the results at low team familiarity because the difference in the team performance across the different learning modes is not significant at lower levels of team familiarity (section 7.1.3). When team familiarity=100%, the team performance is high across all learning modes (Section 7.1.3). Therefore, at this point, even though the role of social learning modes is significant (F=27.3997), the advantages of social observation may not be evident (Section 7.1.3). When the level of team familiarity is in the intermediate range (TF=50-83%), the differences in team performance across the different learning modes are greater (Figure 7.3(a)). Hence, busyness significantly decreases the positive effects of team familiarity on the team performance, Table 7.5.

The explanation of a transition from low significance of the social learning modes, at the lower levels of team familiarity, to higher significance of the social learning modes, at the higher levels of
team familiarity, is supported by the inversion of the curves in Figure 7.5(b), at TF=66% and TF=83%. The rate of decrease in the team performance, with the increase in busyness, is faster when the social learning modes have significant effect on the team performance (higher team familiarity). The rate of decrease in the team performance, with the increase in busyness, is slower when the social learning modes have no significant effect on the team performance (lower team familiarity).

7.2 Social learning modes and team structure:

In the simulations discussed above, the effects of learning modes, busyness level, and team familiarity were explored independent of the variations across the team structure. This section discusses the effects of team structure on TMM formation and team performance through experiments with different combinations of the other variables, i.e., learning modes, busyness level, and level of team familiarity.

7.2.1 Team structure, learning modes and team performance

It was hypothesized (hypothesis 5) that the difference in the team performance across the teams with different learning modes available to the agents will be greater for the teams organized as sub-teams, lower for the flat teams and lowest for the flat teams with social cliques.

Therefore, in these experiments, team structures and learning modes were the independent variables, and the team performance was measured. For each case of learning modes, simulations were conducted with the different team structures. The results from the experiments with the routine tasks and the non-routine tasks are shown in Figure 7.6(a) and Figure 7.6(b), respectively.

![Figure 7.6: Team structure and modes of learning in terms of team performance](image)

Table 7.6 summarizes the ANOVA results that compare the team performance across the learning modes for given cases. For example, the results in the first row show that the effects of the learning modes on the team performance is greatest in the teams organized as social cliques and least in task-bases sub-teams. The experiments in this row were conducted with the routine tasks and at TF=66%.
These findings partially reject hypothesis 5. There is no clear pattern in the effects of the team structure on the team performance across the learning modes, (Figure 7.6 and Table 7.6). If the task is non-routine, the hypothesis is valid. If the task is routine, the opposite holds true. Therefore, the relative role of the different learning modes on the team performance across the different team structures is contingent on the task type.

### Table 7.6: Differences in team performance across learning modes (PI, PI+IO, PI+TO, PI+IO+TO)

<table>
<thead>
<tr>
<th>Task</th>
<th>TF%</th>
<th>Flat</th>
<th>Social Cliques</th>
<th>Sub-teams</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>66</td>
<td>11.333 (&lt;0.0001)</td>
<td>16.872 (&lt;0.0001)</td>
<td>0.2322 (=0.8739)</td>
</tr>
<tr>
<td></td>
<td>83</td>
<td>3.853 (=0.0102)</td>
<td>28.103 (&lt;0.0001)</td>
<td>0.5722 (=0.6339)</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>27.400 (&lt;0.0001)</td>
<td>30.065 (&lt;0.0001)</td>
<td>8.9394 (&lt;0.0001)</td>
</tr>
<tr>
<td>NR</td>
<td>66</td>
<td>1.384 (=0.2483)</td>
<td>0.9579 (=0.4133)</td>
<td>25.4229 (&lt;0.0001)</td>
</tr>
<tr>
<td></td>
<td>83</td>
<td>0.818 (=0.4851)</td>
<td>0.2755 (=0.8431)</td>
<td>25.8352 (&lt;0.0001)</td>
</tr>
<tr>
<td></td>
<td>100</td>
<td>24.120 (&lt;0.0001)</td>
<td>5.8100 (=0.0008)</td>
<td>43.1001 (&lt;0.0001)</td>
</tr>
</tbody>
</table>

#### 7.2.2 Team structure, learning modes and TMM formation

It was hypothesized (hypothesis 6) that the difference in the amount (density) of TMM formation across the different learning modes will be higher for flat teams, lower for flat teams with social cliques, and lowest for teams organized into task-based sub-groups.

Therefore, in these experiments, team structures and learning modes were the independent variables, and the level of TMM formation was measured. For each case of learning modes, simulations were conducted with the different team structures. The results from the experiments with the routine tasks and the non-routine tasks are shown in Figure 7.7(a) and Figure 7.7(b), respectively.

![Figure 7.7: Team structure and modes of learning in terms of level of TMM formation](image_url)
Figure 7.7(a) and Figure 7.7(b) illustrate the level of TMM formation across the different learning modes for each team structure. The findings support hypothesis 6. In both the Figures, when the team is flat, the difference in TMM formation for experiments with PI+IO+TO and PI is highest. These differences in TMM formation are lower if the teams are flat but divided into social cliques and least if the team is organized into task-based sub-groups. Thus, as was discussed in section 6.2.2.1, it is likely that if the overall team sizes are similar, flat teams have the largest effective team size in terms of their effects on TMM formation. The teams organized into task-based sub-teams have the smallest effective team size.

7.2.3 Team structure and efficiency of formed TMM

It was hypothesized (hypothesis 7) that the efficiency of TMM formation is highest in the teams organized into task-based sub-teams, lower in the flat teams, and lowest in the flat teams with social cliques.

In this research, the efficiency of TMM formation is calculated as the ratio of the team performance in the test round to the level of TMM formation in the training round. The results from the experiments with the routine tasks and the non-routine tasks are shown in Table 7.7 and Table 7.8, respectively.

<table>
<thead>
<tr>
<th>TF</th>
<th>Flat Team</th>
<th>Social Cliques</th>
<th>Sub-teams</th>
</tr>
</thead>
<tbody>
<tr>
<td>100%</td>
<td>0.18=(9.33/50.98)</td>
<td>0.38=(9.33/24.30)</td>
<td>1.17=(9.33/7.96)</td>
</tr>
<tr>
<td>66%</td>
<td>0.08=(4.08/50.98)</td>
<td>0.16=(3.99/24.30)</td>
<td>0.88=(7.00/7.96)</td>
</tr>
<tr>
<td>17%</td>
<td>0.05=(2.70/50.98)</td>
<td>0.11=(2.65/24.30)</td>
<td>0.74=(5.79/7.96)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>TF</th>
<th>Flat Team</th>
<th>Social Cliques</th>
<th>Sub-teams</th>
</tr>
</thead>
<tbody>
<tr>
<td>100%</td>
<td>0.05=(2.40/46.34)</td>
<td>0.10=(2.21/22.56)</td>
<td>0.27=(2.47/9.02)</td>
</tr>
<tr>
<td>66%</td>
<td>0.02=(1.08/46.34)</td>
<td>0.05=(1.08/22.56)</td>
<td>0.23=(2.10/9.02)</td>
</tr>
<tr>
<td>17%</td>
<td>0.02=(1.05/46.34)</td>
<td>0.04=(1.00/22.56)</td>
<td>0.23=(2.10/9.02)</td>
</tr>
</tbody>
</table>

Figure 7.8 illustrates that for all the simulation cases, the efficiency of TMM formation is highest for the teams organized into task-based sub-groups, irrespective of the task type or the level of team familiarity. However, all the simulations show that the efficiency of TMM formation is lower in the
flat teams as compared to the flat teams with social cliques. This is opposite to the hypothesis. Therefore, the results only **partially support hypothesis 7**.

![Team Structure and TMM Efficiency](image)

**Figure 7.8: Team structure and efficiency of formed TMM**

The agents with the related task competencies may be distributed across different social cliques. Hence, it was expected that the teams distributed into social cliques will perform worst because opportunities for identifying the relevant experts through social observations are likely to be least in such teams. However, results suggest that this may not be the case.

In these simulations, the expertise distribution across the social groups was not varied. In all these simulations, most of the agents grouped together in the same social group also have related task-dependencies. Thus in-group observations are likely to enhance efficiency of TMM. This may or may not be true for real world teams. Hence, it may be useful to conduct experiments with different expertise distributions across the social groups.

The differences in the efficiency of TMM across the team structures are due to the differences in the amount of important TMM formation and the amount of overall TMM formation. For a team working on the routine tasks, it may be useful to know the competence of each agent in each of the project related tasks (overall TMM). However, the most important aspect for an agent to know is who to allocate the task that follows the task the agent itself performs. Comparative results for the overall TMM formation and the important TMM formation across the different team structures are shown Figure 7.9 and Figure 7.10, respectively. The level of important TMM formation is calculated by considering the number of important elements in the TMM matrix that have been learnt.

At lower busyness levels, the difference in the amounts of overall TMM formation across the different team structures (Figure 7.9) is much higher as compared to the difference in the level of important TMM formation across the different team structures (Figure 7.10).
Based on the simulation results, hypothesis 7 is modified to state that

When all modes of social learning are available to the agents, the increase in the efficiency of TMM formation is highest when the team is organized into task-based sub-teams, lower when the team is flat but grouped into social cliques, and lowest when the team is flat.

7.2.4 Team structure, busyness level and team performance

It was hypothesized (hypothesis 8) that the decrease in the team performance, with the increase in the busyness level, should be highest for the teams organized as task-based sub-groups, lower for the flat teams, and lowest for the flats teams grouped into social cliques.

Therefore, in these experiments, team structure and busyness level were the independent variables, and the team performance was measured. For each team structure, simulations were conducted with different busyness levels. The results from the experiments with the routine tasks and the non-routine tasks are shown in Figure 7.11(a) and Figure 7.11(b), respectively.
Figure 7.11: Team structure and busyness levels in terms of team performance

Figure 7.11 shows that there is no distinct pattern in the effects of busyness on team performance across the team structures. It was also observed in Table 7.1 that busyness level does not necessarily have a significant effect on the team performance, Table 7.1. These findings provide no clear evidence to support or reject hypothesis 8.

If the task is routine, the differences in the team performance, with the increase in the busyness level, is significant in the teams organized as sub-teams and the flat teams with social cliques, but not significant in the flat teams \( F=2.149 (P=0.0618) \). Contrary to this, for the non-routine tasks, the differences in the team performance, with the increase in the busyness level, is significant in the flat teams \( F=3.846 (P=0.002) \), and not significant in the flat teams with social cliques \( F=1.555 (P=0.175) \) or the teams organized as task-based sub-teams \( F=1.674 (P=0.1434) \).

7.2.5 Team structure, busyness level and TMM formation

It was hypothesized (hypothesis 9) that the decrease in the amount of TMM formation, with the increase in the busyness level, should be highest when the team is flat, lower when the team is flat but grouped into social cliques, and lowest when the team is organized into task-based sub-teams.

Therefore, in these experiments, team structure and busyness level were the independent variables, and the level of TMM formation was measured. For each team structure, simulations were conducted with the different busyness levels. The results from the experiments with the routine tasks are shown in Figure 7.9. The results for similar experiments with the non-routine tasks are shown in Figure 7.12.

---

\[34\] For sub-teams, \( F=5.5380 (P<0.001) \). For social cliques, \( F=3.7025 (P=0.003) \). See Table 7.1.
Findings from the simulations with both the routine tasks (Figure 7.9) as well the non-routine tasks (Figure 7.12) support hypothesis 9. However, this primarily relates to the amount of TMM formation, and does not suggest anything about the importance of the TMM or the efficiency of TMM formation, which is found to be exactly opposite, as shown in Figure 7.8.

### 7.2.6 Team structure, team familiarity and team performance

It was hypothesized (hypothesis 10) that the increase in the team performance, with the increase in the level of team familiarity, is highest when the team is organized into task-based sub-teams, lower when the team is flat, and lowest when the team is flat but grouped into social cliques.

Therefore, in these experiments, team structure and level of team familiarity were the independent variables, and the team performance was measured. For each team structure, simulations were conducted with the different levels of team familiarity. The results from the experiments with the routine tasks and the non-routine tasks are shown in Figure 7.13(a) and Figure 7.13(b), respectively.

**Figure 7.12: Team structure and busyness levels in terms of level of TMM formation (Non-routine task)**

**Figure 7.13: Team familiarity and team structure in terms of team performance**
Figure 7.13(a) shows that the increase in team performance with increasing team familiarity is lowest for task-based sub-teams and comparable for flat teams and flat teams with social cliques. Figure 7.13(b) shows similar patterns. Thus, the findings reject hypothesis 10.

This hypothesis was based on the expectation that the efficiency of TMM formation, and, hence, the efficiency of the pre-developed TMM will be greater in the teams organized as task-based sub-teams, lower in the flat teams, and lowest in the flat teams with social cliques. However, while the efficiency of TMM formation is greater in the teams organized as task-based sub-teams, the efficiency of TMM formation is found to be lowest in flat teams rather than in the flat teams with social cliques (Section 7.2.3).

The scope of improvement in the team performance in the task-based sub-teams is lower. The teams organized as task-based sub-teams take considerably fewer messages to complete the task, as compared to the flat teams or the flat teams with social cliques. Further, even at lower levels of TMM formation, the team performance in task-based sub-teams does not decrease as much as it does for the flat teams or the flat teams with social cliques because the efficiency of TMM is significantly higher in the task-based sub-teams, ( section 7.2.3, Figure 7.8, Table 7.7 and Table 7.8).

Given these results, it was conjectured that important TMM (know who to allocate a task) may be a better indicator of team performance rather than the overall TMM. To analyze these, a correlation analysis was conducted for team performance and overall TMM, and for team performance and important TMM formation, as shown in Table 7.9.

The results in Table 7.9 show that the correlation of the overall TMM formation and the team performance is generally higher than or comparable to the correlation of the important TMM and the team performance. This can be explained in terms of the importance of elimination process. While knowing who to allocate tasks is critical, in the exploratory stage of task allocation, it is also useful to know who not to consider for task allocation. Elimination reduces the search space, increasing the team performance. Importance might still be a useful measure, but the determination of what is important, needs careful assessment.

Table 7.9: Comparison of correlation of TMM and important TMM with team performance (Routine task, across BL=0, 25, 33, 50, 66, 75)

<table>
<thead>
<tr>
<th>LM</th>
<th>TS</th>
<th>Correlation (TP-TMM)</th>
<th>Correlation (TP-important TMM)</th>
</tr>
</thead>
<tbody>
<tr>
<td>PI+IO+TO</td>
<td>Flat</td>
<td>0.9165</td>
<td>0.5475</td>
</tr>
<tr>
<td></td>
<td>Social Cliques</td>
<td>0.9424</td>
<td>-0.0087</td>
</tr>
<tr>
<td></td>
<td>Sub-teams</td>
<td>0.8293</td>
<td>0.8683</td>
</tr>
<tr>
<td>PI+TO</td>
<td>Flat</td>
<td>0.9364</td>
<td>0.8602</td>
</tr>
<tr>
<td></td>
<td>Social Cliques</td>
<td>0.6224</td>
<td>0.5824</td>
</tr>
</tbody>
</table>
Thus, TMM formation mediates team performance within a given team structure. Across the team structures, efficiency of TMM formation varies. Hence, the teams with lower level of TMM formation (e.g. in task-based sub-teams) may still perform better than the teams with higher level of TMM formation (e.g. in flat teams).

### 7.3 Social learning and task types:

In the experiments discussed above, all the five independent variables, i.e., learning modes, busyness, team familiarity, team structure, and task types, have been considered. However, in all these experiments task type was considered the contingency factor and its effects were not explicitly studied. In the results discussed in this section, task types are considered as the central parameter, and the results are analyzed in terms of the effects of task types on TMM formation and the team performance.

#### 7.3.1 Task types, learning modes and team performance

It was hypothesized (hypothesis 11) that the decrease in the team performance, with the reduction in the number of learning modes, is greater when the teams are working on the routine tasks as compared to the teams working on the non-routine tasks.

Therefore, in these experiments, team structure and learning modes were the independent variables, and the team performance was measured. For each case of learning modes, simulations were conducted with the different team structures. The results from the experiments with the routine tasks and the non-routine tasks at 100% team familiarity are shown in Figure 7.14.

![Figure 7.14: Task types and learning modes in terms of team performance](image-url)
Figure 7.14 illustrates that when the teams are flat or flat with social cliques the decrease in team performance with decrease in learning modes is greater for routine tasks. These results are consistent with the hypothesis. However, for the task-based sub-teams the effects of task types are opposite to the hypothesis, as shown earlier in Table 7.6. Therefore, the findings partially support hypothesis 11. The validity of the assertion in hypothesis 11 is contingent on team structure.

7.3.2 Task types, busyness level and team performance

It was hypothesized (hypothesis 12) that the decrease in the team performance, with the increase in the busyness level, is greater for the teams working on the routine tasks as compared to the teams working on the non-routine tasks.

Therefore, in these experiments, task type and busyness level were the independent variables, and the team performance was measured. For each task type (routine and non-routine), simulations were conducted with the different busyness levels. The results from the experiments with flat teams that have all modes of learning available to the agents, and 100% team familiarity are shown in Figure 7.15. For the non-routine tasks, the effects of team familiarity on the team performance are only observed at 100% team familiarity. Hence, a comparison across the routine tasks and the non-routine tasks is conducted at 100% team familiarity only.

![Busyness Levels and Team Performance](image)

**Figure 7.15: Busyness levels and team performance for different task types**

Figure 7.15 shows the results for the experiments with the flat teams, but the patterns were comparable for the experiments with the flat teams with social cliques as well as for the teams organized into task-based sub-groups. This is opposite to the hypothesis. However, busyness levels do not necessarily have a significant effect on the team performance, Table 7.1. Therefore, hypothesis 12 is only partially rejected.
Validity of the hypothesis is contingent on the team structure. The hypothesis is valid for the task-based sub-teams and flat teams with social cliques. However, the opposite generally holds true for flat teams, Table 7.1.

7.3.3 Task types, busyness level and TMM formation

It was hypothesized (hypothesis 13) that the decrease in the level of TMM formation, with the increase in the busyness level, is greater for the teams working on the routine tasks as compared to the teams working on the non-routine tasks.

Therefore, in these experiments, task type and busyness level were the independent variables, and the level of TMM formation was measured. For each task type (routine and non-routine), simulations were conducted with the different busyness levels. The results from the experiments with the teams that have all modes of learning available to the agents are shown in Figure 7.16.

![Figure 7.16: Busyness levels and level of TMM formation for different task types](image)

Figure 7.16 does not show a very distinct pattern in the slopes for either task types. Hence, Table 7.10 summarizes the ANOVA results that compare the effects of busyness levels on TMM formation for the given task types. For example, the results in the first row show that if the task is routine, learning mode=PI+IO+TO, and team structure=Flat, the effects of busyness levels on TMM formation is F=66.52. This is lower than the corresponding value for non-routine tasks (F=78.06).

The results in Table 7.10 show that when all social learning modes are available to the agents, the decrease in the formation of TMM, with the increase in the busyness level, is greater for the teams working on the non-routine tasks. However, the results are not consistent for the experiments with the partial learning modes (PI+IO and PI+TO). The patterns are opposite for some cases. These findings partially reject hypothesis 13.

<table>
<thead>
<tr>
<th>Task</th>
<th>LM</th>
<th>F (P)-Flat team</th>
<th>F (P)-Social clique</th>
<th>F (P)-Sub-team</th>
</tr>
</thead>
<tbody>
<tr>
<td>R</td>
<td>PI+IO+TO</td>
<td>66.52 (&lt;0.0001)</td>
<td>22.97 (&lt;0.0001)</td>
<td>13.91 (&lt;0.0001)</td>
</tr>
</tbody>
</table>
7.3.4 Task types, team familiarity and team performance

It was hypothesized (hypothesis 14) that the rate of increase in the team performance, with the increase in the level of team familiarity, is higher for the teams working on the non-routine tasks than that for the teams working on the routine tasks.

Therefore, in these experiments, task type and level of team familiarity were the independent variables, and the team performance was measured. For each task type (routine and non-routine), simulations were conducted with the different levels of team familiarity. The results from the experiments with flat teams that have all modes of learning available to the agents are shown in Figure 7.17.

![Figure 7.17: Team familiarity and team performance for different task types](image)

In Figure 7.17, a comparative trend analysis of the results for the routine tasks and the non-routine tasks across the different team structures shows similar patterns. At higher levels of team familiarity, the slope for non-routine tasks is greater, while at lower levels of team familiarity the slope for routine tasks is greater. These results partially support hypothesis 14.

As was discussed in section 7.1.3, the increase in the team performance, with the increase in the level of team familiarity, is greater at higher levels of team familiarity. As the task complexity increases (moving from routine to non-routine tasks), for the team familiarity to enhance the team performance, the required level of team familiarity tends to move closer to TF=100%. Thus, discounting for the lower levels of team familiarity, the findings conform to hypothesis 14.
However, at lower levels of team familiarity and the routine tasks, there is significant increase in the team performance, with the increase in team familiarity \([F=15.606 \ (P<0.0001)\) and correlation \((TP/TF=0.9773, \text{Table 7.3})]\). For the non-routine tasks, at the lower levels of team familiarity, there is no noticeable increase in the team performance, with the increase in the levels of team familiarity \([F=0.377 \ (P=0.770)\) and correlation \((TP/TF)=0.1803, \text{Table 7.3})\]. Hence, the findings only partially support hypothesis 14.

7.3.5 Task types, team structure and team performance

It was hypothesized (hypothesis 15) that the relative difference in the team performance across the different team structures is higher when the teams are working on the non-routine tasks, as compared to the teams working on the routine tasks.

Therefore, in these experiments, task type and team structure were the independent variables, and the team performance was measured. For each task type (routine and non-routine), simulations were conducted with the different team structures. The results from the experiments with busyness level=0%, and the teams that have all modes of learning available to the agents, are shown in Figure 7.18.

![Team Structure and Team Performance](image)

**Figure 7.18: Team structure and team performance for different task types**

In Figure 7.18, the results are shown for the different levels of team familiarity. In each case, where the differences in the team performance are observed across the team structures, the differences are greater if the task is non-routine\(^{35}\). These findings support hypothesis 15.

---

\(^{35}\) For routine tasks, the differences across team structures at TF=17, 33, 50, 66 and 83% are F=100.686, 182.567, 114.846, 66.223 and 25.697 respectively. For non-routine tasks, the differences across team structures at TF=17, 33, 50, 66 and 83% are F=239.943, 312.891, 309.000, 374.000 and 314.129 respectively.
The team performance is highest for the teams organized as task-based sub-teams, lower for the flat teams, and lowest for the flat teams with social cliques. It is conjectured that the efficiency of TMM formation in task-based sub-teams (Figure 7.8), is the primary factor for the difference in the team performance. Even at the lower levels of team familiarity, the teams organized as task-based sub-teams perform better than the flat teams or the flat teams with social cliques, because in the task-based sub-teams, the search for a relevant expert is narrowed down to fewer team members. In the flat teams with social cliques, the agents can only observe their group members but the related task experts may belong to other social cliques. This explains the marginal difference in performance across the flat teams and flat teams with social cliques. However, the differences in the performances across the flat teams and the flat teams with social cliques are not significant \( e.g. F=0.0459, 0.2538, \ etc \), except for TF=100% and the non-routine tasks, in which case, \( F=11.523 \) (\( P=0.0009 \)).

### 7.3.6 Task types, team structure and TMM formation

It was hypothesized (hypothesis 16) that the relative difference in the level of TMM formation across the different team structures is higher when the teams are working on the routine tasks as compared to the teams working on the non-routine tasks.

Therefore, in these experiments, task type and team structure were the independent variables, and the level of TMM formation was measured. For each task type (routine and non-routine), simulations were conducted with the different team structures. The results from the experiments with the teams that have all modes of learning available to the agents are shown in Figure 7.19.

![Figure 7.19: Team structure and level of TMM formation for different task types](image)

Figure 7.19 shows that the relative difference in TMM formation across the flat teams and the teams organized into task-based sub-teams are greater for the teams working on the routine tasks.
However, the relative difference in TMM formation for the teams organized into task-based sub-teams and the flat teams with social-cliques, show marginal difference across the task types. Therefore, the findings partially support hypothesis 16.
Chapter 8
Conclusions, limitations and future work

This concluding chapter reviews the research outcomes. A summary of the results for the tested hypotheses is presented, followed by a discussion on the limitations of this research and possible future work.

8.1 Review of research objectives

The aim of this research was to explore the role of social learning in the formation of TMMs and team expertise using a computational test-bed. Towards this aim, one of the main objectives was to develop a computational model for investigating the various research hypotheses stated in Chapter 3, in relation to the formation of TMMs and the team performance. TMM was computationally represented as an $m \times n$ matrix, where $m$ is the total number of tasks that the team needs to perform, and $n$ is the total number of agents in the team. The element in the $i$th row and $j$th column of the matrix stores the details of the capability of the $j$th agent in the $i$th task (Section 5.3.2, Section 5.4.2). Each agent starts with a default TMM of the team, and as the agents interact with or observe the other agents and the task performance, the corresponding values in the matrix are updated (Section 5.3.3, Section 55.4.3), thereby developing the TMM. The following research objectives were identified and achieved:

Development of the conceptual framework

A conceptual framework of social learning in teams and formation of TMM was developed (Chapter 4). Adopting the folk theory of mind (Knobe & Malle, 2002; Malle, 2005; Ravenscroft, 2004; Tomasello, 1999) as the conceptual underpinning allows a discrete representation of the different social learning modes that are differentiated as: (1) learning from personal interactions, (2) learning from task
observations, and (3) learning from interaction observations. Agents’ social learning abilities depend on the opportunities for social interactions and observations available to them. To explore the influence of variations in the social learning opportunities on TMM formation, two factors each at agent level and team level were included in the framework. Factors affecting social learning at the agent level are: (1) Learning modes available to the agents, and (2) Agents’ busyness levels. Factors affecting social learning at the team level are: (1) Team structure, and (2) Levels of team familiarity. The four factors, together with the task types, are the five independent variables.

The literature (Kraiger & Wenzel, 1997; Langan-Fox et al., 2004; Lim & Klein, 2006; Rouse et al., 1992) suggests that TMM mediates team performance. Hence, in order to explore how this correlation is affected by the different learning modes, levels of TMM formation and team performance are taken as the dependent variables. Based on the available literature (Edmondson, 1999; Griffin, 1996), the reduction in team communication (number of messages) is taken as the indicator of the increase in team performance.

**Implementation and validation of the computational model**

The conceptual framework is implemented as a multi agent system in JADE (Chapter 5). The implemented system allows: (1) simulations with the various independent variables separately as well as with different superposed combinations (section 6.2), (2) accurate and complete extraction of agents’ TMM in human readable form (Section 5.3.2, Section 5.4.2), and (3) measurement of the team performance by maintaining a log of the messages exchanged between the agents.

The implemented system is flexible and scalable. Agents’ learning is implemented (Section 5.5) as rules based on the folk theory of mind (Ravenscroft 2004; Malle, 2005; Malle, 1997; Tomasello, 1999; Knobe, 2002) and the attribution theory (Wallace, 2009; Irene Frieze, 1971; Iso-Ahola, 1977; Jones, 1958). This rule-based approach to learning can be enriched by addition of new rules.

Preliminary experiments were conducted to validate the model (section 6.1) using comparable scenarios for which results are available in the literature (Moreland et al., 1998; Ren et al., 2006). The findings from these validation simulations conform to the earlier findings (Moreland et al., 1998; Ren et al., 2001; Ren et al., 2006), which prove the validity of the model as a simulation tool (section 6.1).

**Investigation of the research hypothesis**

Experiments were conducted (section 6.2) to test the 16 research hypotheses proposed in Chapter 3. Most of the research hypotheses proposing the different correlations for social learning modes, team structures, levels of team familiarity, busyness levels, and the task types, in terms of the levels of TMM formation, are supported by the experiment results (section 6.2, Chapter 7). However, only some of the
hypotheses stating correlations for social learning modes, team structures, levels of team familiarity, busyness levels, and the task types, discussed in terms of the team performance, are supported (Chapter 7). A summary of the research findings in terms of the research hypotheses is presented in section 8.2.

The results validate the research’s main hypothesis that the modes of social learning have a statistically significant effect on TMM formation (section 7.1.2). However, the results show that the busyness levels and team structure also have a significant effect on TMM formation (section 7.1.2 and section 7.2.2). Learning from task observations has a greater contribution to increasing amounts of TMM formation than learning from interaction observations (section 7.1.2).

In general, the results support the earlier findings that (1) social learning enhance TMM formation and the team performance (Moreland et al., 1998; Ren et al., 2006; Conlon, 2004) (2) TMM mediates team performance (Kraiger & Wenzel, 1997; Langan-Fox et al., 2004; Lim & Klein, 2006; Rouse et al., 1992) (section 7.2.6), (3) Team performance increase with the increase in levels of team familiarity (Harrison et al., 2003; Hinds et al., 2000; Huckman et al., 2008) (section 7.1.3), (4) Team familiarity has greater positive affect on team performance if the task is routine (Huckman et al., 2008) (section 7.1.3), and (5) Busyness reduces the levels of TMM formation (Cramton, 2001; Driskell et al., 1999; Gilbert & Osborne, 1989) (section 7.1.2). The results also show that though TMM mediates team performance, higher TMM formation may not always indicate high team performance. The efficiency of TMM formation varies across the team structures such that TMM formation is more efficient in the teams organized as task-based sub-teams as compared to the flat teams (section 7.2.3). Within a given team structure, the level of TMM formation is correlated with the team performance (section 7.2.6). Findings suggest that in general, busyness levels have no significant effect on the team performance (section 7.1.1, section 7.1.4, and section 7.2.4) but they significantly affect the level of TMM formation (section 7.1.2, and section 7.2.5).

These findings will be useful for design team managers in deciding the team composition (level of familiarity), work loads (busyness level), and the team structure, contingent on the nature of the design task, the available technical support for social interactions and observations (social learning) in distributed teams, and the project goals. For example, if it is a long-term term project, where time is not a major constraint in the initial phase, then the project team can initially be organized as flat teams to facilitate higher levels of TMM formation. At later stages of the project, the team can be re-organized into task-based sub-teams to enhance the team performance. Similarly, if an organization intends to hire new employees, it might be a better option to introduce them into the project teams working on routine tasks. Even if team familiarity levels reduce, in the teams working on routine tasks, the decrease in the team performance is gradual. The decrease in the team performance, with the decrease in team familiarity, is much steeper for the project teams working on non-routine tasks. Thus, once the
new employees have developed prior-acquaintance with the other employees, while working on the projects involving routine tasks, they can be inducted in to the projects involving non-routine tasks.

8.2 Summary of results

Table 8.1 lists the research hypotheses and comments on the findings from the experiments.

<table>
<thead>
<tr>
<th>Hypotheses</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>H1</td>
<td>When compared to the teams that have all modes of learning available to the agents, the decrease in team performance, with the increase in busyness levels, is lower in the teams that have partial modes of learning available to the agents. The decrease in team performance, with the increase in busyness levels, is lowest for the teams in which the agents learn only from personal interactions.</td>
</tr>
<tr>
<td>H2</td>
<td>When compared to the teams that have all modes of learning available to the agents, the decrease in levels of TMM formation, with the increase in busyness levels, is lower in the teams that have partial modes of learning available to the agents. The decrease in levels of TMM formation, with the increase in busyness levels, is lowest for the teams in which the agents learn only from personal interactions.</td>
</tr>
<tr>
<td>H3</td>
<td>When compared to the teams that have all modes of learning available to the agents, the increase in team performance, with the increase in levels of team familiarity, is lower in the teams that have partial modes of learning available to the agents. The increase in team performance, with the increase in levels of team familiarity, is lowest for the teams in which the agents learn only from personal interactions.</td>
</tr>
</tbody>
</table>
agents learn only from personal interactions. on team performance are significant, only, if the team familiarity level is close to 100% (section 7.1.3).

**H4** The increase in team performance, with the increase in team familiarity, is higher when busyness levels are lower.

**H4** is partially rejected.

Busyness levels influence the rate of change in team performance, with the increase in team familiarity, only, if team familiarity is higher (> 50%), and, if the team has not reached near optimal performance, e.g., the teams with 100% team familiarity, low busyness levels (< 50%), and working on routine tasks (section 7.1.4).

**H5** The increase in team performance, with the increase in the number of modes of social learning, is highest when the team is organized into task-based sub-teams, lower when the team is flat and lowest when the team is flat but grouped into social cliques.

**H5** is partially rejected.

The relative role of the different learning modes on team performance, across different team structures, is contingent on the task type. The hypothesis is valid if the task is non-routine, and the opposite is true if the task is routine (section 7.2.1).

**H6** The increase in levels of TMM formation, with the increase in the number of modes of social learning, is highest when the team is flat, lower when the team is flat but grouped into social cliques, and lowest when the team is organized into task-based sub-teams.

**H6** is supported (section 7.2.2).

**H7** When all modes of social learning are available to the agents, the increase in the efficiency of TMM formation is highest when the team is organized into task-based sub-teams, lower when the team is flat, and lowest when the team is flat but grouped into social cliques.

**H7** is modified to (section 7.2.3):

*When all modes of social learning are available to the agents, the increase in the efficiency of TMM formation is highest when the team is organized into task-based sub-teams, lower when the team is flat but grouped into social cliques, and lowest when the team is flat.*

**H8** The decrease in team performance, with the increase in busyness levels, is highest when the team is organized as task-based sub-teams.

**H8** is neither supported nor rejected, as no clear pattern in results is observed.

For some cases, busyness level has no significant
lower when the team is flat, and lowest when the team is flat but grouped into social cliques.

effect on the team performance. The results are mixed in other cases (section 7.2.4).

**H9** The decrease in the amount of TMM formation, with the increase in busyness levels, is highest when the team is flat, lower when the team is flat but grouped into social cliques, and lowest when the team is organized into task-based sub-teams.

**H9** is supported (section 7.2.5).

**H10** The increase in team performance, with the increase in team familiarity, is highest when the team is organized into task-based sub-teams, lower when the team is flat, and lowest when the team is flat but grouped into social cliques.

**H10** is rejected (section 7.2.6).

**H11** The decrease in team performance, with the reduction in the number of learning modes, is greater for the teams are working on routine tasks as compared to the teams working on non-routine tasks.

**H11** is partially supported.

The validity of this hypothesis is contingent on the team structure. The hypothesis is valid for the flat teams and the flat teams with social cliques. However, the opposite is true for the task-based sub-teams (section 7.3.1).

**H12** The decrease in team performance, with the increase in busyness levels, is greater for the teams working on routine tasks as compared to the teams working on non-routine tasks.

**H12** is partially rejected.

The correlation of task types, busyness levels and team performance is contingent on the team structure (section 7.3.2). The hypothesis is valid for the task-based sub-teams and flat teams with social cliques. However, the opposite generally holds true for flat teams, Table 7.1

**H13** The decrease in levels of TMM formation, with the increase in busyness levels, is greater for the teams working on routine tasks as compared to the teams working on non-routine tasks.

**H13** is partially rejected.

Patterns vary across the task types and show opposite trends (section 7.3.3).

**H14** The rate of increase in team performance, with the increase in team familiarity, is higher for the

**H14** is supported.

However, at lower levels of team familiarity, the
teams working on non-routine tasks than that for the teams working on routine tasks. pattern is ambiguous, with mixed results. But, since for non-routine tasks, team familiarity has weak correlation with team performance at lower levels of team familiarity, these results can be safely ignored (section 7.3.4).

**H15** The relative difference in team performance across the different team structures is higher for the teams working on non-routine tasks as compared to the teams working on routine tasks. **H15** is supported (section 7.3.5).

**H16** The relative difference in levels of TMM formation across the different team structures is higher for the teams are working on routine tasks as compared to the teams working on non-routine tasks. **H16** is partially supported. The relative difference in TMM formation across the teams organized into task-based sub-teams and flat teams with social-cliques show marginal difference across the task types (section 7.3.6).

### 8.3 Strengths and limitations

The strength of this research is the simplification of the experimental scenarios. Through a computational model, representing the agents’ TMM in a matrix form (Section 5.3.2, Section 5.4.2), this study focuses specifically on TMM (other mental models for task, process and context are assumed to be well-developed). The social learning modes are distinctly identified and represented, using simple rules (Section 5.5, Table 5.4). Only a few variables (team structure, levels of team familiarity, busyness levels, learning modes) are considered (Table 6.6). The use of a computational method ensures controlled experiments that facilitate data collection and analysis (TMM formation, number of messages). The conformity of the results from the validation simulations, to the literature (section 6.1), suggests that this computational model of TMM and social learning can provide useful insights into the theories of team building and team performance.

However, the simplified model is also the main limitation of this research. Currently, in this model, the knowledge of agent’s intentionality is perfect, i.e., if an agent refuses to perform a task, it is assumed that it does not know how to perform the task. These assumptions are true in these simulations because if an agent can perform a task, it does. Similar assumptions and modelling decisions ensure that there are no errors in the agents’ learning. However, in the real world scenario, other factors such
as trust and motivation may influence an agent’s willingness to perform a task such that even if an agent knows how to perform a task, it may refuse to do so.

Also, in this research, the modes of learning are based on personal interactions, task observations and interaction observations. The results are discussed in terms of their relative contributions to the level of TMM formation and the team performance. However, personal interactions in real world scenarios may include interactions such as recommendations (informing an agent about another agent’s competence) and query (asking an agent about another agent’s competence), where agents explicitly exchange information about the other agents. Such interactions have not been included in the simulations reported in this research. Similarly, only formal interactions have been modelled in this research. However, informal interactions are critical to social learning in team environments (Bobrow & Whalen, 2002; Borgatti & Cross, 2003). Thus, variations in results can be expected if these additional interactions are included in the model.

Another important aspect that may affect the agents’ learning is the agent architecture. Agents in this model remember what they have learnt. Additionally, the task related capabilities of agents do not change over time, i.e., the mental models for task, process and context are assumed to be fixed. This is a narrow view of the world, which is dynamic and changing. Since the focus of this research was to explore the relative contributions of each of the learning modes to TMM formation, these modelling decisions were not critical, but they may influence the results. For example, it is possible that the capability of an agent in performing a task may reduce if it has not performed that task for a long time. Similarly, agents may learn to perform a task by observing the other agents perform that task. However, such learning capabilities are dependent on the task complexity and the agent architecture.

Thus, incorporating such changes in the model would require cognitively richer agents, with attributes such as short term memory, recency, constructive memory, and so on, which may change the way an agent learns and uses its past interactions and observations, to adapt to new situations. A cognitively rich agent will be required if the agents are to recognize and learn patterns in the team. This would allow agents to make generalizations about the typical agents in the team. For example, in this model, the typical solution span of the capability range of an agent (section 5.4.3), (e.g. MinWindow=3, MaxWindow=5) was pre-coded as generalized knowledge. Thus, when an agent A\textsuperscript{1} observes a solution provided by agent A\textsuperscript{2}, it can narrow down the possible solution space that it can expect from agent A\textsuperscript{2}. However, if the agent is cognitively rich, rather than needing a pre-coded solution span, it may recognize a pattern in the solutions provided by all the agents, to learn the solution span of a typical agent. Similarly, it is likely that some of the tasks are related such that if an agent can perform a task T\textsuperscript{1}, it may also be able perform the task T\textsuperscript{2}. If agents can learn and identify such patterns, their task allocation capabilities and TMM formation may influence the results.
However, in order to model and test these characteristics, modifications may also be required to the simulation conditions, such as the number of agents, number of tasks, number of training runs, and input data (i.e., ensure that there is a pattern in the task competencies of the agents) so that there are enough training cases to learn from and generalize.

The results may vary with the complexity of the task modelled and the knowledge and coordination required by the agents. This model adopts one of the ways to represent a design task. The experiments reported in this research have been conducted with simple non-routine tasks. However, as discussed in section 4.1.6.2, this representation can be used to investigate more complex task environments, through consideration of weights and constraints.

Similarly, the simulation conditions were kept similar for all cases such that results are comparable, i.e., each simulation consisted of two simulation rounds, i.e., a test round and a training round. However, it is likely that for the teams working on non-routine tasks, the effects of team familiarity may change if the agents have experience of working together over multiple projects.

In the end, as with the other computational studies, these results indicate social behavioural patterns, and further investigations must be conducted in real world settings to determine their veracity.

8.4 Future research

This section discusses some of the planned future work. The section is divided into two segments. The first segment discusses the short-term extension plans. The second segment discusses the possible directions that can be adopted for long term research, including the field studies that may be conducted in real world scenarios.

8.4.1 Short-term extension

The short-term extensions to the research are related to the details of the computational model. The following changes are planned towards the enhancement of the model:

*Measuring sharedness of TMM formation*

As discussed in section 5.3.4, TMM formation is measured in terms of the amount (density) of TMM. Measuring accuracy was not required because all that the agents learn is correct. The other measures of TMM that have been analyzed are importance and efficiency. However, sharedness (commonality) is another possible measure of TMM that has not been analyzed. Sharedness was not measured because the TMM is defined as the aggregate of the TMMs maintained and formed separately by each agent. For task allocations, the agents use their own TMM to identify the relevant experts, and since expertise
is explicitly distributed across the agents, sharedness is not needed for task performance. However, the analysis of sharedness may still provide useful insights. For example, let us consider the following scenario. Both agent A\(^1\) and agent A\(^2\) can perform a task T\(^1\). Another agent A\(^3\) is the only agent that can perform a task T\(^2\), which immediately follows T\(^1\). Thus, if a new team is formed, it is possible that either A\(^1\) or A\(^2\) get to perform T\(^1\). Hence, the team is likely to perform better if both A\(^1\) and A\(^2\) know about A\(^3\)'s competence in T\(^2\). However, it does not matter whether both A\(^1\) and A\(^3\) know about A\(^2\)'s competence in T\(^1\), because that is unlikely to affect the team performance. Therefore, even if A\(^1\) and A\(^2\) do not explicitly exchange the information about A\(^3\)'s capability to perform T\(^2\), they can learn so by observing the other allocating the task to T\(^2\). Hence, even though sharedness is not required for task performance, sharedness may be useful for task allocation, in few of the cases, where more than one agent has the competence in the same task.

Thus, measuring sharedness will be useful but such an analysis needs to be selective, i.e., sharedness needs to be analyzed only for the agents that have common competence and not across the entire team, which otherwise may show redundancy in results.

**Analyzing group effects**

For the teams organized into groups, the pattern of TMM formation may differ across the groups. In future research, it may be interesting to study (1) the group effect, i.e., comparison of in-group TMM with non-group TMM, and (2) how the re-organization of the groups may effect TMM formation and the team performance.

**Enrichment of agent learning**

At present the rule base is small. New rules can be added to the rule base to enhance the agent capabilities. For example, even if an agent cannot observe the response of an agent A\(^1\) to an allocated task T\(^1\), but at some later cycle in the same project the agent observes the same task T\(^1\) being allocated to another agent A\(^2\), then the observer can infer that A\(^1\) could not provide an acceptable solution for T\(^1\), because T\(^1\) is being reallocated.

In order for these kinds of rules to be added, the past experience, i.e., A\(^1\) was allocated the task T\(^1\), should be retained in the observer agent’s memory. Moreover, the current sense data (Task T\(^1\) is allocated to A\(^2\)) should trigger the recall of that experience. Hence, for these kinds of learning to take place, characteristics such as short term memory, recency, pattern matching, and memory recall need to be implemented. Hence, it is planned to use a cognitively rich agent in the future simulations.
Implementing busyness as cognitive process

Once a cognitively rich agent is used, busyness can be implemented as a Monte Carlo variable such that it is part of the cognitive process, and is not an externally specified parameter that is same for all the agents. Accordingly, it might be possible to incorporate an inverse relationship of busyness with the levels of personal interaction or task performance, i.e., if an agent has higher number of personal interactions or task performance, it should have fewer opportunities for social observation because it is pre-occupied with its own activities. For this, it may be useful to run simulations where the agents are simultaneously part of multiple projects such that busyness can be related to engagement with the other activities that are not related to the current project. This has been assumed in this research as well.

8.4.2 Long-term extension

In the long term, this computational model can be developed along different directions depending on what aspects of TMM are being investigated. In any case, it will be useful to include informal interactions and additional learning modes (e.g. instructional learning, explicit information seeking about other agents and tasks, etc). Some of the possible directions are discussed below:

Real world studies with design teams for collecting social interaction data

Real world field studies can be conducted to develop a taxonomy of design actions and communicative terminologies used in social interactions in the design teams. For example, Milne and Leifer (2000) classify information handling activities in the design teams into six broad categories namely, generate, access, analyze, elaborate, verify and navigate. This kind of classification can allow modelling detailed design activities. Using further real world studies, social interaction data can be collected to observe how the team members reason about and update their mental models of each other, in terms of these activities. Thus, TMM formation in the design teams can be studied in greater detail, in terms of the design and communicative actions.

This approach can build on similar work reported in formalization of folk theory for use in agent-based modelling (Gordon & Hobbs, 2004; Hobbs & Gordon, 2005), and prior-work on learning styles in design teams (Carrizosa & Sheppard, 2000; Milne & Leifer, 2000).

Focus on interaction between the different mental models

As discussed earlier (section 8.3), one of the limitations of this study is to assume that the mental models for the task, process and context are fixed. Now that the effects of TMM on team performance have been studied with these constraints, i.e., without the influence of task, process or context mental models, it will be useful to implement scenarios where the agents learn about the task, process and the
context mental models, in addition to the TMM. In the real world scenario, all the different types of mental models develop over time. Hence, the role of TMM on team performance may be influenced by changes in the other mental models. Thus, a study in which the agents learn about the process and context will provide an understanding of the correlation of the different mental models, i.e., TMM, task mental model, process mental model and context mental model, and how this correlation is affected by the available learning modes.

Focus on the social attributes of the agents

Social attributes are innate characteristics of an agent, which may influence the agent’s cognitive abilities. Factors such as motivation (Harvey, 1963; Mitchell, 1982; Osterloh & Frey, 2000), curiosity (Berlyne, 1966; Renner, 2006), trust (Dirks & Ferrin, 2001; LaPorta et al., 1997), power relationships (Ashforth & Mael, 1989; Emerson, 1976; Thye, 2000), group threshold (social tipping) (Granovetter, 1978), social ties (Granovetter, 1973; Krackhardt, 1992), and so on determine the agent’s social and cognitive behaviour. It would be useful to include the social attributes of the agents as part of the TMM. This study will require the agent behaviour to be influenced by the social attributes such as trust and motivation so that the agents reason about each other in those terms.

For example, if an agent $A^2$ recommends agent $A^1$ to agent $A^0$, then how much confidence $A^0$ will have in the competence of $A^1$ will depend on how much $A^0$ trusts $A^2$. Similarly, the agents will seek information from the agents that they trust. Thus, trust may determine how the agents use each others’ TMM to modify or update their own TMM, which in turn may influence the sharedness of TMM across the team members. TMM formation, influenced by trust, may also result in biases for the task allocation, thereby affecting the team performance.

Similarly, in a team environment, individual’s actions may be influenced by the group decision. Hence, direct inferences from an agent’s action to an allocated task may not be possible in all the cases. For example, if an agent is allocated a task $T^1$, in private (individually), it may provide a solution $S^1$. However, if the same agent is allocated the same task $T^1$, in public, it may provide a solution $S^0$ if it observes that all (or majority, given by some threshold) the other agents have proposed the solution $S^0$. Therefore, the agents may need to reason about another agents action to decide whether the action of the other agent was influenced by the group decision or not. This would require the maintenance and update of a detailed TMM that maps these causal relationships.

Since most of these social behavioural attributes have been described or modelled separately in various works (Brazier & Wijngaards, 2002; Cascalho et al., 2006; Castelfranchi & Falcone, 2000; Conte & Castelfranchi, 1995; Goldstone & Janssen, 2005; Kaplan & Oudeyer, 2006; Kathryn, 2007; Norman & Long, 1995; Saunders & Gero, 2001, 2004), it may also be worth trying to include many of
these attributes together in a single model such that some attributes dominate the other attributes in different situations.

8.4.3 In the end

Social simulations have often generated sceptic remarks and criticism. During the course of this research, as I dealt with the tough choice of choosing the variables and making the modelling decisions, I have begun to appreciate some of these concerns but I have equally realized the potential of computational methods in advancing social theories, especially the “what if” scenarios that are difficult to control and simulate in real world studies. It was evident that the modelling decisions and assumptions are critical to the development of a valid computational model. Prior work (Axtell et al., 1996; Carley & Newell, 1994; Levitt et al., 2005) provided the guidelines and benchmark to assess the usability of this model. At this point, there are more questions than answers that I set out with at the start of this research, and it has been difficult to leave out some of the relevant details from the model that seemed interesting. However, there is only as much that one can do with the time constraints, and there is much more to be done.

In the end, as much as a steep learning experience, this research has been equally fulfilling and fun.
References


## Glossary

**A-**

<table>
<thead>
<tr>
<th>Term</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Agent</td>
<td>An agent is an autonomous entity that observes and acts in an environment (Russell &amp; Norvig, 2002). (section 2.4)</td>
</tr>
<tr>
<td>Accuracy</td>
<td>Accuracy is the measure of correctness of a team mental model, i.e., How much of what the agent knows about the other members of the team is correct. (section 2.2.2.1)</td>
</tr>
<tr>
<td>Agent mental model (AMM)</td>
<td>The knowledge about the competence of an agent in terms of the tasks to be performed by the team. Computationally, the AMM is represented as an $m$-dimensional vector, representing the competence of the agent in each of the $m$ tasks to be performed by the team. (section 5.3.2, section 5.4.2)</td>
</tr>
<tr>
<td>Agent management System (AMS)</td>
<td>The Agent Management System (AMS) is the default agent in JADE (Java Agent Development Environment) which exerts supervisory control over access to and use of the Agent Platform.</td>
</tr>
<tr>
<td>Aggregation</td>
<td>Aggregation is the TMM measurement technique that assumes that TMM is an aggregate of the mental models of individual agents, which can each be measured separately. (section 2.2.2.3)</td>
</tr>
<tr>
<td>Attribution theory</td>
<td>Attribution theory is concerned with the ways in which people explain the behaviour (e.g., failures and success) of others or themselves. (section 2.1)</td>
</tr>
<tr>
<td>Audience design</td>
<td>Audience design is the ability of the task performer to adapt solutions to suit the task allocator. Task performers develop a mental model of the task allocator, and they use this mental model of the task allocator to choose solutions that they expect to be acceptable to the task allocator. (section 4.1.3)</td>
</tr>
</tbody>
</table>

**B-**
<p>| <strong>BDI Agent</strong> | BDI agents are agents whose architecture is defined in terms of belief, desire and intentions. Beliefs are the agent’s knowledge about the environment, which may be incomplete or inaccurate. Desires are the agent’s objectives or goals, and intentions are the desires that the agent has committed to achieve. Plans are part of the belief that a particular action will lead to the desired goal. (2.4) |
| <strong>Busyness level</strong> | Busyness is the probability that an agent is not able to sense the observable data (interactions among other agents, and task-performance by some other agent), available at that instance. (section 4.1.4) |
| <strong>Capability range</strong> | Capability range is the range of solutions that an agent can provide for a given task, which it can perform. The capability range is defined by a lower and upper value. (section 4.1.6.2, section 5.4.3) |
| <strong>Client Agent</strong> | An agent that is not a part of the team, but interacts with the team to call for the initial project bid, nominate the team leader, and approve the overall solution. (section 5.7) |
| <strong>Cognitive agent</strong> | An agent that has the capability for recognition and categorization, decision making and choice, perception and interpretation, prediction and monitoring, problem solving and planning, reasoning and belief maintenance, execution and action, interaction and communication and remembering, reflection and learning (Langley et al., 2009). (section 2.4) |
| <strong>Common sense psychology</strong> | An alternative term used for the “Folk theory of mind”, a conceptual framework that explains social behaviour and mental states in terms of commonly used words such as actions, beliefs, intentions, observations, and so on. (section 2.1) |
| <strong>Competence</strong> | Measure of expertise of an agent in a given task. This is calculated as the ratio the number of times an agent performed a given task to the number of times the task was allocated to the agent. |
| <strong>Competence mental model</strong> | The shared understanding within a team about what it means to be competent. This is assumed to be known to all the agents. Therefore, the definition of “competence” is same for all the agents in this model. |
| <strong>Computational sociology</strong> | Study of social behaviour through computer simulations. |
| <strong>Context mental model</strong> | The understanding of how and what works for the team in a given context. In this research, the context mental model is pre-coded into the agents. |</p>
<table>
<thead>
<tr>
<th>Term</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Conspecific</td>
<td>Belonging to the same species, i.e., in this model, agents assume all other agents to be similar to themselves in their intentionality and actions.</td>
</tr>
<tr>
<td>Creative task</td>
<td>Non-routine tasks for which the solution space is not defined.</td>
</tr>
<tr>
<td>Critical task network</td>
<td>A network of agents (as identified by an external observer, e.g., experimenter) in the team who are connected such that each agent can perform one of the sub-tasks and knows who to allocate the resulting sub-task. Thus, the critical task-network can lead to optimum performance because each task allocation is informed and accurate. (section 5.10)</td>
</tr>
<tr>
<td>Docking</td>
<td>Docking is the equivalency test for two computational models used for similar social simulations. If results from similar simulations using the candidate computational model and the benchmark computational model are comparable, the candidate model is deemed valid. (section 2.3)</td>
</tr>
<tr>
<td>DF (Directory Facilitator) agent</td>
<td>The Directory Facilitator (DF) is the default agent in JADE which provides the default yellow page service in the platform.</td>
</tr>
<tr>
<td>Efficiency of TMM</td>
<td>Efficiency of TMM is measured as the ratio of the team performance to the level of TMM formation. (section 7.2.3)</td>
</tr>
<tr>
<td>Expertise distribution</td>
<td>The number of agents with expertise in a given task. For example, expertise distribution 4(2)3(1) means there are 4 such tasks for which there are 2 agents than can perform that task, and there are 3 such tasks for which there are only 1 agent each that can perform those tasks. (section 6.1)</td>
</tr>
<tr>
<td>Finite state machine</td>
<td>A model of behaviour composed of a finite number of states, transitions between those states, and actions. This computational model is implemented as a finite state machine with finite states for tasks, solutions, messages, actions and TMM.</td>
</tr>
<tr>
<td>Flat teams</td>
<td>Flat teams are teams with no hierarchy and no sub-divisions. Flat teams allow unrestricted access to all agents in the team for task allocations as well observations. (section 2.2.1)</td>
</tr>
<tr>
<td>Flat teams with social cliques</td>
<td>Flat teams distributed into social cliques. In flat teams with social cliques, agents can allocate tasks to any other agent in the team, but their ability to observe other agents is limited to members within their social cliques. (section</td>
</tr>
</tbody>
</table>
Folk theory of mind

A conceptual framework that relates different mental states to each other and connects them to behaviour. Folk theory explains social behaviour in terms of commonly used terms such as actions, observations, intentionality, beliefs, desires, and plans. (section 2.1)

FIPA protocol

Specifications to deal with pre-agreed message exchange protocols for Agent Communication Language (ACL) messages.

Fractionation matrix

An indicative matrix mapping the level of detail of agent capabilities and the environment complexity to provide a guideline for design of agent architecture based on the research questions to be investigated using the simulation environment (Carley & Newell, 1994).

G-

Generalization

Generalization is the ability of the agent to identify patterns, and learn the causal history of relationships between the enabling factors and the actions of a typical agent (Malle, 2005).

In this research, agents do not generalize. Hence, the causal relationships between enabling factors and actions are pre-coded. (section 4.1.3)

H-

Heterogeneous knowledge distribution

Knowledge distribution in a team such that each agent has specialized knowledge, i.e., each agent has competence in difference tasks. However, it is possible to have more than one agent to have competence in the same task. (section 2.2)

I-

Importance

Importance is the measure of the TMM that captures the central attributes of a task or team that may have a greater influence on team performance (Badke-Schaub et al., 2007). (section 2.2.2.1)

Intentionality

Intentionality is used to refer to actions that are intentional. In this research all actions of the agents are assumed to be intentional. Thus, in this research it is assumed that: (a) if an agent has the competence to perform a task, it will; (b) agents always intend to allocate a task to an agent that it expects to have the highest competence to do the task; and, (c) agents will refuse to do a task only if they do not have the competence to do it. (section 2.1, section 5.5)

Interaction

Agents’ ability to observe interaction among two agents. Thus, the observer
observations identifies one agent allocating a task to another agent or replying to an allocated task. Through interaction observations, agents can know about the competence of both the interacting agents in the given task. (section 5.5, section 5.6)

**J-**

**JADE** Java Agent Development Environment, a Java based software platform that provides middleware functionalities that facilitate implementation of multi-agent systems and distributed applications (Bellifemine et al., 2007).

**K-**

**Knowledge elicitation** Techniques used to determine the content of the mental model (Mohammed et al., 2000). (section 2.2.2.3)

**Knowledge representation** Technique used to reveal the structure of data or determine the relationships between elements in an individual’s mind (Mohammed et al., 2000). (section 2.2.2.3)

**L-**

**Lead agent** The agent selected by the Client Agent to perform the first task. In this research, for all the simulations the lead agent is selected through a bidding process. (section 5.2)

**M-**

**MAS** Multi-agent system, a system composed of multiple interacting intelligent agents. In this research, the team is modelled as MAS, where each agent is a team member. (section 2.3)

**MaxWindow** MaxWindow is the highest possible value for the capability range of an agent, i.e., if an agent can perform a task, it can provide at most MaxWindow number of solutions for that task. (section 5.4.3)

**MinWindow** MinWindow is the lowest possible value for the capability range of an agent, i.e., if an agent can perform a task, it can provide at least MinWindow number of solutions for that task. (section 5.4.3)

**N-**

**Non-routine tasks** Non-routine tasks are a combination of sequential and parallel tasks. These tasks have multiple valid solutions, such that two or more agents performing the same task may provide different solutions depending on their capability and knowledge of the solutions. (section 4.1.6.2)
<table>
<thead>
<tr>
<th><strong>NR-Agent</strong></th>
<th>Agent working on non-routine tasks. (section 5.4)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ORGAHEAD</strong></td>
<td>A multi-agent system modelling organizations. ORGAHEAD is used for developing theories relating to organizational design and organizational learning (Carley &amp; Svoboda, 1996). (section 2.3)</td>
</tr>
<tr>
<td><strong>P-</strong></td>
<td><strong>Parallel tasks</strong> Tasks where two or more sub-tasks are generated from the same higher level task, and can be performed simultaneously by different agents. Solutions generated for parallel tasks may be independent or may be dependent, in which case, compatibility of solutions needs to be evaluated. (section 4.1.6.3)</td>
</tr>
<tr>
<td><strong>Prior-acquaintance</strong></td>
<td>In this thesis, team familiarity and prior-acquaintance are used interchangeably. Prior-acquaintance is used to refer to dyadic relationships, while team familiarity is used at the collective level. However, higher team familiarity need not necessarily mean prior-acquaintance between all the agents that were part of the same team earlier.</td>
</tr>
<tr>
<td><strong>Process mental model</strong></td>
<td>The knowledge of team processes and task handling. In this research, the process mental model is pre-coded into the agents.</td>
</tr>
<tr>
<td><strong>Project-based teams</strong></td>
<td>Project-based teams are teams put together for the duration of a single project. In general, in such teams it is possible that members may or may not have worked together earlier in any other project. (section 2.2)</td>
</tr>
<tr>
<td><strong>R-</strong></td>
<td><strong>R-Agent</strong> Agents that work on routine tasks. (section 5.3)</td>
</tr>
<tr>
<td><strong>Rework</strong></td>
<td>If a task is not accepted, it is re-allocated to the agent.</td>
</tr>
<tr>
<td><strong>Routine task</strong></td>
<td>Tasks that are purely sequential and have unique solutions, such that two or more agents performing the same task will provide the same solution. Solutions to routine tasks are independent of the task performer. (section 4.1.6.1)</td>
</tr>
<tr>
<td><strong>S-</strong></td>
<td><strong>Sharedness</strong> Sharedness is an important characteristic of TMMs. The term shared is used to mean both (a) knowledge held in common by the team members, and (b) knowledge divided across the team members to form complementary knowledge. (section 2.2.2.1)</td>
</tr>
<tr>
<td><strong>Sequential tasks</strong></td>
<td>Tasks for which sub-tasks can only be performed if the preceding sub-task has been completed. (section 4.1.6.3)</td>
</tr>
<tr>
<td>----------------------</td>
<td>-----------------------------------------------------------------------------------------------------</td>
</tr>
<tr>
<td><strong>Simulation Controller</strong></td>
<td>A reactive agent that is required to: start and monitor the simulations; check the number of simulation runs; switch between training rounds and test rounds of the simulation; and, shut down the simulations based on the parameters set by the experimenter. (section 5.8)</td>
</tr>
<tr>
<td><strong>Simulation lifecycle</strong></td>
<td>The simulation lifecycle consists of the period from the start of the simulation platform to the closing down of the simulation platform. In these simulations, a single simulation lifecycle consists of 60 simulation runs.</td>
</tr>
<tr>
<td><strong>Simulation round</strong></td>
<td>A simulation round consists of one complete project. A simulation round can either be a training round or a test round.</td>
</tr>
<tr>
<td><strong>Simulation run</strong></td>
<td>A simulation run is one complete set of simulation from which results can be obtained. A single simulation run consists of two simulation rounds, such that one round is the training round and the other round is the test round.</td>
</tr>
<tr>
<td><strong>Social agent</strong></td>
<td>An agent that exhibits some degree of interdependence with other agents, where agents are part of a community in which they interact based on common rules and protocols that are either given or developed by the agents. In this research, the common rules and protocols are given to the agents in form of the process and context mental models. (section 2.4)</td>
</tr>
<tr>
<td><strong>Social learning</strong></td>
<td>The ability of agents to learn from social interactions and observations, which includes personal interactions, task observations and interaction observations. (section 2.1, section 5.5, section 5.6)</td>
</tr>
<tr>
<td><strong>Solution span</strong></td>
<td>The range of solution defined by the lower and upper values of solutions within which all solutions are either acceptable to an agent or define the capability range of an agent. (section 5.4.3)</td>
</tr>
<tr>
<td><strong>Social Turing test</strong></td>
<td>A test to validate a computational model developed to conduct social simulations (Carley &amp; Newell, 1994). (section 2.3)</td>
</tr>
<tr>
<td><strong>Source agent</strong></td>
<td>Agents from whom the message is received. In some cases, it has been used to refer to task allocator.</td>
</tr>
<tr>
<td><strong>Subsumption architecture</strong></td>
<td>A reactive agent architecture, which is organized hierarchically as layers of finite state machines.</td>
</tr>
<tr>
<td><strong>Target agent</strong></td>
<td>Agents to which the message is directed. In some cases, it is used to refer to the</td>
</tr>
</tbody>
</table>
agent expected to perform the task.

<table>
<thead>
<tr>
<th>Task allocator</th>
<th>Agents that allocate the task. For non-routine tasks, the task allocator also evaluates the solutions provided by task performers for their compatibility.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Task-based sub-teams</td>
<td>Teams organized as sub-teams based on expertise. In teams organized as task-based sub-teams not only is the agents’ ability to observe other agents limited within the sub-team but even most of the task-allocation interactions are within the sub-team. (section 2.2.1)</td>
</tr>
<tr>
<td>Task coordination</td>
<td>For non-routine tasks, a task may be required to be decomposed into sub-tasks and allocated in parallel to the relevant experts. The solutions received for each of the sub-tasks needs to be evaluated for compatibility. If solutions are not compatible, then some sub-tasks need to be chosen for re-work and re-allocated to the experts, and the cycle continues until the all the solutions are compatible.</td>
</tr>
<tr>
<td>Task decomposition</td>
<td>Generating sub-tasks for parallel allocation. In this research, task decomposition is applicable only to non-routine tasks.</td>
</tr>
<tr>
<td>Task observations</td>
<td>Agents’ ability to observe a task being performed by another agent. Thus, the observer knows that the observed agent can perform the observed task. (section 5.5, section 5.6)</td>
</tr>
<tr>
<td>Task handling</td>
<td>Activities related to task identification, coordination, decomposition and performance. (section 4.1.6.3, section 5.3.3, section 5.4.4)</td>
</tr>
<tr>
<td>Task mental model</td>
<td>The understanding and knowledge of the tasks to be performed by the team.</td>
</tr>
<tr>
<td>Team expertise</td>
<td>Team expertise is said to develop as agents in the team develop mental models for task, process, context and the team. In this research, the task, process and context mental models are pre-coded into the agents. Therefore, as the TMM is formed, it leads to the formation of team expertise, i.e., team expertise develops as agents learn to efficiently utilize each other’s expertise and allocate tasks to agents that have the expertise in performing the given task. (section 2.2.2.4)</td>
</tr>
<tr>
<td>Team familiarity</td>
<td>The percentage of agents that were part of the same team earlier. Even if agents may have been part of the same team earlier, it does not necessarily mean that agents have a pre-developed mental model of each other at the start of the new project because they may not have had the opportunity to interact or observe each other in the earlier project. (section 4.1.5)</td>
</tr>
</tbody>
</table>
| Team mental model (TMM) | The knowledge of an agent about its own competence, and the competence of all the other agents in the team, in terms of the tasks to be performed by the
team.

Computationally, the TMM is represented as an $m \times n$ matrix, representing the competence of each of the $n$ team members in each of the $m$ tasks to be performed by the team. (section 5.3.2, section 5.4.2)

| TMM formation | TMM formation is the amount of information about the team that any agent acquires through social learning. (section 5.3.4, section 2.2.2.3) |
| Team performance | Team performance is the performance and abilities of the team as a unit (Cook & Whitmeyer, 1992). In this research, team performance is measured as the amount of team communication, i.e., the total number of messages exchanged by the team members. (section 2.2.2.4) |
| Team structure | How the agents in a team are organized in terms of their task allocation, personal interactions, and social observations (i.e., task observations, interaction observations) (section 2.2, section 2.2.1) |
| Transactive memory system | A system through which groups collectively store and retrieve knowledge. This knowledge remains distributed across the team members. An important aspect of transactive memory systems is that each member should know where what knowledge is stored. (section 2.2.2.1) |
| V- | Virtual Design Team, a multi-agent system modelling organizations. VDT is focused on identifying the influence of organizational structure and information processing tools on team performance, assessed mainly from the perspective of project management and scheduling. The VDT involves modelling the processing time, work flow, and tool usage. (section 2.3) |
| Virtual teams | Virtual teams are special case of distributed teams in which it is likely that members may have never met each other in a face-to-face interaction (Griffith et al., 1998; Katzy, 1998; Leinonen et al., 2005; McDonough et al., 2001). (section 2.2, section 2.2.1) |
| W- | “What if” scenarios Hypothetical scenarios created by superposition of different independent variables that are difficult to control and simulate in real world studies. |
| Y- | “yellow page” services A service provided in JADE through the DF agents, through which all the |
agents in the team can access details of all the other agents in the team. In these simulations the “yellow page” services are used selectively. Team members access the DF agent only to identify group members but not the details of their expertise.