Socially Capable Conversational Agents for Multi-Party Interactive Situations

Rohit Kumar

CMU-LTI-11-013

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15213
www.lti.cs.cmu.edu

Thesis Committee:

Carolyn P. Rosé, Chair
Alan W. Black
Ian R. Lane
Jason D. Williams, AT&T Research

*Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
in Language and Information Technologies*

Copyright © 2011, Rohit Kumar

This research was supported in part by U. S. National Science Foundation (Grant numbers: EEC 0935145, DRL 0835426, SBE 0836012, DUE 0837661). The views and conclusions in this document are those of the author and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government.

Keywords:
Conversational Agents, Artificial Intelligence, Small Group Communication, Human-Computer Interaction, Social Interaction, Multi-Party Interaction, Collaborative Learning, Group Decision Making

To the motion pictures

that shape our sense of good, right and useful.

Abstract

Since the inception of AI research, great strides have been made towards achieving the goal of extending natural language conversation as a medium of interaction with machines. Today, we find many Conversational Agents (CAs) situated in various aspects of our everyday life such as information access, education and entertainment. However, most of the existing work on CAs has focused on agents that support only one user in each interactive session.

On the other hand, people organize themselves in groups such as teams of co-workers, family and networks of friends. With the mass-adoption of Internet based communication technologies for group interaction, there is an unprecedented opportunity for CAs to support interactive situations involving multiple human participants. Support provided by these CAs can make the functioning of some of these groups more efficient, enjoyable and rewarding to the participants.

Through our work on supporting various Multi-Party Interactive Situations (MPIS), we have identified two problems that must be addressed in order to embed effective CAs in such situations. The first problem highlights the technical challenges involving the development of CAs in MPIS. Existing approaches for modeling agent behavior make assumptions that break down in multi-party interaction. As a step towards addressing this problem, this thesis contributes the Basilica software architecture that uses an event-driven approach to model conversation as an orchestration of triggering of conversational behaviors. This architecture alleviates the technical problems by providing a rich representational capability and the flexibility to address complex interaction dynamics.

The second problem involves the choice of appropriate agent behaviors. In MPIS, agents must compete with human participants for attention in order to effectively deliver support and interventions. In this work, we follow a model of human group interaction developed by empirical research in small group communication. This model identifies two fundamental processes in human group interaction, i.e., Instrumental (Task-related) and Expressive (Social-Emotional). Behaviors that constitute this expressive process hold the key to managing and regulating user attention and serve other social functions in group interaction.

This thesis describes two socially capable conversational agents that support users in collaborative learning and group decision making activities. Their social capabilities are composed of a set of behaviors based on the Social-Emotional interaction categories identified by work in small group communication. These agents demonstrate the generalizability of our methodology for designing and implementing social capabilities across two very different interactive situations.

In addition to the implementation of these agents, the thesis presents a series of experiments and analysis conducted to investigate the effectiveness of these social capabilities. First and foremost, these experiments show significant benefits of the use of socially capable agents on task success and agent perception across the two different interactive situations listed above. Second, they investigate issues related to the appropriate use of these social capabilities specifically in terms of the amount and timing of the constituent social behaviors. Finally, these experiments provide an understanding of the underlying mechanism that explains the effects that social capabilities can achieve.

Acknowledgements

At the conclusion of this significant step in my career, I want to thank some of the many individuals and groups who have enabled my journey so far. First among them is my advisor Carolyn, who has made a long lasting impact on my approach towards research and towards people. *Thank you Carolyn. I hope to pass on some of your teachings forward.*

I wish to express my gratitude towards my mentors and advisors who have provided me with endless opportunities to explore areas of my interest. Most recently: Carolyn Rosé, Alan Black, Jason Williams, Ian Lane, Alex Rudnicky, Anatole Gershman, Robert Frederking, Lewis Johnson, Diane Litman, Kishore Prahallad, Sanjeev Sofat and Rahul Jindal. Also, I have acquired a wealth of knowledge by observing and interacting with the some of the extra-ordinary faculty and researchers at CMU: Jack Beuth, Jack Mostow, Roni Rosenfeld, Tanja Schultz, Bruce McLaren and Vincent Aleven.

Thank you to my colleagues who have provided an eco-system to nourish my thought and work. I think I will never be able find a replacement to that anywhere. Gahgene, Mahesh, Yi-Chia, Iris, Gregory, Hua, Moon, Nitin, Alicia, Le, Vasco, José, John, Brian, Satanjeev, Dan, Antoine, Jaime, Emil, Rashmi and Sharath. The Dialogs on Dialogs group has been instrumental at keeping me going when self-doubt fogged my path.

Several campus groups helped me achieve a work-life balance including LTI Student Activities Committee, Indian Graduate Students' Association, Graduate Student Assembly and Project Olympus. Thanks to Radha, Dana, Brooke, Linda, Mary Jo, Kelly and Stacey for facilitating my academic and socio-cultural activities at CMU for the past six years. *So many things would have been impossible without your support.*

Beyond my life on campus, my friends made it possible to keep Pittsburgh interesting for over six years. They have continually served me with a balanced diet of all round social life. While a quarter of this dissertation deals with discovering the right amount of social behavior for agents, I must credit these guys for having figured out that formula well in advance. Mudit, Ayesha, Mohit Kumar, Shivangi, Sana, Shailendra, Laxmi, Arpit, Sujoyeeta, Sourish, Prince Udhyakumar, Vamshi, Matt, Venkat, Swapnil, Kaustav, Suyash, Vishnu, Mohit Aggarwal, Priyanka, Betty, Aarav, Ponguru, Pranjali, Saurabh. While the cycle of meeting new people and sharing memorable experiences will continue, I doubt I will ever find another group of such incredible friends in my life again.

While the pages are unlimited, my memory never forgets to fail me. However, I cannot forget to thank my parents for many obvious reasons. Most of all, I want to thank them for allowing me to find my way as it is the one reason which I have often convinced myself to take for granted. *Thanks Mummy, Papa.*

Contents

Chapter 1 Introduction 1

1.1 Conversational Agents 1

1.2 Multi-Party Interactive Situations 2

1.3 Conversational Agents in Multi-Party Interactive Situations 3

1.3.1 Organizing space for Agents 5

1.4 Thesis Objective & Organization 8

Chapter 2 Basilica: Software Architecture 9

2.1 Desiderata 9

2.1.1 Lack of Rich Representational Capability 9

2.1.2 Inflexibility to address Complex Interaction Dynamics ... 10

2.1.3 Development Effort 13

2.2 A Model of Interaction 13

2.3 Basilica: The new architecture 14

2.4 Integrating existing behavior within Agents 17

2.5 An Example Agent: SecondLife Tutor 18

2.6 Supporting the Agent Development Process 21

2.6.1 Re-Use of Decomposable Components 21

2.6.2 Development Tools 21

2.7 Related Work 23

Chapter 3 Agents built using Basilica 25

3.1 CycleTalk Tutor 26

3.2 PsychChallenge Peer 30

3.3 Emergency Response Interpreter Agent 32

3.4 Types of Behavioral Components 35

Chapter 4 Socially Capable Conversational Agents 37
4.1 Need for Social Capabilities 37
4.2 Small Group Communication 38
4.3 Social Interaction Strategies 40
4.4 Related Work 41
4.5 Implementation of Social Behavior 42
4.6 Alternative Perspectives 45

Chapter 5 Application: Collaborative Learning 47
5.1 Methodology & Metrics 47
5.1.1 Recruitment 47
5.1.2 Design 48
5.1.3 Procedure 48
5.1.4 Materials 48
5.2 Experiment 1: Benefits of Social Behaviors 48
5.2.1 Experimental Design 49
5.2.2 Learning Outcomes 53
5.2.3 Perception Ratings 53
5.3 Analysis of performed Social Behavior 55
5.4 Analysis of effect of Social Behavior 56
5.4.1 Coding Tutoring Episodes 57
5.4.2 Structural Equation Modeling 58
5.4.3 Interpretation 62
5.5 Experiment 2: Amount of Social Behavior 62
5.5.1 Agent Implementation 63
5.5.2 Experimental Design 65
5.5.3 Learning Outcomes 67
5.5.4 Survey Outcomes 68

5.5.5 Exposure Effect with Tutors 69
5.5.6 Estimating the Optimal Amount of Social Behavior 70
5.5.7 Summary of Experiment 2 71

Chapter 6 Triggering Policy for Social Behavior 73

6.1 Modeling Human Social Behavior 74
6.1.1 Data 74
6.1.2 Learning Problem 78
6.1.3 Metrics 78
6.1.4 Features 80
6.1.5 Generating Social Behaviors 81
6.1.6 Baseline Experiments 82
6.1.7 Proposed Algorithm 83
6.1.8 Social Ratio Filtering 87
6.1.9 Results 88
6.2 Experiment 3: Evaluating a Human-like Triggering Policy 92
6.2.1 Procedure & Materials 92
6.2.2 Experimental Design 94
6.2.3 Results 94
6.2.4 Analysis of Tutoring Episodes 97
6.2.5 Discussion 100
6.2.7 Summary 103

Chapter 7 Application: Group Decision Making 105

7.1 Non-Combatant Evacuation Operation 105
7.1.1 Red Cross Rescue Scenario 105
7.1.2 Procedure 106
7.1.3 Metrics 107
7.2 Agent for supporting Group-Decision making 108

7.2.1 Agent Capabilities 108
7.2.2 Implementation 111
7.3 Experiment 4: Supporting Group Decision Making 113
7.3.1 Experimental Design 114
7.3.2 Participants 114
7.3.3 Results 115

Chapter 8 Conclusion 121
8.1 Thesis Contributions 122
8.1.1 Building Agents for Multi-Party Interactive Situations.. 122
8.1.2 Socially Capable Conversational Agents 123
8.2 Directions 125
8.2.1 Outlook 127
Appendix A Test administered during Wrench Lab 129
Appendix B Test administered during Thermodynamics Lab 131
Appendix C Collaborative Learning Perception Survey 135
Appendix D Design Sheet for Collaborative Wrench Design Activity 137
Appendix E Design sheet for Collaborative Power Plant Design Activity 139
Appendix F Scoring Rubric for Non-Combatant Evacuation Planning 141
Appendix G Group Decision Making Perception Survey 143
Appendix H Knowledge Test: Non-Combatant Evacuation 147
Appendix I Design of the Annotation Interface 151
Appendix J Rules for Triggering Social Behaviors 153
Bibliography 155

List of Figures

Figure 1.1: Percentage of publications on Multi-Party Interactive Situations out of the publications on Conversational Agents / Dialog Systems 4

Figure 2.1: Failure of Even Participation assumption in (a): Two-Party Interaction and (b): Multi-party interaction 11

Figure 2.2: A behaviorist model for Conversational Agents 14

Figure 2.3: Example of an Agent's Component Network (T) 15

Figure 2.4: Logic Diagram of Behavioral Components 16

Figure 2.5: Example XML Specification of a Basilica Agent 17

Figure 2.6: Two students interacting with the Second Life Tutor 18

Figure 2.7: Component Network of the Second Life Tutor 19

Figure 2.8: Basilica Visual Debugging Interface 22

Figure 3.1: ConcertChat Collaboration Environment 27

Figure 3.2: Component Network of the CycleTalk Tutor 29

Figure 3.3: The PsychChallenge Game Web-Interface (Showing the Guesser Interface) 31

Figure 3.4: Component Network of the PsychChallenge Peer 32

Figure 3.5: Component Network of the Emergency Response Interpreter Agent 34

Figure 4.1: Component Network of the WrenchTalk Tutor 44

Figure 5.1: Average ratings for the Tutor (Q1-Q6) and the Learning Task (Q7-Q9). 54

Figure 5.2: Venn Diagram of Episode Turn Annotations 57

Figure 5.3: SEM discovered using all 6 variables in our dataset 59

Figure 5.4: SEM including the *MeanResponseTime* and *UnrespondedTurns* variables 60

Figure 5.5: SEM with normalized variables 61

Figure 5.6: Component Network of the CycleTalk Tutor used in Experiment 2 64

Figure 5.7: Average ratings for the Tutor and the Learning Task 68

Figure 5.8: Interaction between our Experimental manipulation and Prior Exposure to Tutors 69

Figure 5.9: Scatter plot between Adjusted Post-Test scores and Social Ratio of the tutors in High and Low conditions........ 70

Figure 6.1: Mean Label Confidence & No. of Social Turns for different values of Confidence Threshold Θ 77

Figure 6.2: Pseudo-code of our Large-Margin Learner 84

Figure 6.3: Estimated function for SRTutor 87

Figure 6.4: Example of Social Behavior being generated by the Learnt Model (1)... 90

Figure 6.5: Example of Social Behavior being generated by the Learnt Model (2)... 91

Figure 6.6: SEM applied to data from this experiment........ 98

Figure 6.7: SEM from Meta-Analysis of Experiment 1 and Experiment 3 99

Figure 7.1: Component Network of the NEO Agent 112

Figure 7.2: Communication Environment for NEO Group Decision Making Activity 113

Figure 7.3: Plot of Total Score and Coarse & Fine grained Penalties 116

Figure 7.4: Mean Rating by the participants for the Agent and their Teammates.... 117

Figure 7.5: Average Ratings about Team, Task and Discussion 118

List of Tables

Table 1.1: Examples of conversational agents split across Task and Role 6

Table 2.1: Excerpt of a conversation between two students and a tutor 12

Table 2.2: Excerpt of a conversation between two students and the Second Life Tutor 20

Table 3.1: Excerpt showing the Attention Grabbing Strategy (Turn 61) 27

Table 3.2: Excerpt showing the Ask when Ready Strategy (Turn 41 & 42) 28

Table 4.1: Excerpt of an interaction between a CA (tutor) and a group of students .. 37

Table 4.2: Interaction Categories of Bales' Interaction Process Analysis Scheme 39

Table 4.3: Social Interaction Strategies based on three of Bales' Socio-Emotional Interaction Categories 40

Table 4.4: Excerpt of a conversation between three students and WrenchTalk Tutor 43

Table 5.1: Excerpt of a tutor providing a lesson to a team of four students 49

Table 5.2: Excerpts showing examples of the Social Interaction Strategies 51

Table 5.3: Average number of social behavior turns displayed by tutor 55

Table 5.4: Excerpt of a Conceptual Tutoring Episode 56

Table 5.5: Excerpt of an interaction between a team of students and a Human tutor 63

Table 5.6: Excerpt of an interaction between a team of students and an automated tutor 63

Table 5.7: Average Pre & Post test scores for each condition (Standard deviation in paranthesis) 67

Table 6.1: Labeling Categories 75

Table 6.2: Confusion matrix for 5-class labels (Θ=0.65) 76

Table 6.3: Summary of Baseline Results 82

Table 6.4: Evaluation results of our proposed Triggering Policies 89

Table 6.5: Mean and Standard Deviation of Adjusted Post Test Scores for Short Essay Type Questions 96

Table 6.6: Mean and Standard Deviation of Tutor Ratings 97

Table 6.7: Mean and Standard Deviation of Duration of Tutoring Episodes 98
Table 6.8: Excerpt showing a social behavior being triggered too late (1) 102
Table 6.9: Excerpt showing a social behavior being triggered too late (2) 102
Table 7.1: Excerpt of an interaction between Agent and Participants 108
Table 7.2: Social Interaction Strategies used by Agent to support Group Decision Making 109
Table 7.3: Examples of Instantiation of Social Interaction Strategies during the NEO Group Decision Making Activity... 110

Chapter 1

Introduction

1.1 Conversational Agents

Human Society has used conversation as an efficient, reliable and adaptive means of exchanging knowledge. One of the earliest grand challenges of artificial intelligence has been to extend this conversational medium to interactions with automated participants like computers or robots. We refer to these automated participants as Conversational Agents (CAs) or simply as Agents hereon.

Agents are autonomous interfaces that extend Conversation (spoken or text-based) as a medium of interaction with machines. A large number of such agents have been built and experimented with over the last three decades in a variety of interactive situations. Specifically, a number of agents have been developed and deployed for information access applications. They are also an extension of a history of interactive voice response systems (IVRS) and are commonly also referred to as spoken dialog systems. Some of the agents deployed in commercial / public use include: Amtrak Julie, AT&T How May I Help You (Gorin et. al., 1997), CMU Let's Go (Raux et. al., 2005) and Siri Virtual Personal Assistant. A much larger number of such agents have been developed in laboratories, some of which have been deployed publicly for academic study and evaluation purposes. Some examples of such agents include NJFun (Litman et. al., 2000), ConQuest (Bohus et. al., 2007a), Project54 (Kun et. al., 2007), AthosMail (Turunen et. al., 2004), Trains/Trips (Allen et. al., 1996), Clarissa (Rayner et. al., 2005), etc.

Another application of CAs which has been extensively researched is automated tutoring. Various research groups have developed agents in a variety of domains including reading (Aist and Mostow, 2009), algebra (Patel et. al., 2003), geometry (Aleven et.al., 2001), calculus (Murray et. al., 2001; Callaway et. al., 2007), physics (Rosé et. al., 2001a; Litman and Silliman, 2004; Jordan et. al., 2006; VanLehn et. al., 2007), computer literacy (Graesser et. al., 2003), programming (Lane and VanLehn, 2004), foreign languages (Johnson, 2007), research methods (Arnott et. al., 2008) and thermodynamics (Rosé et. al., 2006). Many evaluations show that CAs can be effective tutors (Arnott et. al., 2008; Kumar et. al., 2007a; Graesser et. al., 2005).

Other common applications of CAs include customer support (NoHold), translation assistance (Nallasamy et. al. 2008), entertainment (Foner, 1997; A.L.I.C.E.), marketing (Cassell et. al., 1999), navigation (Edlund et. al., 2004), security (Pakucs and Melin, 2001), therapy (Bickmore and Cassell, 2001; Weizenbaum, 1966; Ferguson et. al., 2009), personal assistance (Siri), etc. Dan Bohus and Staffan Larsson maintain a listing of several spoken dialog systems on their websites[1].

Several commercial products and platforms (Nuance Café, Voxeo) are publicly available which have made development tools for agents for simple task domains easily accessible to a large number of developers using standardized frameworks like VoiceXML. Besides the mass availability and presence of these agents/systems, the Loebner Prize, an annual competition to evaluate a class of CAs continues to contribute to the popularity and wide-spread interest in these conversational agents.

However, despite their popularity and presence in the real world, most of the work on creating CAs over the last five decades has focused on agents that can engage in interaction with a single human user. More recently, there is an emerging interest in building agents that can be one of the many human and automated participants in interactive situations.

1.2 Multi-Party Interactive Situations

Everyday, we participate in interactive situations involving two or more people. These multi-party situations are as common as sharing a meal with friends and family to having a meeting with colleagues at our workplace. Besides serving the need for companionship, these situations provide opportunities for group work. In Intellective Tasks such as problem solving, several researchers (Laughlin, 1980; Davis, 1969)

---

[1] http://research.microsoft.com/en-us/um/people/dbohus/SDS/index.html
http://www.ling.gu.se/~sl/dialogue_links.html

find that groups are more effective (more right answers) than individuals. Even though this increased effectiveness comes at the cost of time (“man-hours”), it is preferred practice to function in groups in a variety of situations. The set of situations where the costs of time is affordable is of interest to organizational behavior, communication and management scholars.

Advances in communication technologies over the last two decades have enabled a larger variety of groups to participate in joint activities in a variety of interactive situations. These multi-party interactive situations include situations where the participants interact with each other face-to-face (FTF) as well as situations where communication is mediated through artificial environments like tele-conferencing, instant messengers and virtual worlds. Each environment affords different modalities for interaction. Comparisons of these mediums based on their affordances relative to face-to-face interaction are discussed widely in literature on Computer Mediated Communication (CMC). Some examples of Multi-Party Interactive Situations include

- Collaborative Learning
- Online communication
- Collaborative Work
- Online shopping / auctioning
- Multi-Player games
- Collaborative content creation
- Social networking

As this research in CMC continues to progress, communication technologies will continue to enable and create a variety of multi-party interactive situations.

1.3 Conversational Agents in Multi-Party Interactive Situations

Advances in digital telephone systems in the latter half of the twentieth century led to the mass proliferation of Conversational Agents that could interact with individual users to help them with routine interactive tasks over the telephone. We can see an increasing interest in Conversational Agents that can participate in multi-party interactive situations enabled by currently available CMC environments. An estimation of this increased interest can be seen from the graph in Figure 1.1 which shows the percentage of annual publications on conversational agents that are also about multi-party interactions.

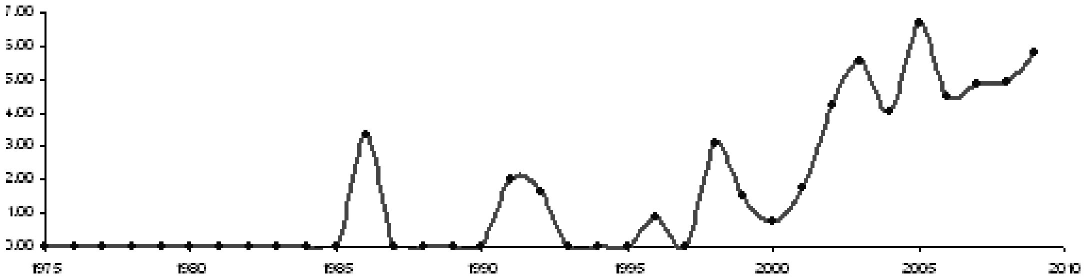


Figure 1.1: Percentage of publications on Multi-Party Interactive Situations out of the publications on Conversational Agents / Dialog Systems

One of the earliest agents deployed in a multi-user chat room was CoBot (Isbell et. al., 2001). It accumulates presence information of users and shares that information with other users who may be looking for their friends. Isbister et. al. (2000) developed a Helper Agent that introduces safe topics of discussion in a multi-cultural human-human interaction environment to improve group functioning and the perceptions of group participants about each other. In the context of the Virtual Humans project, Traum & Rickel (2002) developed an act-based model for creating conversational agents capable of interacting with multiple users using verbal as well as non-verbal interaction modalities. Elva (Zheng et. al., 2005) is an embodied tour guide that facilitates multi-party interaction in an interactive art gallery environment.

In educational domains, Kumar et. al. (2007a) have shown that agents playing the role of a tutor in a collaborative learning environment can lead to over one grade improvement. Other work (Liu & Chee, 2004; Kumar et. al., 2007b, Chaudhuri et. al., 2008 & 2009) has explored a variety of interaction patterns and tactics that could be used in multi-party educational situations.

Bohus & Horvitz (2009) have demonstrated multiple open-world interactive situations in which multiple human users are able to engage in interaction with an agent. Dohsaka et. al. (2009) created an agent that can engage a group of users in a quiz-style thought-evoking dialogue.

Besides the work on such agents in the academia, the entertainment industry has developed several scripted agents that can operate in massively multi-player games like Sims. Further, other non-conversational support tools for multi-party situations are being studied (Banerjee & Rudnicky, 2006).

While situations involving one agent and multiple human participants are the primary focus of this thesis, it must be noted that the general case of multi-party interactive situations can involve multiple agents and one or more human users. An example of this case is explored in the TeamTalk project (Harris & Rudnicky, 2007) that supports a human controller during search and rescue mission using multiple robots.

These initial efforts for building conversational agents to support multi-party interaction are based on extending existing approaches for creating agents that interact with one user at a time. This approach leads to two problems identified and addressed by this thesis. The first problem, discussed in Chapter 2, pertains to the assumptions made by approaches for modeling one-on-one interaction that do not generalize to multi-party interaction. Chapter 4 discusses the problems resulting for the lack of necessary communication skills that the agents must possess to participate in groups.

1.3.1 Organizing space for Agents

Within the collection of agents for MPIS listed above, we can see a diversity of applications and environments including agents that can participate in face-to-face interactions. In this section, I will present an organizing space to classify and compare between different agents. A space like this would help in generalizing design principles between agents by observing the similarities and differences between the agents.

Isbister & Doyle (2002) proposed a taxonomy for organizing research in embodied conversational agents. In order to organize the agents described and evaluated in this thesis, a narrower organizing space is described below. This space is split orthogonally along two dimensions, i.e., task and role.

Task

This dimension is similar in motivation to the application domains class of the Isbister & Doyle. The task domain within which the agent is situated determines most of the primary functions of the agent. Further the application for which the agent is being used specifies the metrics that must be used to evaluate the usefulness/effectiveness of the agent.

		Task			
		Behavioral	**Conceptual**	**Cooperative**	**Competitive**
Role	**Same**	CoBot	Agent as peer learner	Agent as team member	Agent as another buyer in Marketplace
	Different	Information Access (Bohus & Horovitz)	Tutoring Agent (Kumar et. al.)	Helper Agent (Isbister et. al.)	Shop keeper / Auctioneer in Marketplace

Table 1.1: Examples of conversational agents split across Task and Role

As we design and evaluate categories of social behaviors exhibited by the agent, we want to study the generalizability of those behaviors across agents involved in different types of tasks. Broadly, the tasks can be classified along four classes under this dimension based on the group task circumplex proposed by McGrath (1984). These tasks cover most of the common types of applications of interest currently.

1. **Behavioral tasks:** Tasks involving a sequence of physical and cognitive actions to achieve a predetermined objective such as asking for information. (Isbell et. al., 2001; Bohus & Horovitz, 2009)

   Metrics: Task completion, Reduction in time taken to complete the task, Task Satisfaction

2. **Conceptual tasks:** Tasks such as problem solving that require participants to acquire and share knowledge to achieve conceptual agreement (Kumar et. al. 2007; Liu & Chee, 2004)

   Metrics: Concept coverage, Concept learning / retention, Task Satisfaction, Task Quality/Creativity

3. **Cooperative tasks:** Tasks such as planning and decision-making during which participate with shared task goals generate and evaluate options to achieve their goals. (Isbister et. al., 2000)

   Metrics: Cost of Resources (time, money, people, equipment), Reward, Group Cohesion/Bonding, Task Completion Rate, Task Satisfaction

4. **Competitive tasks:** Tasks that involve contest over conflicting goals where participants attempt to maximize achievement of individual goals. (Dohsaka et. al., 2009)

   Metrics: User resource depletion, Agent resource accumulated, Time to victory, Number of exchanges

Role

Agents can play a variety of roles in different interactive situations. For example tutors play the roles of instructors and moderators in learning groups. An agent can play the role of another participant in a competitive task group and compete with the other participants. In a multi-lingual or multi-cultural task, agents can play the role of mediators as interpreters. In other tasks, agent can play the role of helpers, observers, referee, etc.

The role an agent plays is crucial to the study of agent behaviors. It determines the relevance of various behaviors an agent could exhibit. Further, the role sets up expectations of performance / capability / knowledge from the agent with respect to the user. For example: An agent playing the role of a peer student is not expected to know the answers to conceptual questions while a tutor agent is expected to be capable of responding accurately on such matters.

In order to keep the space of agents relatively small to allow possible generalizations between agent classes, the role dimension in the proposed space will have only two levels – i.e. same and different – defined with respect to the other participants. When we discuss specific agents in Chapter 3, their role will be subjectively elaborated in addition to categorizing the agent within these two levels.

Note that it is possible to add other dimensions that will define this organizing space at an increasingly fine level of detail. Foremost in the list of those dimensions would be the type of interactive environment (e.g., mobile, chatroom, voice, virtual worlds, face to face, etc.) employed for the interactive situation as that determines the affordances the agent can utilize while performing its functions.

1.4 Thesis Objective & Organization

In our work on building Conversational Agents for Multi-Party Interactive Situations, we have identified two problems that must be addressed to develop these agents. As discussed earlier, this thesis focuses on addressing both of these problems.

Foremost is the problem of building these software agents. While several representations and formalisms have been developed for developing agents that interact with a single user, all of these formalisms make simplifying assumptions (discussed in Section 2.1) that do not hold in the multi-party case.

Besides this technical problem, we also need to address the problem of designing agent behaviors that demonstrate necessary communication skills suitable for the interactive task and the role of the agent. In the case of agents participating in conversation with multiple users, the agents must perform social behaviors, which may be task-specific and task-independent, to engage the users as it competes with the other human participants to hold the floor.

Through this thesis, we are investigating general solutions to both of these problems. Chapter 2 of this document elaborates the technical challenge and describes a new software architecture that helps in alleviating the problems related to building CAs for MPIS. Chapter 3 describes the implementation of three agents built using this architecture. Two other agents built using this architecture that have been used in the experiments presented here are discussed in Chapter 4 and Chapter 7 respectively.

Chapter 4 motivates the need for socially capable conversational agents in multi-party interactive situations and presents a model of social behavior. We also describe an implementation of an agent with these capabilities. Chapter 5, Chapter 6 and Chapter 7 describe a series of experiments conducted using our socially capable agents to study their effects on user productivity and perception ratings.

Chapter 2

Basilica: Software Architecture

2.1 Desiderata

The first challenge that must be addressed towards building Conversational Agents in Multi-Party Interactive Situations is the technical challenge of implementing such agents as software.

Work on conversational agents and dialog systems in single user applications has explored several representations (Constantinides et. al., 1998; Rudnicky and Xu, 1999; Freedman, 2000; Rosé et. al., 2003; Bohus and Rudnicky, 2003) and implementation solutions (Seneff et. al., 1998; Turunen and Hakulinen, 2003; Bohus et. al., 2007b; Nakano et. al., 2008) that can be potentially extended to CAs in the multi-party case. However, as we borrow these approaches to build agents in multi-party situations, we note that there are certain shortcomings in these approaches. In this chapter, we will present these shortcomings that led us to define the desiderata for a new architecture, Basilica.

2.1.1 Lack of Rich Representational Capability

To achieve autonomous behavior by agents, characterized as involving a combination of simulated cognition and control, it was important to achieve a level of representational richness that allows us to model the agent in the concerned general class of conversational situations. However, while it would be relatively simple to add complexity to the representation if that was the only consideration, it could easily lead to the downside that the effort involved in authoring (or programming) the knowledge and the procedures that enable the agent to participate in specific situations would increase beyond what is practical. Thus, the consideration of representational adequacy and efficiency of implementation often conflict with each other.

For example, a simple representation like a finite state machine (Rosé et. al., 2003) is suitable only for relatively short and simple interactions, but the advantage of using them is that the development effort involved is relatively small and is often facilitated using state-machine authoring tools (Jordan et. al., 2007). Richer representations like the ones used in plan-based approaches (Freedman, 2000; Bohus and Rudnicky, 2003) model the conversational goal(s) of the agent and use planning algorithms to determine a sequence of steps that can achieve the goal(s). While such approaches have been shown to be flexible and robust in conversational situations like mixed-initiative dialog, a considerable amount of effort is involved in specifying the goal representations, operators, potential steps, pre-conditions, etc., required by the underlying planning algorithms.

In the new architecture proposed here, we adopt a rich representational capability that is not restricted by a small set of interaction operators. The interactive behaviors are specified using the full representational capability of a high-level programming language. This enables the developers of CAs in multi-party interactive situations to program complex interactive behaviors like the ones we describe in the case-studies in Chapter 3.

2.1.2 Inflexibility to address Complex Interaction Dynamics

Considering the amount of research that has gone into developing approaches for building CAs that are capable of conversational interaction with one user in each session, it is natural to push the envelope in order to deploy agents in multi-user interactive situations. However, typical approaches to developing CAs for single user settings make heavy use of the simplifying assumption that there are only two participants in the interaction, namely the human user and the agent. From this fundamental assumption come two more practical assumptions. One is that there will be a relatively even participation of both parties, which will typically mean that speakers take turns alternately. And the other assumption is the known addressee assumption, namely that if there are only two participants, then the addressee must always be the one who is not the speaker. Here we discuss why these assumptions break down in multi-party scenarios and what practical implications that has.

Figure 2.1a represents a typical interaction in a single user (two-party) scenario. The white dots represent the agent turns and the grey dots represent the user turns. Notice that typically white and grey alternate with one another. This is not the case 100% of time. Nevertheless, it is true often enough that if the system behaves in a way that presupposes that this will always be the case, it won't make mistakes very often. In this specific example, at turn 7, imagine that the user is responding to a agent prompt for information, and after the user has provided an initial answer, he then provides additional information (or a correction) to the system. However, since the agent (system) is based on an even participation assumption, at turn 8, the system is still trying to respond to turn 6 from the user and ignores the information provided in turn 7. In the last few years, work on flexible turn taking (Raux, 2008; Benuš, 2009) has proposed sophisticated models that can help the system anticipate the possibility of failure of such an assumption and avoid (or recover from) potential failure in the interaction. However, as illustrated in Figure 2.1b, in the case of a multi-party interaction, the failure of even-participation assumption is not an exception to be recovered from. Instead it is a normal characteristic of the dynamic interaction in multi-party settings. In this example, each color represents a different speaker. As can be seen, speakers do not alternate in any predictable pattern of even participation. Assuming the white dots represent the agent turns, we can see that ambiguity about which contribution to consider as an answer to its prompts is common rather than a rare exception.

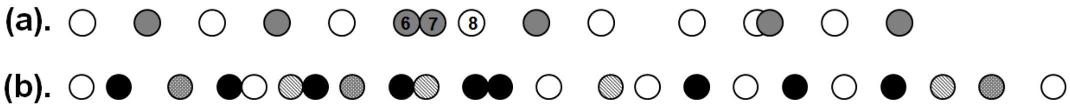


Figure 2.1: Failure of Even Participation assumption in
(a): Two-Party Interaction and (b): Multi-party interaction

Related to the problem just described is the problem of knowing who is the addressee of an utterance. When there are multiple speakers, sometimes the speakers will be talking to each other, and not the agent. Mainly the agent needs to know when it is being addressed, but this is far from trivial in this multi-user case. The known addressee assumption on which conversational agents for two-party interactions are developed implies in the two-party case that the addressee is the other speaker. A naïve extension of this assumption to the multi-party case would be that contributions from each participant are addressed to all the other participants, which includes the agent. Failure of this assumption happens when the user says something that is not addressed to the agent, or even if the agent is among the users addressed, but the agent’s prompts are not addressed. This is illustrated in the excerpt from a thermodynamics tutorial dialog system shown in Table 2.1.

When Student2 asks Student1 to respond to the tutor’s first question, the tutor follows the known addressee assumption and considers Student2’s turn as a response to its previous question. The response is evaluated as an incorrect (or non-understandable) answer and the tutor provides the correct answer. Meanwhile, Student1's response to the first question is considered as a response to the second question from the tutor.

Student1	OK, lets start
Tutor	What would happen to the power output of a Rankine Cycle at a higher operating temperature?
Student2	hmmm … Can you answer that?
Student1	I think it will increase.
Tutor	The correct answer is that at a higher operating temperature, more heat is added to the cycle and hence the power output increases too. What about the heat rejected by the cycle though?
Student1	You are right S1. It increases too. Lets move on to the next topic.

Table 2.1: Excerpt of a conversation between two students and a tutor

State-of-the-art conversational agents implement error recovery strategies (Bohus, 2007) designed to deal with non-understandings or mis-understandings in order to recover from local failures of the known addressee assumption. In multi-party scenarios, the dynamics of responding to a turn from the user becomes increasingly complicated and task specific. For example, in a collaborative learning setting, students may choose to discuss the answer to a tutor turn among themselves before responding to the tutor.

An additional complicating factor is the duration of interaction with agents in typical collaborative learning scenarios. Agents that interact with one user at a time have been developed for interactive situations that do not require any more than a few minutes of interaction. As we extend the application of these agents to applications such as collaborative learning where a learning session could last from 30 minutes to multiple hours, the structure of the conversation becomes increasingly complex, and the breakdown of the above assumptions become increasing likely.

Group dynamics is another consideration. As the number of users participating in the interaction increases, the exchange between them becomes an increasingly larger factor in the interaction. This interaction may include interpersonal conflict, free riding, or other sources of process loss. Thus, agents may have to monitor and regulate the interaction between users in order to provide appropriate support. For example, in an extended collaborative learning interaction, the tutor may elicit participation from students who are contributing less than other students. Such task / interactive situation specific interaction dynamics may be more complex than the scenarios that have been explored in current work.

The proposed architecture provides the flexibility to develop conversational agents that can implement interaction strategies (and tactics) which would enable them to participate in multi-party interactive situations without failure as mentioned earlier in this section.

2.1.3 Development Effort

While the primary motivations that guided the design of our new architecture was to increase the representational capability that encodes the knowledge and the behavior of the agent, as well as implement agents capable of performing sophisticated interaction tactics and strategies, we note that these improvements could easily increase the effort involved in developing the conversational agents. Thus, a final objective of our effort has been to develop the architecture in a way that reduces development effort. To alleviate additional effort to some extent, the architecture adopts principles of object-oriented design. Modeling the agent as a collection of appropriately sized, independent objects allows incremental development as well as reusability as discussed in the next section. Earlier related work on architectures for conversational agents (O'Neill et. al., 2003) has also employed similar object-oriented programming principles for developing conversational agents.

Before we proceed to discuss the details of the architecture for building conversational agents proposed in this chapter, we present a model of interaction between a conversational participant and the environment.

2.2 A Model of Interaction

The participant could be a human user or an agent and the environment includes other participants and observers. The environment also specifies the modalities of interaction (e.g., text, voice, video, gesture, etc.) based on its affordances.

In the model of interaction shown in Figure 2.2, the participant (Agent) observes environmental stimuli (like entrance of a new participant, action by one of the current participants, change in environment such as server notifications, etc.). These stimuli are conveyed to the perception components of the agent, which process the stimuli in order to determine whether any relevant behavior is to be triggered in response to the stimuli. For example, in a telephone based interaction, the listener component (ears) triggers the behavior of hearing upon receiving the voice stimulus from the handset (medium/environment). The triggered behavior may respond by generating events (that are internal to the agent) as well as by sending a response back to the environment. The generated events are transferred to other components. This way the environmental stimulus is propagated through a collection of components that implement all the singular behaviors an agent can perform. As a result of the stimulus propagation, a response may be sent back to the environment, and internal states of each component may be updated.

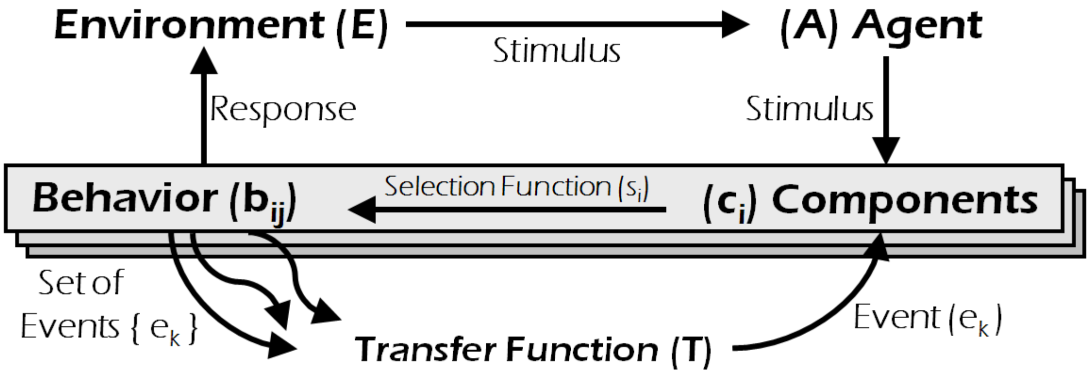


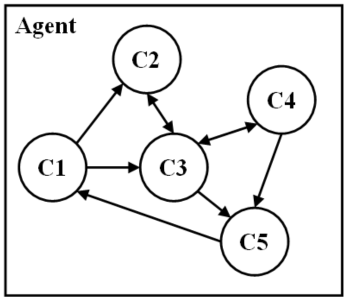
Figure 2.2: A behaviorist model for Conversational Agents

2.3 Basilica: The new architecture

The Basilica architecture is based on the above described model of interaction between conversational participants and the environment. Agents built using the Basilica architecture are implemented as a collection of what we refer to as behavioral components. Computational capabilities like perception, action, cognition and memory are implemented as behaviors. The selection function (s_i) for each component (c_i) is implemented as a one to one mapping function that maps the type of event (e_j) to behavior (b_{ij}). Each behavior is programmatically defined as a function that responds to a type of event by generating a set of zero or more events (e_k).

$$b_{ij}\left(type\left[e_j\right]\right) \rightarrow \{e_k\} \tag{2.1}$$

The transfer function (T) is specified by a network of components. Events generated by a behavior b_{ij} are propagated to all components that are connected to component c_i. The connections of the network are unidirectional, i.e., if component c_1 can receive events from c_2, then c_2 do not necessarily receive events from c_1 (unless so connected). For example, in the network shown in Figure 2.3, only components c_2 and c_3 receive events generated by c_1.



T = { (C1 ,C2), (C1, C3),
(C2, C3),
(C3, C2), (C3, C4), (C3, C5),
(C4, C3), (C4, C5),
(C5, C1) }

Figure 2.3: Example of an Agent’s Component Network (T)

The Basilica architecture provides a set of core abstract classes (implemented in Java) for defining agents, components, connections and events. While the behaviors performed by the agent are specific to the agent’s implementation and change between agents, the Basilica architecture provides low-level functionality required to implement the agents.

Foremost within this scope is the control mechanism for propagating events between components. Events are propagated as a broadcast to all connected components. For example, all events generated by c_1 are received by c_2 and c_3. While sometimes this might cause components to receive events that they do not need to process, the broadcast mechanism allows for relatively simpler specification of the network. Additionally, the architecture provides developers the ability to selectively transmit events to a subset of all connected components. Basilica is responsible for initializing and maintaining the connection between components over which events are transmitted. Besides maintaining these connections, the architecture provides observer interfaces that allow developers to observe events as they are transmitted over the connections. This supports creation of graphical displays that can be used by facilitators and moderators. The debugging interface discussed in section 3.5 uses this mechanism.

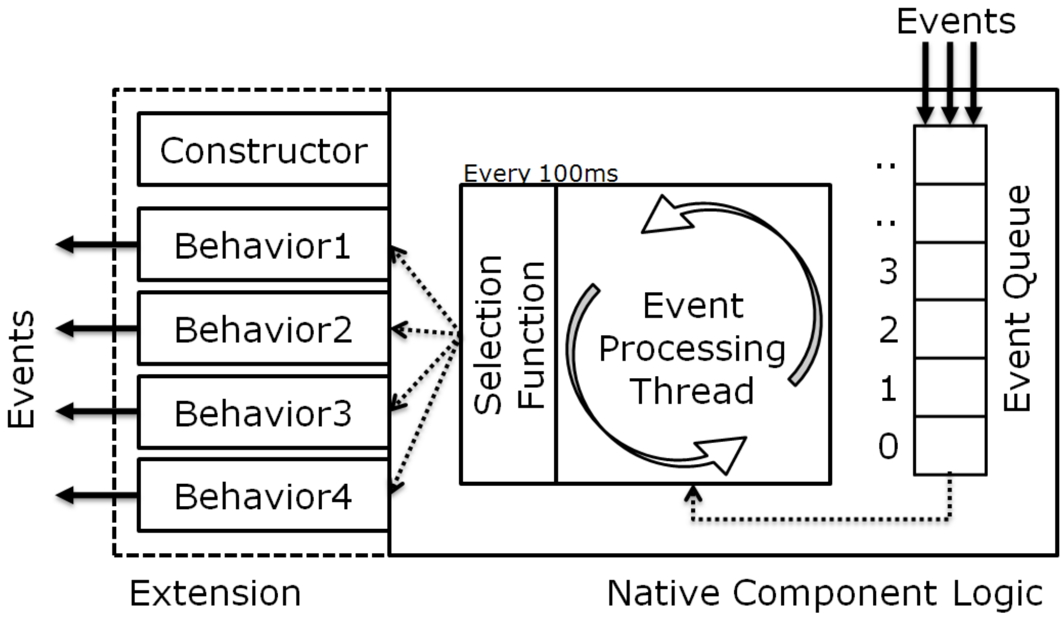
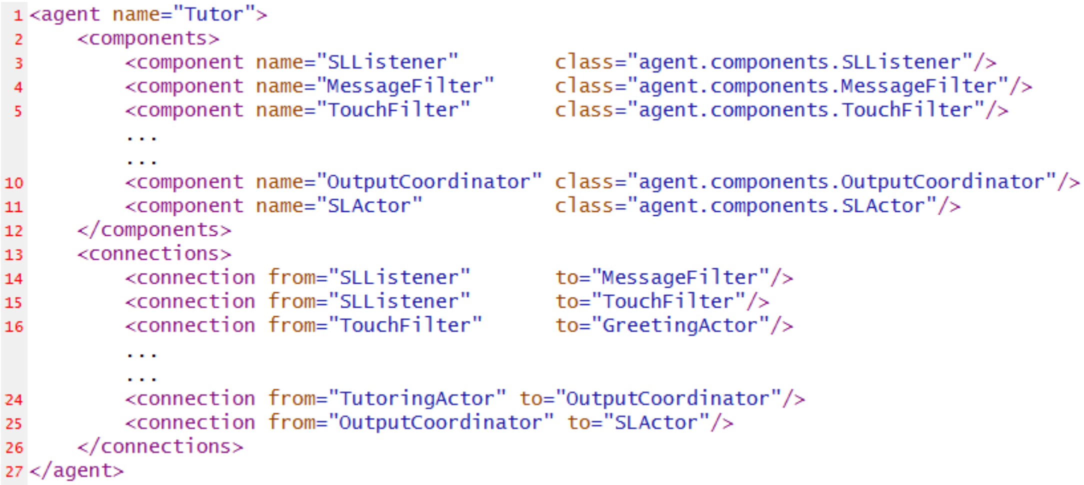


Figure 2.4: Logic Diagram of Behavioral Components

Second, the abstract classes used for defining behavioral components provide a generic mechanism for initializing, executing and observing each component. Figure 2.4 shows a logic diagram of behavioral component. By default this mechanism allows each component to perform its behaviors asynchronously by allowing each component to run in a separate thread. So, if a particular component (like a parser) takes an extended amount of time for processing its events, the other components are not blocked from processing their events.

Third, the selection function (s_i) within each component is responsible for accumulating incoming events and triggering their corresponding behaviors (b_{ij}). Basilica implements a generic mechanism for this function. By default, events are buffered and processed sequentially in the order in which events are received. However, the object oriented implementation of Basilica allows developers to override this default mechanism for special purpose components to prioritize certain kinds of stimuli (like a user barge-in or a resource unavailability notification).



```
<agent name="Tutor">
    <components>
        <component name="SLListener"        class="agent.components.SLListener"/>
        <component name="MessageFilter"     class="agent.components.MessageFilter"/>
        <component name="TouchFilter"       class="agent.components.TouchFilter"/>
        ...
        ...
        <component name="OutputCoordinator" class="agent.components.OutputCoordinator"/>
        <component name="SLActor"           class="agent.components.SLActor"/>
    </components>
    <connections>
        <connection from="SLListener"       to="MessageFilter"/>
        <connection from="SLListener"       to="TouchFilter"/>
        <connection from="TouchFilter"      to="GreetingActor"/>
        ...
        ...
        <connection from="TutoringActor" to="OutputCoordinator"/>
        <connection from="OutputCoordinator" to="SLActor"/>
    </connections>
</agent>
```

Figure 2.5: Example XML Specification of a Basilica Agent

Besides the core classes, Basilica provides a generic class for a memory component that provides the ability to keep state-based information accessible across components. An application of the memory component is discussed in Section 4.5. Additionally, the architecture provides an agent factory class that allows runtime agent construction from an XML specification like the one shown in Figure 2.5, which also enables more dynamic forms of behavior.

2.4 Integrating existing behavior within Agents

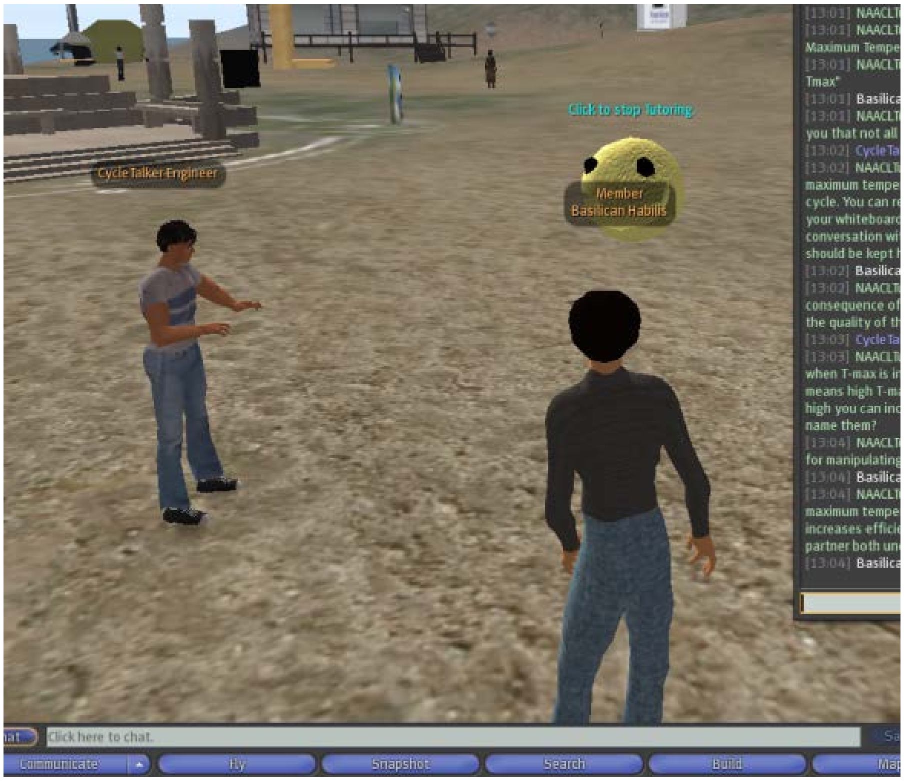
Basilica allows for integration of a wide range of behavioral components, but one that we have used frequently in our agents is the TuTalk dialog engine (Jordan et. al., 2007). In the next section, we will discuss its role within an example Basilica agent, illustrated in Figure 2.6. TuTalk is a state-based dialogue engine that operates using what are referred to as tutoring scripts. Tutoring scripts compatible with the TuTalk dialog engine define directed lines of reasoning composed of a sequence of steps that implement an Initiation – Response – Feedback interaction pattern with the goal of leading a student to construct a correct explanation for a complex concept as independently as possible. The dialog engine executes these steps by presenting the Initation question, matching the student response and presenting appropriate feedback before moving on to the next step. The script formalism also allows introducing another intervening sequence of remedial steps as feedback to incorrect responses.

]

Thus, support is provided on an as-needed basis. In order to facilitate authoring of these scripts, TuTalk provides a set of authoring tools for rapid development of these scripts by subject matter experts who may not be technology experts.

Integration of TuTalk within Basilica's tutoring components demonstrates the flexibility to integrate existing tools and interactive representations within agents built using this architecture. Note that the TuTalk dialog engine inherently does not provide a mechanism to address the issues related to multi-party interaction discussed earlier. However, Basilica allows us to augment these tutoring components with other necessary behavior to address the issues related to complex interaction dynamics without needing to add any sophistication to component technologies themselves. Table 2.1 and Table 2.2 illustrate example interactions with authored TuTalk agents.

2.5 An Example Agent: SecondLife Tutor



```
[13:01] NAACLT
[13:01] NAACLT
Maximum Tempe
[13:01] NAACLT
Tmax"
[13:01] Basilica
[13:01] NAACLT
you that not all
[13:02] CycleTa
[13:02] NAACLT
maximum tempe
cycle. You can re
your whiteboard
conversation wi
should be kept h
[13:02] Basilica
[13:02] NAACLT
consequence of
the quality of th
[13:03] CycleTa
[13:03] NAACLT
when T-max is in
means high T-ma
high you can inc
name them?
[13:04] NAACLT
for manipulating
[13:04] Basilica
[13:04] NAACLT
maximum tempe
increases efficie
partner both un
[13:04] Basilica
```

Figure 2.6: Two students interacting with the Second Life Tutor

Now we will demonstrate how an agent built using the Basilica architecture works using an example agent that tutors a team of students in the SecondLife (SL) virtual environment. The SecondLife Tutor shown in Figure 2.6 is implemented as a SecondLife object (see as the spherical object in the figure).

The SecondLife tutor performs two types of user observable behaviors, i.e., greeting and tutoring. To customize the tutoring behavior, the tutor can be augmented with a list of TuTalk scripts and the tutor sequentially executes those scripts. We see two students interacting with the tutor object using text chat. The users activate the tutor by clicking on it (touch stimuli).

Connectivity between the tutor and SecondLife environment is enabled using a HTTP Middleware (Weusijana et. al., 2008). The component network of the SecondLife tutor, shown graphically in Figure 2.7, is made of nine components and twelve connections. It receives two types of stimuli from the SecondLife environment, i.e.,

1. the user touching the agent to activate it and
2. the user sending message in the agent's vicinity

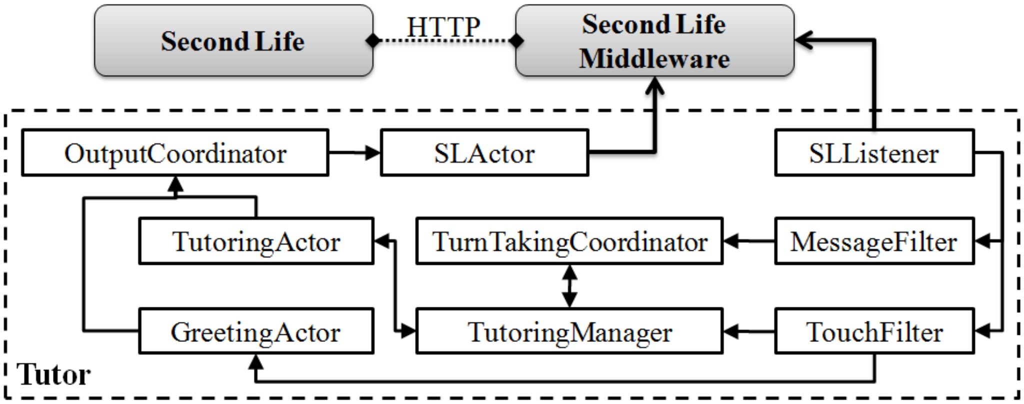


Figure 2.7: Component Network of the Second Life Tutor

When the tutor is activated, a *LaunchEvent* is propagated to the *GreetingActor* and the *TutoringManager*. *GreetingActor* sends a greeting message back to the environment via the *OutputCoordinator* and the *SLActor*. The *TutoringManager* encapsulates the TuTalk dialog engine. When triggered, it starts tutoring by sequentially executing the available TuTalk scripts. Tutor turns (questions and feedback) are sent to the environment via the *TutoringActor*, the *OutputCoordinator* and the *SLActor*. Student answers received via the *MessageFilter* are collected by the *TurnTakingCoordinator* when the Tutor is expecting the students to respond to its question (e.g. after the first turn of Table 2.2). If multiple students respond within a few moments of each other to the same tutor turn, the *TurnTakingCoordinator* accumulates those responses and sends them to the *TutoringManager* to be matched to the tutor's question. The *TutoringManager* filters responses collected by the *TurnTakingCoordinator* through prioritization rules that look for a correct or an incorrect answer, i.e., a relevant response to the tutor's questions.

[09:21]	**Tutor**	Consider the effect of T-max on the efficiency of a cycle. Do you know whether T-max should be kept high or low?
[09:22]	**Student1**	we should keep tmax high
[09:22]	**Tutor**	But let's think about one important consequence of keeping T-max high. At high T-max what happens to the quality of steam.
[09:24]	**Student2**	it increases i believe
[09:24]	**Tutor**	The turbine exit will have better steam quality when T-max is increased. So increasing steam quality and efficiency means high T-max is good. Nevertheless, there are limitations to how high you can increase T-max. There are mainly two problems. Can you name them?
[09:25]	**Student2**	safety is one... efficiency is the other?
[09:25]	**Tutor**	Mainly, safety concerns and exceeding the maximum temperature the material can withstand. Keeping Tmax high increases efficiency and also steam quality. Make sure you and your partner both understand these ideas.

Table 2.2: Excerpt of a conversation between two students and the Second Life Tutor

Here we have adopted a solution that is specific to the situation of tutoring interaction to deal with the problem of not knowing the addressee of a student turn since responses that are relevant to the tutor's question can be considered as addressed to the Tutor. If no relevant responses are received for an extended amount of time, the *TutoringManager* can employ interaction tactics such as repeating the question, urging the students to respond, giving a hint or moving on.

As mentioned earlier, the flexibility to incorporate components such as the *TurnTakingCoordinator* and *TutoringManager* is an example of how Basilica allows developers to address the complex interaction dynamics of multi-user interactive situations.

2.6 Supporting the Agent Development Process

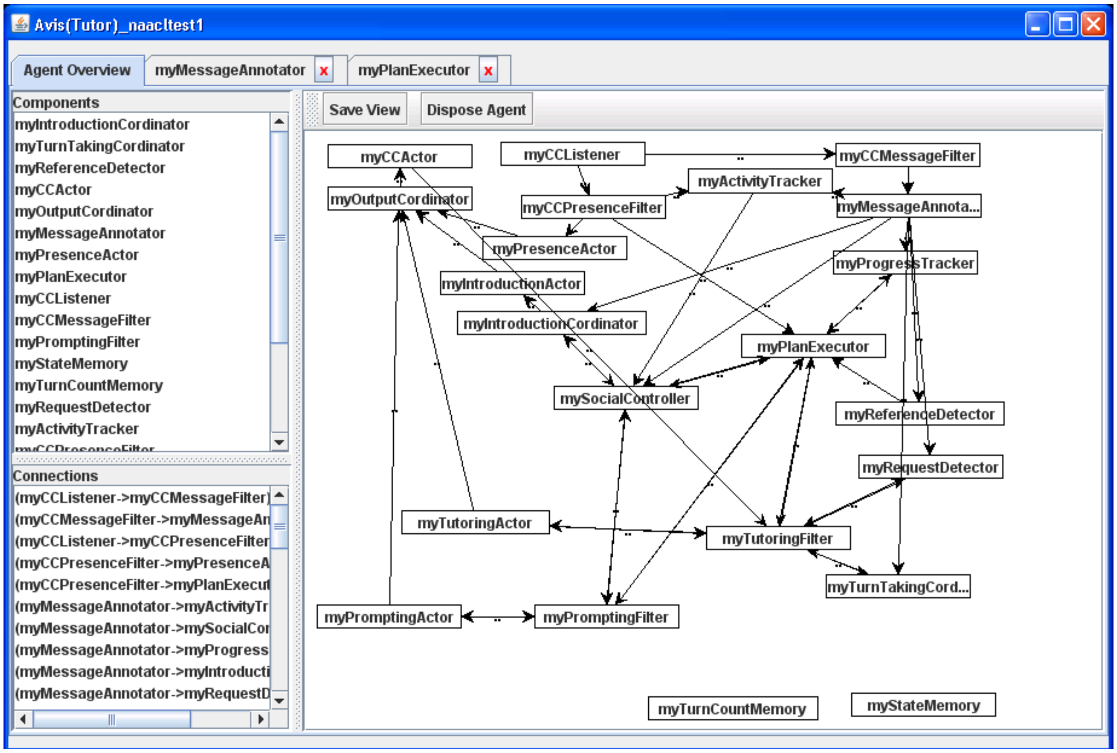
2.6.1 Re-Use of Decomposable Components

We have a growing set of behavioral components which can be re-used to build tutors for several learning situations. For example, as shown in the example in the previous section, the *TutoringManager* and the *TutoringActor* can be used to include tutoring scripts developed for the TuTalk system within the agent's interactive behavior. The agents discussed in Chapter 3 show extensive re-use of these components.

Furthermore, the interaction with the SecondLife environment is isolated to the *SLListener* and *SLActor* components. These components can be replaced to make the same agent work in other similar environments (like chatrooms or other multi-user virtual environments). It is useful to allow agents to operate in multiple environments with comparable affordances especially in the online learning situation to allow the students to be able to interact with the agent from an environment of their choice. Overall, decomposing agents into small, loosely coupled components that encapsulate behavior allows application of object-oriented programming principles that facilitate incremental and distributed development in teams. Just as these principles have enabled scalable software development, we believe that they will facilitate development of complex and highly interactive instructional agents for mass use.

2.6.2 Development Tools

Besides the core classes of the architecture, Basilica provides a variety of debugging utilities through loggers and observer classes. A visual debugging interface is available as a part of these utilities to help developers verify the connections between components and track event propagation as components are incrementally added to the agent. Figure 2.8 shows a screenshot of this debugging interface. The component network shown in the interface is animated as events propagate through the network. Developers can click on any component or connection to get a detailed look at the events generated and processed by each component.



Components
myIntroductionCordinator
myTurnTakingCordinator
myReferenceDetector
myCCActor
myOutputCordinator
myMessageAnnotator
myPresenceActor
myPlanExecutor
myCCListener
myCCMessageFilter
myPromptingFilter
myStateMemory
myTurnCountMemory
myRequestDetector
myActivityTracker

Connections
(myCCListener->myCCMessageFilter)
(myCCMessageFilter->myMessageAn
(myCCListener->myCCPresenceFilter
(myCCPresenceFilter->myPresenceA
(myCCPresenceFilter->myPlanExecut
(myMessageAnnotator->myActivityTr
(myMessageAnnotator->mySocialCo
(myMessageAnnotator->myProgress
(myMessageAnnotator->myIntroducti
(myMessageAnnotator->myRequestD

Figure 2.8: Basilica Visual Debugging Interface

To support the deployment of agents built using the Basilica architecture for large experiments, the architecture provides an operation class that can launch and manage several agents. Another utility built within the architecture is the Timer which can has been useful for implementing behaviors that should be triggered periodically rather than in a state-based fashion (e.g., checking for student participation).

Additionally, we have developed a set of simulated environment listener and actor components that allow developers to simulate user input from a previously available transcript of interaction between the user and a tutor. These transcripts can serve as extended testing scripts during development. Replacing the agent's environment listener and actor components with these components can help the developers in consistently and effortlessly test for previously known bugs.

2.7 Related Work

Recently, there has been other work on modeling conversational agents as a decomposition of components. Queen's Communicator (O'Neill et. al., 2003) applies principles of Object-Oriented programming to decompose CAs into classes that abstract task-independent conversational behavior and allows developers to programmatically extend them to create task-dependent behaviors.

Jaspis (Turunen and Hakulinen, 2003) models the agent as a collection of *managers*, *agents* and *evaluators* which synchronize with each other through *transactions*. Also, while Jaspis agents are stateless, actors in our architecture need not be stateless. RIME (Murray et. al., 2001) R. Charles Murray, Kurt VanLehn and Jack Mostow, 2001, *A decision-theoretic architecture for selecting tutorial discourse actions*, AIED-2001 Workshop on Tutorial Dialogue Systems, San Antonio, Texas

(Nakano et. al., 2008) distributes cognitive capabilities across a collection of *experts* of two types. In contrast to Basilica that allows components to be connected in any manner, in these architectures, the *evaluators* and the *agents* are configured as a collection of parallel modules. Hence, designing conversational agents with Basilica gives the flexibility to change the network topology. A recently proposed multi-policy approach to dialog management (Lison, 2011) models the agent as a collection of hierarchically connected concurrently operating policies. This approach is very similar to the Basilica architecture in terms of the flexibility it provides for modeling individual policies using different dialog modeling techniques and for interconnecting each of these policies.

In other work on event-based multi-layered architectures (Raux and Eskenazi, 2007), events are used for communication between layers as a mean to provide higher degree of reactivity compared to pipeline architectures. While we share this motivation, definition of events is extended here as events are used for all kinds of communication, coordination and control in Basilica. Also, recent interest in the use of incremental processing techniques for various components as well as entire dialog systems (Skantze and Schlangen, 2009; DeVault et. al., 2009) is enabled by the use of event-driven architectures. While Basilica does not explicitly model the input and output (left and right) buffers for all of its components by default, the principle that updates from preceding components in a pipeline should be incrementally processed by subsequent components is applied within agents built using Basilica by conveying updates as events.

Through this comparison of the Basilica architecture with the other approaches discussed here in terms of their similarities and differences, we find that the event-driven approach that the Basilica architecture is based on allows this architecture to serve as a meta-architecture. Basilica allows systems developers to combine representations and design principles from multiple approaches to create highly complex agents with rich behavioral capabilities.

In the next chapter, we will describe three agents building using Basilica. These agents will demonstrate the advantages of using the Basilica architecture while providing design patterns that can be used to develop other agents.

Chapter 3

Agents built using Basilica

The Basilica approach is applicable for building a variety of conversational agents that are situated in complex, extended, multi-party interactive situations. In this chapter, we will describe our work on developing Conversational Agents that support teams of two or more users in broad classes of multi-party interactive situations like Collaborative Learning and Mediated Interaction.

We will discuss the implementations of three agents developed by us using the Basilica architecture to highlight how these agents showcase the strengths of the proposed architecture. These agents explore the organization space of agents described in Section 1.3.1 both in terms of tasks and role. While the CycleTalk tutor and the PsychChallenge peer presented in Section 3.1 and 3.2 respectively support conceptual tasks, the 9-1-1 emergency response interpreter presented in Section 3.3 supports a behavioral task. The PsychChallenge peer agent plays the same role as the users while the other two agents play a role that is different compared to the users.

A study of the component networks of the agents described here as well as the agents described in Chapter 4 and Chapter 7 will show some common patterns. First of all, we notice that these agents use different types of components. Section 3.4 discusses the types of components commonly used in our agents.

Second, we can notice that the agents built using the Basilica architecture are an extension of the traditional pipeline approaches. We can observe multiple input-process-output pipelines that transport events from right to left in Figure 3.2, Figure 3.4, Figure 3.5, Figure 4.1 and Figure 7.1. We note that many components especially filters that perceive and process user input are often shared between these pipelines.

Third, we note that by choosing an appropriate level of decomposition of behaviors into different components, we can reuse components across agents. An example of this reuse can be seen between Figure 3.2 and Figure 4.1 where the

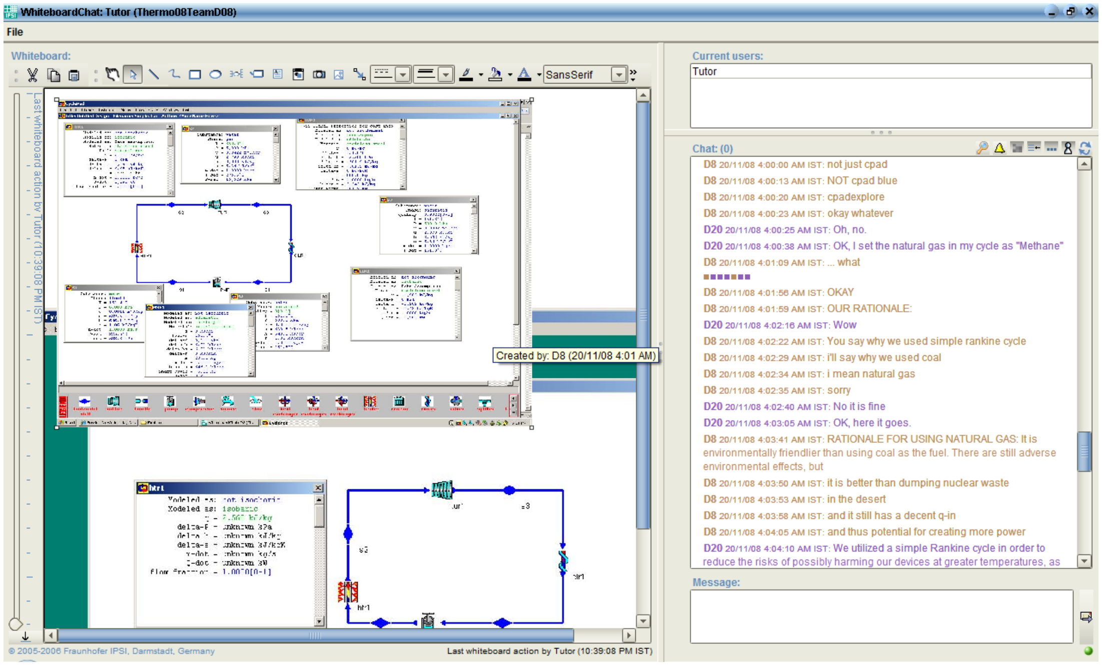
TutoringManager, *TutoringActor*, *TurnTakingCoordinator* and *RequestDetector* components are reused. The *RequestDetector* component that implements the *Ask When Ready* strategy discussed in the next section. It is separated from the *TutoringManager* because other components such as the *PlanExecutor* in Figure 4.1 that wait on verbal triggers from users to precondition the move to next steps can also use the *RequestDetector*'s capabilities. So, we can see that appropriate decomposition allows reuse of different types.

Additionally, the choice of level of decomposition of behaviors is determined by the need to change certain components without changing the behavior of other components. The *TutoringManager* component needs to be connected to a component that generates events to trigger the start of a directed tutoring episode. In Figure 3.1 the *AttentionGrabbingFilter* provides this event whereas in Figure 4.1 the *PlanExecutor* provides this event.

3.1 CycleTalk Tutor

CycleTalk is an intelligent tutoring system that helps sophomore university students learn principles of thermodynamic cycles (specifically Rankine Cycle) in the context of a power plant design task. Teams of two students work on designing a Rankine cycle using a Thermodynamics simulation software package called CyclePad (Forbus et. al., 1999). As a part of the design lab during which this learning task is performed, students participate in a collaborative design interaction for 30-45 minutes using ConcertChat, a text based collaboration environment (Mühlpfordt and Wessner, 2005) shown in Figure 3.1. Our work (Kumar et. al., 2007a) has shown the effectiveness of this collaborative design activity.

An automated tutor participates in the design interaction along with the two students. The CycleTalk tutor provides instructional support to the students to ensure that they learn underlying thermodynamic concepts as they design. The CycleTalk tutor was the first tutor implemented using the Basilica architecture and has been modified over the last three years in accordance with the evolution of the research studies conducted using this tutor. Improvements to the CycleTalk tutor served as requirements for improving the the Basilica architecture. Specifically, these improvements included development of various types of components, efficiencies in the architecture's event propagation mechanism and creation of suitable abstractions and interfaces to the architecture's core classes to facilitate development and re-use. Some of these aspects are discussed below as we present a recent implementation of the CycleTalk tutor implemented using the Basilica architecture.


WhiteboardChat: Tutor (Thermo08TeamD08)
File
Whiteboard:
SansSerif
Last whiteboard action by Tutor (10:39:08 PM IST)
Created by: D8 (20/11/08 4:01 AM)
© 2005-2006 Fraunhofer IPSI, Darmstadt, Germany
Current users:
Tutor
Chat: (0)
Message:

D8 20/11/08 4:00:00 AM IST: not just cpad
D8 20/11/08 4:00:13 AM IST: NOT cpad blue
D8 20/11/08 4:00:20 AM IST: cpadexplore
D8 20/11/08 4:00:23 AM IST: okay whatever
D20 20/11/08 4:00:25 AM IST: Oh, no.
D20 20/11/08 4:00:38 AM IST: OK, I set the natural gas in my cycle as "Methane"
D8 20/11/08 4:01:09 AM IST: ... what
D8 20/11/08 4:01:56 AM IST: OKAY
D8 20/11/08 4:01:59 AM IST: OUR RATIONALE:
D20 20/11/08 4:02:16 AM IST: Wow
D8 20/11/08 4:02:22 AM IST: You say why we used simple rankine cycle
D8 20/11/08 4:02:29 AM IST: i'll say why we used coal
D8 20/11/08 4:02:34 AM IST: i mean natural gas
D8 20/11/08 4:02:35 AM IST: sorry
D20 20/11/08 4:02:40 AM IST: No it is fine
D20 20/11/08 4:03:05 AM IST: OK, here it goes.
D8 20/11/08 4:03:41 AM IST: RATIONALE FOR USING NATURAL GAS: It is environmentally friendlier than using coal as the fuel. There are still adverse environmental effects, but
D8 20/11/08 4:03:50 AM IST: it is better than dumping nuclear waste
D8 20/11/08 4:03:53 AM IST: in the desert
D8 20/11/08 4:03:58 AM IST: and it still has a decent q-in
D8 20/11/08 4:04:05 AM IST: and thus potential for creating more power
D20 20/11/08 4:04:10 AM IST: We utilized a simple Rankine cycle in order to reduce the risks of possibly harming our devices at greater temperatures, as

Figure 3.1: ConcertChat Collaboration Environment

59	[09:35:07]	**Student2**	the pressure stays constant through the boiler?
60	[09:35:47]	**Student1**	Yeah
61	[09:35:55]	**Tutor**	Now might be a good time for some reflection.
62	[09:36:00]	**Tutor**	Consider the effect of increasing Qin. What happens to power out when Qin is increased?
64	[09:36:46]	**Student2**	i am pretty sure we want high Qin

Table 3.1: Excerpt showing the Attention Grabbing Strategy (Turn 61)

While we have shown that instructional support provided by automated tutors is effective, as compared to students working individually, we have observed that teams of students often ignore and abuse the tutor (Kumar et. al., 2007a). In our recent studies in this thermodynamics learning domain, we have investigated the use of interaction strategies that can help in engaging the students more deeply in the instructional conversation with the tutors. One of these strategies (*Attention Grabbing*) was designed to intrusively grab the student's attention (Chaudhuri et. al., 2008). It prompts the students to pay attention to the tutor before the tutor starts the instructional conversation. An excerpt of this strategy is shown in Table 3.1. The tutor prompts the students (Turn 61) to grab their attention and waits for a silence (5 seconds) between the students to infer that the students are now paying attention to the tutor.

Another strategy (*Ask when Ready*) developed as an improvement to the *Attention Grabbing* strategy informed the students that the tutor has a relevant instructional topic to discuss and asks them to let the tutor know when they were ready to talk about the topic (Chaudhuri et. al., 2009). This strategy allows the students to complete their current topic of discussion before engaging in conversation with the tutor. An example of this strategy is shown in Table 3.2. The tutor informs the students that it is ready to talk about P_{max} (maximum pressure in a rankine cycle) in turn 41 & 42. The students finish their current topic of discussion and indicate that they are ready to discuss P_{max} in turn 47. Note that the turn 41 is similar to principle to the attention grabbing prompt shown in turn 61 in Table 3.1.

40	[08.52.24]	**Student5**	and then Power out vs. the same things
41	[08.52.26]	**Tutor**	Lets review the effect of changing P-max on the cycle.
42	[08.52.27]	**Tutor**	Type: HELP WITH PMAXKCD if you want to discuss it with me.
...			...
47	[08.54.08]	**Student7**	HELP WITH PMAXKCD
48	[08.54.14]	**Tutor**	When P-max increases, is the need to reject heat from the cycle increased or decreased?
49	[08.54.51]	**Student5**	decreased

Table 3.2: Excerpt showing the Ask when Ready Strategy (Turn 41 & 42)

Both these agents employ the turn-taking and tutoring components discussed in the example agent in Section 2.5. Figure 3.2 shows the component network implemented for a tutor that employs the *Ask when Ready* strategy. It is made of 13 components and 21 connections. There are six types of components, i.e., *Listeners*, *Actors*, *Filters*, *Detectors*, *Coordinators* and *Managers*. *Listeners* listen to stimuli from the environment and translate them into events internal to the agent. *Actors* perform actions which may be directly observable by other participants in the environment. *Filters* process information that events carry and propagate them further based on their programmed conditions. *Detectors* are special kinds of Filter which detect specific semantic concepts/phrases and send out a detection event. *Coordinators* control the flow of events between related components to achieve coordinated behavior. *Manager* components exhibit a variety of behavior like planning, execution and control. A summary of the various types of components used in the agents developed in our work can be found in Section 3.4.

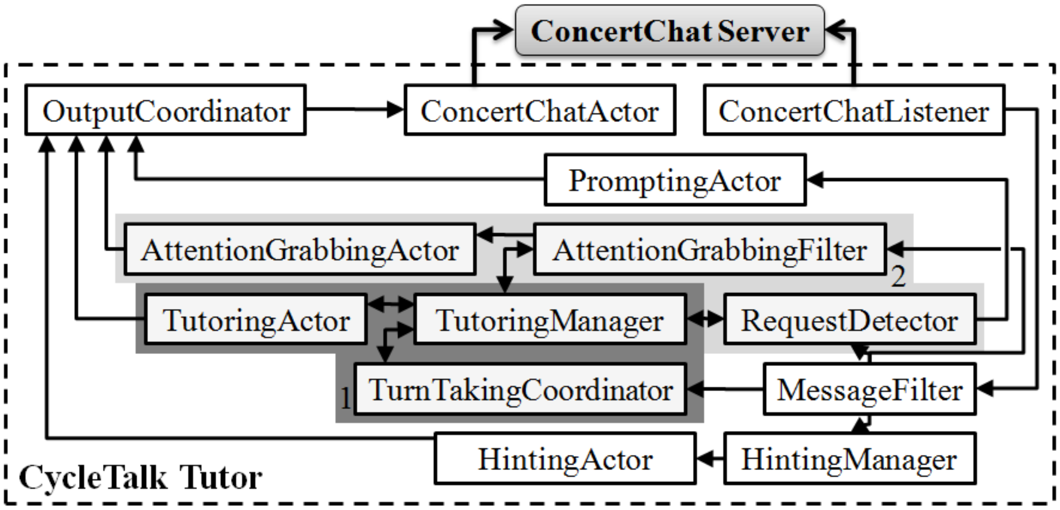


Figure 3.2: Component Network of the CycleTalk Tutor

We can note that components shown in the shaded area labeled as 1 in Figure 3.2 are connected the same way as those components in the example agent. In order to implement the behavioral capabilities that allow the agent to use the *Attention Grabbing* and *Ask when Ready* interaction strategies, we have added three new components shown in the shaded area labeled 2. When the *TutoringManager* decides that an instructional topic is relevant to the current discussion, it informs the *AttentionGrabbingFilter* to grab the students' attention using an appropriate interaction strategy. The *TutoringManager* also informs the *RequestDetector* to look out for the appropriate trigger phrase. Once the trigger phrase is detected, the

TutoringManager starts the TuTalk script corresponding to the requested instructional topic. Besides the tutoring behavior, the CycleTalk tutor has a hinting behavior implemented using the *HintingManager* and *HintingActor* that use a topic model to provide relevant hints based on the interaction between the students (Kumar et. al., 2007a).

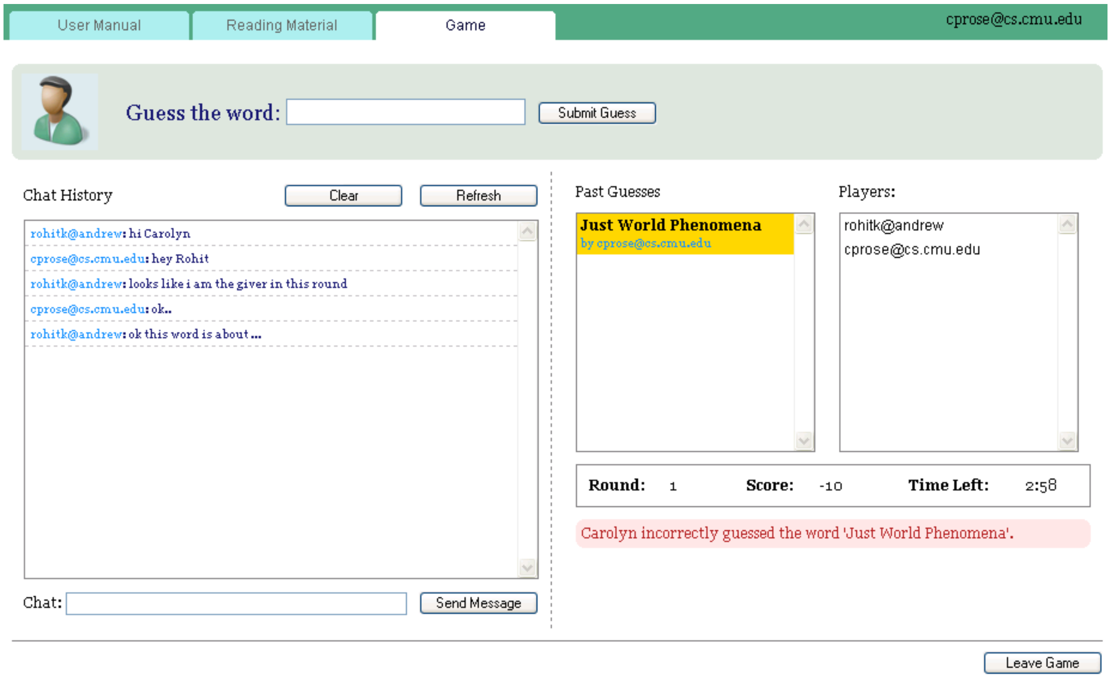
Using the Basilica architecture to develop the CycleTalk tutor has supported this line of investigation in multiple ways. Foremost, it may be noted that components corresponding to behaviors that do not change because our experimental strategies remain constant between prototypes because they can be isolated within the agent network in a modular way. This allows us to incrementally add behavioral components that implement the experimental strategies. Further, because of the use of a programmatic approach to building these agents, we are not restricted to a small set of operators provided by typical agent authoring languages making it possible to implement strategies like *Attention Grabbing* and *Ask when Ready*. Finally, the ability to integrate existing natural language processing modules as *Filter* components makes Basilica a helpful architecture for creating complex and highly interactive conversational agents.

Besides implementing strategies for beginning tutoring conversations with a team of students, we have been investigating the use of role assignment to the students (Chaudhuri et. al., 2008, Ai et. al, 2010). Teams of students are divided into Pro-Environment vs. Pro-Power roles to elicit broad coverage of arguments within the team during the learning interaction. Basilica has allowed us to create components that can take into account the task-specific learner roles while presenting instructional content to the students.

3.2 PsychChallenge Peer

The Basilica architecture does not make any specific assumptions about the role of the agent in the interaction. Agents supporting learners may not only be tutors. In this section we will discusses an agent that demonstrates the ability to build agents that play a variety of roles.

PsychChallenge is a vocabulary game that is part of a learning portal of an introductory psychology text book. Students play this game as part of an assignment as they progress through each chapter of the book. The game involves learning vocabulary related to the each chapter by helping each other guess terms related to the chapter by providing hints about the term. One of the players takes on the role of a hint *Giver* and the other players play the *Guesser* role. The game is accessible to the students through a special purpose web-interface shown in Figure 3.3.



rohitk@andrew: hi Carolyn
cprose@cs.cmu.edu: hey Rohit
rohitk@andrew: looks like i am the giver in this round
cprose@cs.cmu.edu: ok..
rohitk@andrew: ok this word is about ...

Just World Phenomena
by cprose@cs.cmu.edu

rohitk@andrew
cprose@cs.cmu.edu

Round: 1 **Score:** -10 **Time Left:** 2:58

Carolyn incorrectly guessed the word 'Just World Phenomena'.

Figure 3.3: The PsychChallenge Game Web-Interface
(Showing the Guesser Interface)

Students can choose to play the game with other students who are online at the same time. Teams (or individuals) can choose to add the PsychChallenge peer agent to the game. Note that this agent is different from the other agents discussed in this chapter in that it plays the role of a peer with respect to the student instead of a tutor. Figure 3.4 shows the component network of the peer agent. The agent is connected to the web-interface through a middleware component that operates in a way similar to the SecondLife HTTP middleware (Weusijana et. al, 2008). The peer agent plays the role of the *Guesser* or the *Giver* based on the role that it is assigned by the game. The *GuessingActor* and the *HintingActor* components are augmented with instructional content corresponding to each term in the game's vocabulary to allow the agent to behave intelligently. When the agent is playing the *Guesser* role it tries to elicit better hints from the *Giver*. When the agent is the *Giver*, it provides useful hints to the students to help them guess the correct term. Other behavior that this Peer agent displays includes greeting the students at the start of the game and informing them about its role at the change of every round.

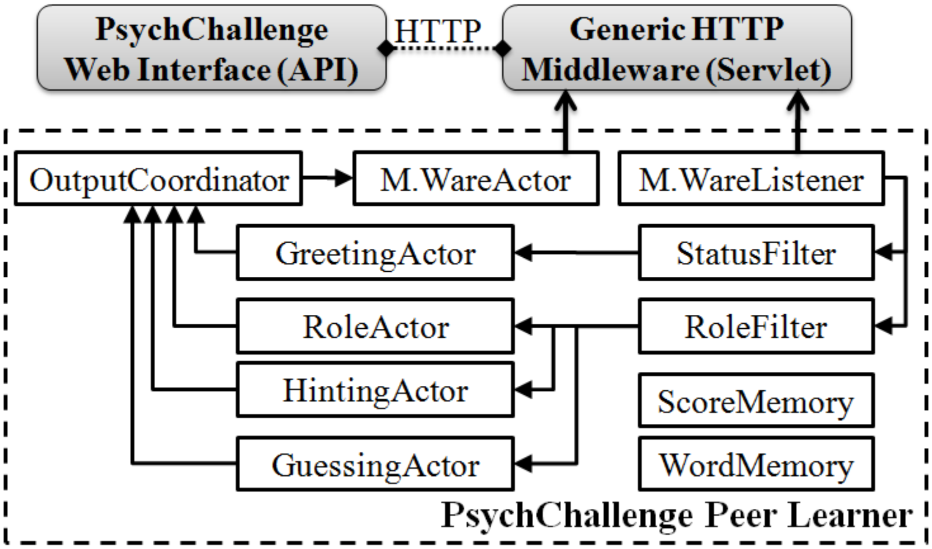


Figure 3.4: Component Network of the PsychChallenge Peer

3.3 Emergency Response Interpreter Agent

The interactive situations for the agents described earlier support involve two or more users where all the users have the same explicit task objectives (e.g. designing a power plant, or winning a game). Also, the implicit goals of the users (learners), which shape their interaction with the agents, are similar (i.e. learning the underlying domain concepts). In contrast to that, the emergency response interpreter agent described in this section supports an interactive situation where the two users have very different task objectives and goals of interaction.

The emergency response interactive situation involves two human participants i.e. a distressed caller and an emergency dispatch operator such as people employed at 9-1-1 response centers (Nallasamy et. al., 2008). In states such as Louisiana, emergency calls from Spanish speakers (especially at night) are mediated by third party human translation service that is dialed in when the dispatch operator determines that he/she cannot communicate with the caller because of the language barrier.

The emergency response interpreter agent is designed as a replacement to the third party translation service (for at least a subset of calls). The caller, dispatch operator (dispatcher) and the agent communicate using eJabberd, a general purpose communication backend based on the XMPP protocol. Caller dials in to the response center where a caller proxy makes a connection to an available dispatcher through the XMPP server. The caller proxy is responsible for packaging and broadcasting speech and recognition output of the caller turns to all participants in the call (dispatcher / agent).

We have developed a special purpose GUI for the dispatcher that allows the dispatcher to attend to the callers as well as interact with the agent using a variety of actions. Specifically the dispatcher can command the agent to perform:

- **MetaAct** (e.g. Ask-Emergency-Type) which are available at all times during the interaction
- **IQAAct** (e.g. Ask-Victim-Bleeding) which are specific to type of incident the caller is reporting based on the response center's protocols
- **ConfirmationAct** to ask the caller to confirm a piece of information that was provided earlier (e.g. confirming address of the victim)

Besides these controls, the GUI lets operators communicate with the available / dispatched units with updated information that the caller provides. A discourse history display in the GUI allows the operator to read the text of all the turns in a call (along with partial translations).

We can note that in this situation the agent providing a channel for communication between two users (the caller and the dispatch operator) who have very different task roles and hence different objectives. The caller's objective is to the get the required emergency resources (like ambulance, police, fire-truck) based on the incident. The dispatcher's objective is to ask the caller for the details of the incident and follow protocols to dispatch all the required resource and provide the dispatched resources with the information they require. While the objectives of the two participants do not conflict with each other, the agent must communicate with both the participants in a way that serves the individual objectives of each of the participant.

Figure 3.5 shows the components network of an agent we developed to participate in this interactive situation. We can notice that at a high level this network implements two pipelines (events flows right to left). One of the pipelines (shown in the shaded area marked as 2 in Figure 3.5)

- Takes information from the caller
- Annotates it with the syntactic as well as domain specific semantic information
- Identifies the type of incident or a piece of information related to the incident being reported
- Communicates that information to the dispatcher with appropriate displaying information such that it is organized properly on the dispatcher GUI

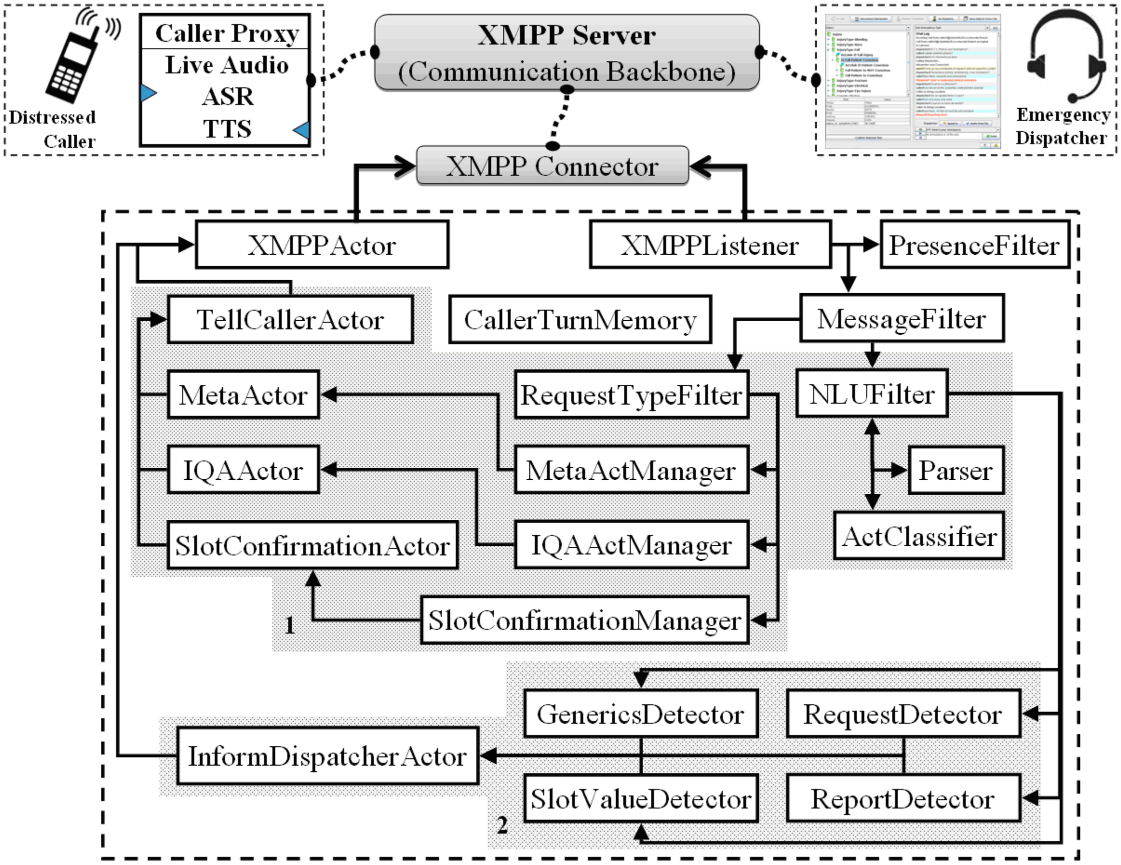


Figure 3.5: Component Network of the Emergency Response Interpreter Agent

The other pipeline (shown in the shaded area marked as 1) takes commands from the dispatcher and performs the corresponding action to provide or elicit information to/from the caller. The choice of which pipeline should a message over the XMPP broadcast channel be passed through is made by the *MessageFilter* components based on the sender of the message packet.

The implementation of this agent demonstrates the flexibility the Basilica architecture provides to build agents that can interact with users with different roles within the same interactive session. Agent behaviors required to understand and communicate with the two participants are implemented in the corresponding pipeline. This helps us in incrementally adding additional behaviors required to interact with the user (e.g. additional types of incident report from the caller).

3.4 Types of Behavioral Components

Based on the three agents developed using the Basilica architecture that are described in this chapter, we can identify the characteristics of different types of behavioral components that are necessary and/or common to most agents implemented using Basilica.

Foremost of these are the environment listener and environment actor components. They are necessary for all agents as they integrate the agent with a collaboration environment such as SecondLife (*SLListener* and *SLActor*) or ConcertChat (*ConcertChatListener* and *ConcertChatActor*). These can be readily re-used for new agents being developed for already supported environments. On the other hand, these components are among the first components to be implemented while developing agents for a new environment.

We can find multiple filter and detector components being used among the three agents. They perform a variety of operations (like parsing, classification, annotation, etc.) and transformations on the data encapsulated within events they receive. Further these components can be used to control the flow of events. For example the *MessageFilter* keep the presence events from the environment from propagating to components that do not need those events. All agents the process student input have atleast one of these components.

Different types of manager components make up the controllers of the different user observable behaviors of the agent. These managers keep track of knowledge resources and interaction states for the behaviors they perform and provide triggers to other components that realize the behaviors these managers control. All agents that actively interact with students have atleast one of these manager components. Note that for very simple behaviors the management logic is sometimes built within the actor components that realize those behaviors (e.g. the *GreetingActor* in Figure 3.4 and Actors in Figure 3.5).

Memory components while not mandatory to any agent implementation are often used for agents that need to have several managers. Finally, we find that all agents implement several special purpose components like actors that realize the user-observable behaviors and coordinators that facilitate agent participation in complex interaction dynamics. As discussed in section 2.6.1, many of these components can be reused among agents that display the same behaviors.

Chapter 4

Socially Capable Conversational Agents

4.1 Need for Social Capabilities

In our prior work on supporting Collaborative Learning, which is a type of Multi-Party Interactive Situation as it involves multiple students working together on a learning task, we have shown that students benefit both by learning as a group and receiving tutorials from agents (Kumar et. al., 2007a). However, students learning in groups ignore the tutor's messages most of the time, unlike the case where students were individually tutored. Also, groups often abused tutors as shown in the excerpt in Table 4.1.

Tutor	There will be more potential for cooling. Is there more or less potential for power generation?
St16	not necessarily
Tutor	There will be more potential for power generation ... Where might the increased heat input go instead?
St16	this tutor is really annoying
St5	Agreed

Table 4.1: Excerpt of an interaction between a CA (tutor) and a group of students

We reason that the presence of other students in collaborative learning scenarios caused the agents to compete for the attention of the students. Since the agents were not adept at performing social interactive behavior, which is a significant part of the formative phase of the group interaction, the agents were quickly pushed to the periphery of the learning group.

This observation demonstrates that besides the technical challenges underlying the development of CAs, we also find ourselves with a shortage of design principles that can help us make effective CAs in multi-party interactive situations. In particular, the presence of multiple human users in the interaction initiates additional communication processes within the group. We should design agents which can observe and participate in all of these processes with reasonable effectiveness when necessary.

Towards addressing this goal of creating agents as effective communicators, we must adopt principles of effective communication into agent design. Specifically in the case of multi-party interactive situations, investigations on small group communication offers relevant theories which are largely ignored in work on Conversational agents. One of the goals of this work is to bridge this gap between computational aspects of realizing CAs and theories in small group communication.

As discussed in the next section, we find that the agents which were being used in our earlier work do not perform social behaviors which are prominent and necessary in human group interaction.

4.2 Small Group Communication

Theoretical and empirical study of group interaction processes has been of interest in sociology and communications research communities since the 1950’s. McGrath (1984) reviews various theories that address the functions of group interaction processes. Of particular interest among these are the theories proposed by Robert F. Bales (1950) and Wilfred R. Bion (1961).

Both of these theories propose that two fundamental processes operate within groups. Bales identified that these two process correspond to instrumental (task-related) and expressive (socio-emotional) interaction. Bion divides group interaction between similar categories of Work and Emotion processes. Over attention to any one of these processes causes lapses on the other. Hence, interaction shifts between these two in order to keep the group functional.

Bales developed an Interaction Process Analysis (IPA) scheme to analyze group interaction along twelve interaction categories shown in Table 4.2. Six of these interaction categories correspond to instrumental (task-related) interaction and the other six correspond to expressive (social-emotional) interaction. IPA is of particular interest from the point of view of designing appropriate social behavior for CAs because the units of interactions that IPA is applied to are fine grained and correspond roughly to the size of conversational turns. Another analysis scheme developed by Bion and Thelen (1956) analyzes units at a coarse level of granularity such that each unit is made of about 30 - 50 utterances.

Positive Expressive Interaction Categories	Shows Solidarity
	Shows Tension Release
	Agrees
Instrumental Interaction Categories	Gives Suggestion
	Gives Opinion
	Gives Orientation
	Asks for Orientation
	Asks for Opinion
	Asks for Suggestion
Negative Expressive Interaction Categories	Disagrees
	Shows Tension
	Shows Antagonism

Table 4.2: Interaction Categories of Bales' Interaction Process Analysis Scheme

Most of the existing research on interaction strategies for Conversational Agents used in various interactive settings has focused on task-related strategies. In the case of conversational tutors, the task (or work) related interaction include aspects like instructing students about the task, delivering appropriate interventions in suitable form (e.g. socratic dialog, hints), providing feedback and other such tactics (Graesser et. al., 2001). Some studies (Rosé et. al., 2001b; Wang and Johnson, 2008) have evaluated the effect of these task related conversational behavior in tutorial dialog scenarios. Work in the area of affective computing and its application to tutorial dialog has focused on identification of student's emotional states (D'Mello et. al., 2008) and using those to improve choice of task related behavior by tutors.

However, there has been only limited study of expressive (socio-emotional) aspects of the agent's conversations. The focus of our work is to systematically use research in Small group communication to design expressive behavior that is relevant and appropriate for conversational agents in the interactive situations we are investigating.

4.3 Social Interaction Strategies

1. **Showing Solidarity:** Raises other's status, gives help, reward

1a. Do Introductions: Introduce and ask names of all participants
1b. Be Protective & Nurturing: Discourage teasing
1c. Give Reassurance: When student is discontent, asking for help
1d. Compliment / Praise: To acknowledge student contributions
1e. Encourage: When group or members are inactive
1f. Conclude Socially

2. **Showing Tension Release:** Jokes, laughs, shows satisfaction

2a. Expression of feeling better: After periods of tension, work pressure
2b. Be cheerful
2c. Express enthusiasm, elation, satisfaction: On completing significant task steps

3. **Agreeing:** Shows passive acceptance, understands, concurs, complies

3a. Show attention: To student ideas as encouragement
3b. Show comprehension / approval:
To student opinions and orientations

Table 4.3: Social Interaction Strategies based on three of Bales' Socio-Emotional Interaction Categories

In this section, we present our process of designing and implementing Social Behavior motivated from the literature in Small Group Communication discussed in the last section.

As discussed earlier, current state-of-the-art conversational agents do not perform the socio-emotional function of interaction that is known to be a fundamental aspect of group interaction. Hence, we hypothesize that socially capable agents will be able to perform better in multi-party interactive situations. In order to further specify social capability, we use the interaction process analysis (IPA) schema developed by Bales (1950).

IPA identifies three positive socio-emotional interaction categories: showing solidarity, showing tension release and agreeing. We have mapped these categories to practically implementable conversational strategies that are relevant to collaborative learning situations since this is the multi-party interactive situation we will use in most of our experiments. This mapping from IPA categories to Social Interaction Strategies is shown in Table 4.3.

These strategies were developed over two iterations. We conducted a pilot evaluation with 6 subjects between the iterations to verify if the strategies were successful at eliciting their intended perception from users. Subjects were shown crafted excerpts of interactions between users and an agent. The agent's turns in the excerpts that corresponded to social interaction strategies were highlighted and the subjects were asked to report their perception of friendliness, tension release and agreeing after reading each excerpt. Some strategies and corresponding prompts were modified in the cases where the reported scores were relatively low.

4.4 Related Work

While our work employs research in small group communication to design social behaviors for conversational agents, there has been some other work on CAs that employ social (non-task) behavior.

Bickmore et. al. (2009) report that users found agents with autobiographies i.e. back stories in first person more enjoyable and they completed more conversations with such agents. Dybala et. al. (2009) found that agents equipped with humor were evaluated as more human-like, funny and likeable. In a multi-party conversational scenario, Dohsaka et. al. (2009) found that an agent's use of emphatic expressions improved user satisfaction and user rating of the agent.

In educational applications, Wang and Johnson (2008) found that learners who received polite tutorial feedback reported higher increase in self-efficacy at the learning task. In our earlier work (Kumar et. al. 2007b) we used personal preferences determined from engaging in small talk with learners to construct word problems to engage the learners in the problem solving activity. Other work in affective computing has attempted to endow spoken dialog systems with emotional intelligence (Andre et. al., 2004) to respond to user's affective states during the interaction.

Besides the use of verbal social behavior, work on developing virtual humans and embodied conversational agents has investigated the user of non-verbal behavior through gesture and gaze to convey social messages to participants in the interaction.

Most of the earlier work on verbal social or non-task behavior focuses on only one type of behavior such as human, politeness, etc. The work presented in this thesis takes a holistic approach to the design social behaviors using descriptions of the six socio-emotional interaction categories of IPA as a model of social behaviors. Besides showing that these behaviors can help in achieving higher performance and perception outcomes, Chapter 7 also shows that this approach can be used to systematically create social capabilities for conversational agents for a variety of applications.

4.5 Implementation of Social Behavior

Table 4.4 shows an excerpt of an interaction that shows the agent's use of some of these interaction strategies. The agent is playing the role of a tutor in this collaborative learning situation. Turn 142 concludes a TuTalk script about the relationship of stress and ease of use of a wrench. At turn 144, the agent compliments (Strategy 1d) one of the students for participating in the discussion. Also note that one of the students exhibits enthusiasm about their team's designs in turns 143 and 146. These types of turns are detected using dictionaries built using a corpus analysis of collaborative learning interactions. In response, the agent exhibits cheerfulness (Strategy 2b) in turn 145 and 147 to reciprocate the student's enthusiasm.

Figure 4.1 shows the components network of the WrenchTalk tutor agent. This agent, built using the Basilica architecture, is capable of employing the social interaction strategies alongside the task-related (tutoring) behavior.

We use two specialized types of manager components (*PlanExecutor* and *SocialController*) in this network. The *PlanExecutor* is responsible for executing the tutor's task-related interaction plan, which is comprised of 37 steps. The plan is executed largely sequentially; however the plan controller can choose to skip some steps in the interest of time. The *SocialController* implements the eleven social interaction strategies listed in Table 4.3. The strategies are triggered by rules (Appendix J) based on combinations of four features: the most recent plan step, semantic annotations of the most recent student turns (by the *AnnotationFilter*), activity levels of each student (measured by the *ActivityDetector*) and the percentage of tutor turns generated by *SocialController* to regulate the amount of social behavior by the tutor. Once the controllers determine a step or a strategy that is to be generated they are conveyed to the corresponding observable behavior generation components (*PromptingManager*, *TutoringManager*, *IntroductionsManager*).

139	**Tutor**	One last thing on this topic, Does more (or less) stress in a wrench make it easier to use?
140	**S95**	no change?
141	**Tutor**	You are correct about that.. Stress doesn't determine ease of use.
142	**Tutor**	It's the moment achieved by the wrench that determines the ease of use.
143	**S89**	yay!
144	**Tutor**	It's good to have your contributions in the discussion Jackie :-)
145	**Tutor**	Go team :-)
146	**S89**	Go team yay
147	**Tutor**	I am happy to be working with our team
148	**S89**	Me too
149	**S95**	whoa the bot knows my name
150	**Tutor**	Unfortunately maximum stress (12800) in our design1 is way above the maximum allowed stress (i.e. 8750)
151	**Tutor**	This wrench cannot be safely used!

Table 4.4: Excerpt of a conversation between three students and WrenchTalk Tutor

The flexibility that the Basilica architecture provides is demonstrated in this case by the use of the two primary controllers (*PlanExecutor* and *SocialController*), unlike typical dialog systems, which use only one primary controller (Dialog Manager). Coordination between the two controllers is achieved by connecting them. For instance, when the *PlanExecutor* is working, it blocks the *SocialController* and vice versa. Control is shared between the two by transferring control at the end of every interaction step. We also notice the re-use of the tutoring components from the CycleTalk tutor agent discussed in Chapter 3

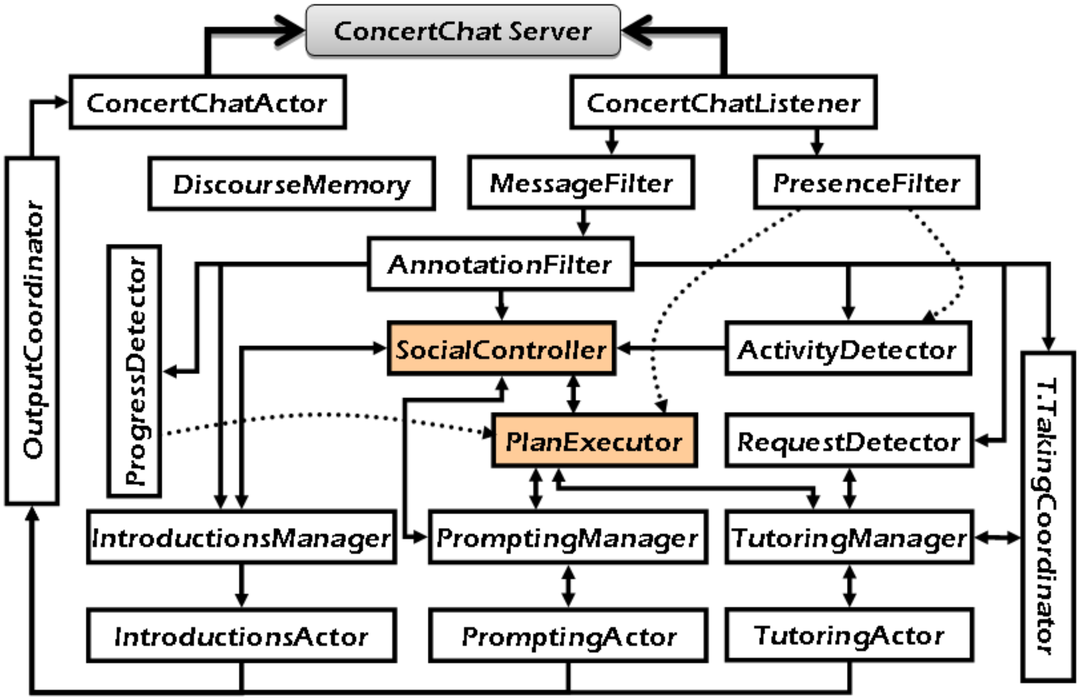


Figure 4.1: Component Network of the WrenchTalk Tutor

The *ActivityDetector* component provides a way to monitor the uneven student participation by students which is characteristic of multi-party interaction. When one or more students become critically inactive in the interaction, this component can choose to trigger Strategy 1e listed in Table 4.3 through the *SocialController* to encourage the inactive students to contribute to the discussion.

An additional type of component we see in this network is the memory component (*DiscourseMemory*). It maintains a history of the discourse state messages and interaction steps. Memory components circumvent the usual event propagation mechanism by using a specialized faster commit / retrieve mode of access from other components. This is to reduce the computational overhead of using two events (request and response) otherwise required for accessing memory components.

An experiment, described in the next chapter, conducted to evaluate the WrenchTalk tutor also demonstrates the usefulness another feature of the Basilica architecture, i.e., Observer components. In order to compare our implementation of the social interaction strategies with human quality social interaction, we augmented the tutor with a human observer user interface that allowed a human tutor to insert social prompts at any time during the interaction with the students. The students would see that human modified prompt as another prompt from the tutor. This was implemented by assigning an observer to the *AnnotationFilter* and *PlanExecutor* components.

4.6 Alternative Perspectives

While the rest of this thesis focuses on the use of the social interaction strategies based on IPA categories, in this section, we consider alternative perspectives from recent work in the area of small group communication. Bales (1958) suggested the emergence of leaders in group interaction that serve the roles of task and social experts/leaders. Recent views on shared leadership (Pearce and Conger, 2003) discussed in contrast to traditional approaches to leadership (e.g. hierarchical) suggests the distribution of leadership among the participants of a groups wherein the participants of the groups influence each other to support the achievement of shared goals.

The shared leadership paradigm is consistent with the emergent task and social roles differentiation suggested by Bales. IPA provides a mechanism to study the emergence of these expert roles during group interaction by tracking contributions of each participant across the categories corresponding to each role. Relationships between the previous and emerging leaders of each role have been studied in terms of their initiation and reception of contributions across the various interaction categories.

Within the context of shared leadership, the concept of self-in-relation described in the Stone Center Relational Theory (Miller, 1976) allows us to consider a unified model of relationships between the participant's goals towards themselves as well as towards the group. While IPA does not consider self as a recipient of a contribution, interaction categories may be mapped to objectives directed at self (e.g mitigating face threats) or at the group.

While we do not apply these concepts within the analysis of the experiments discussed in this thesis, they offer directions for further integration of research in the fields of small group communication and conversational agents.

Chapter 5

Application: Collaborative Learning

Availability of an implementation of the Social Interaction Strategies for Conversational Agents presents the opportunity to investigate a collection of interesting research questions. Foremost of these corresponds to evidence of impact of using these Social Interaction Strategies. Besides the efficacy of using these strategies, we want to investigate the appropriate use of these strategies in terms of amount and timing of these strategies. Finally, we are interested in investigating the generalizability of these social behaviors to other multi-party interactive situations.

In this chapter, we will describe two experiments that were conducted to investigate the benefits and the optimal amount of the social interaction strategies listed in Table 4.3. Chapter 6 describes our work on investigating the appropriate use of these strategies in terms of timing considerations. The issue of generalizability is explored in Chapter 7 where we consider the use of socially capable conversational agents to support a group decision making application.

5.1 Methodology & Metrics

Before we discuss the experiments, this section provides a general description of the methodology we adopt in this line of work.

5.1.1 Recruitment

Subjects participating in our experiments are recruited either from public announcements or through voluntary consent of students participating in courses that employ our instructional agents. In the experiments described in this chapter, students enrolled in freshmen and sophomore mechanical engineering courses interact with our agents during one of the labs of each of these courses. All the enrolled students participate in this lab and go through the same procedure including interacting with the agent. Subjects are compensated through gift cards or lab credit depending on the mode of recruitment.

5.1.2 Design

Our experimental designs generally involve conducting controlled experiments with a small number (3-6) of conditions in which one or two experimental variables are manipulated. Subjects are randomly divided into groups of 2-4 participants. Groups are evenly distributed between the experimental conditions i.e. each group participates in only one of the experimental conditions.

5.1.3 Procedure

1. Pre-manipulation tests and surveys (if any) are administered.
2. Subjects are given instructions about the group activity along with a tutorial on the collaboration environment. Subjects are not made aware of the experimental manipulation.
3. Subjects participate in the group activity.
4. Post-manipulation tests and surveys are administered. While subjects work in groups during step 3 of this procedure, tests and surveys in step 1 and step 3 are administered individually.

5.1.4 Materials

We will discuss specific materials used for each of the experiment as we present the experiments. The same perception survey (Appendix B) has been used in all of these experiments. The survey comprised of nine items to be rated on a seven point Likert-scale ranging from Strongly Disagree (1) to Strongly Agree (7). Six of the items were based on Burke's (1967) survey rephrased to elicit ratings about the tutor's behavior. The other three questions were designed to elicit ratings of task satisfaction, satisfaction with group discussion and perceived task legitimacy.

5.2 Experiment 1: Benefits of Social Behaviors

Foremost, we are interested in studying the productivity and perception effects of the use of social interaction strategies by conversational tutors in collaborative learning settings. This section describes the conditions, procedures and results from a controlled experiment we conducted to validate our hypothesis that socially capable tutors will be able to perform better in collaborative learning scenarios.

5.2.1 Experimental Design

58	**Tutor**	Lets talk about making the wrench easy to use.
59	**Tutor**	Are you ready?
62	**S19**	o yeah
64	**Tutor**	Intuitively, if you wanted to make the wrench easier to use, would you make it longer or shorter?
67	**S5**	longer
70	**Tutor**	That's right. A longer wrench is better.
72	**Tutor**	Why is a longer wrench easier to use though! Let's look at the concept of Moment.
73	**Tutor**	When you use a wrench to turn a bolt, do you want a higher or lower moment?
74	**S16**	higher
75	**S13**	higher
77	**Tutor**	That's right. A higher moment means the Wrench turns easier.
79	**Tutor**	We know that moment = force x length. So...

Table 5.1: Excerpt of a tutor providing a lesson to a team of four students

We designed a study with three conditions. In the experimental condition (*Social*), students interacted with a tutor that was equipped with the eleven social interaction strategies, unlike the control condition (*Task*) which is our lower baseline condition. In a third (gold standard) condition, a human tutor controlled the social interaction while the students interacted with a tutor similar to the *Task* condition. The human tutors used in the third condition were instructed to not give any task related information/instructions. They were asked to trigger appropriate social prompts (from the same list the automated tutor uses) when they thought it was appropriate. Human tutors were allowed to make modifications to the prompts before triggering them. They were also allowed to type in new prompts.

In all three conditions, students would receive the same task related information (instructions / lessons / feedback) through the automated tutor. As shown in the excerpt in Table 5.1, the tutor has instructional capabilities (like asking questions and giving feedback) that are found in state-of-the-art tutors that perform only task-related behavior. These instructional capabilities are common to the tutors used in all three conditions in our experiment. The time allotted for the interaction is the same for each group.

Strategy 1a: Do Introductions		
7	**Tutor**	Hi, I am your tutor for today's lab.
8	**Tutor**	Lets introduce ourselves briefly. I am Avis.
9	**S083**	Hey Avis! I'm _name1_
10	**S073**	Im _name2_
11	**S089**	i'm _name3_
12	**Tutor**	Its nice to meet you all. :)
13	**Tutor**	Let's get started by reviewing the base design in your worksheet.

Strategy 3b: Show Comprehension / Approval		
48	**S083**	R we using the same material?
49	**S073**	I assume so
50	**S073**	just changing the length
51	**S089**	yeah we have multiple design steps
52	**S089**	so probably for now
53	**Tutor**	cool :)
54	**S083**	O ok

Strategy 1e: Encourage (inactive members)		
119	**Tutor**	Is this a safe wrench?
120	**S073**	and then is the same for design 2
121	**S073**	so yes the wrench is safe
122	**Tutor**	_name1_ ... any thoughts you'd like to contribute?
123	**S083**	its a safer wrench if its in steel

Strategy 1d: Compliment / Praise		
143	**S073**	high
144	**Tutor**	Right, higher yield stress is better.
145	**S089**	so steel or titanium
146	**S089**	but have fun paying for that...
147	**Tutor**	All other things being equal, you want to choose a stronger material.
148	**Tutor**	It's good to have your contributions to the discussion :-)
149	**S073**	yay

Strategy 2c: Express Enthusiasm, Elation, Satisfaction		
150	**S073**	:)
151	**Tutor**	Let's improve design 1 by using Steel for our 2nd _truncated_
152	**Tutor**	I am happy to be working with our team
153	**S083**	thanks :-)
154	**Tutor**	You can start calculating the fields in the worksheet _truncated_
155	**S089**	woo...

Table 5.2: Excerpts showing examples of the Social Interaction Strategies

The only manipulation in this design is the amount of social interaction performed by the tutors. It varies from minimal (*Task*) to computationalizable (*Social*) to ideal (*Human*). In automated tutor used in *Social* condition was the same as the one described in Section 4.5. The social behaviors of this tutor were triggered by a set of hand crafted rules (Appendix J). Table 5.2 shows some instantiations of the social interaction strategies triggered by these rules.

According to our hypothesis, socially capable tutors used in the *Social* and the *Human* conditions will perform better than the *Task* condition. We conducted a between-subjects experiment during a college freshmen computer-aided engineering lab project. 98 mechanical engineering students enrolled in the lab participated in the experiment, which was held over six sessions spread evenly between two days. The two days of the experiment were separated by two weeks.

Students were grouped into teams of three to four individuals. Each group communicated using ConcertChat (Mühlpfordt and Wessner, 2005) which is a shared workspace environment that allows participants to communicate with each other using text messages. Figure 3.1 shows a screenshot of the ConcertChat environment. No two members of the same group sat next to each other during the lab. The groups were evenly distributed between the three conditions (*Task*, *Social* and *Human*) in each session.

Each session started with a follow along tutorial of computer-aided analysis where the students analyzed a wrench they had designed in a previous lab. A pre-test with 11 questions (7 multiple choice questions and 4 short essay questions) was administered after the analysis tutorial. The experimental manipulation happened during a Collaborative Design Competition after the pre-test. Students were asked to work as a team to design a better wrench taking three aspects into consideration: ease of use, material cost and safety (Appendix C). Students were instructed to make three new designs and calculate success measures for each of the three aspects under consideration.

They were also told that a tutor will help them with the first and the second designs so that they are well prepared to do the final design. No additional details about the tutors were given. Besides receiving lab credit for participating in the design competition, students were told that every member of the team that performs best overall will receive a $10 gift card as prize.

After the students spent 35 minutes on the design competition, a post-test was administered (Appendix A). Following the test, student filled out a perception survey.

5.2.2 Learning Outcomes

The pre-test and post-test were graded by two different graders who were provided answer keys for the tests. The graders were not aware of the condition assigned to each student.

Using an ANOVA, we were unable to find any significant differences ($p = 0.680$) between pre-test scores for the three conditions (*Task*, *Social*, *Human*). Another ANOVA using test-phase (*Pre*, *Post*) and condition as independent variables showed that there was a general positive effect of the learning task indicated by a significant improvement in test scores between the pre-test and the post-test $F(1,190) = 16.67$, $p < 0.001$, effect size = 0.51 standard deviations. There was no interaction between test-phase and condition. Students in all conditions learned between the pre-test and the post-test.

To evaluate the effect of the tutor's social capability on the post-test achievement, we used an ANCOVA model with day of the experiment and the condition as independent variables. Pre-test score was used as a covariate. We found a significant main effect of the condition variable $F(2, 93) = 10.56$, $p < 0.001$. A pairwise Tukey test post-hoc analysis revealed that both the *Human* and *Social* conditions were significantly better than *Task* condition. This is consistent with our hypothesis. The *Social* and *Human* conditions were not significantly different on this measure. The relative effect sizes with respect to the *Task* condition was 0.93 standard deviations (σ) for the *Human* condition and 0.71σ for the *Social* condition. There was no main effect of day of experiment on this outcome.

5.2.3 Perception Ratings

Figure 5.1 shows the average rating by the students for the survey items about the tutor. Using condition and day of the experiment as independent variables in an ANOVA, we modeled the ratings for the items about tutor (Q1-Q6). There was a significant main effect of condition ($p < 0.05$) on the first five items i.e. liking, being friendly, providing good ideas, trying to release tension and being part of the team. We found no significant difference on the item about tutor agreeing with the students (Q6). Also, there was no main effect of day of experiment on these outcomes.

Pairwise Tukey test post-hoc analysis showed the only tutors in the *Human* condition were significantly ($p < 0.05$) better than *Task* condition for the first five questions (Q1-Q5). The tutor in *Social* condition was rated significantly ($p < 0.05$) better only for Q2 (being friendly) and marginally better ($p < 0.08$) for Q5 (being part of the team). The social tutors were not significantly better than our lower baseline (*Task*) on the other four items (Q1, Q3, Q4, Q6).

Figure 5.1 also shows the average rating about the learning task. Once again, ANOVA using condition and day of experiment as independent variables showed that there were significant main effects of condition on Q8 (task satisfaction) F(2,92) = 4.91, $p < 0.01$ and day of experiment F(1, 92) = 11.57, p = 0.001. Day2 (Mean=5.77, σ=1.56) was significant better than Day1 (Mean=4.66, σ=1.67). Also, *Social* condition was the worst of the three conditions on this measure, even though only the difference between *Human* and *Social* conditions was significant. An interaction analysis showed a marginal interaction effect of the two independent variables on this item F(2, 92) = 2.78, $p < 0.08$. There were no main effects on Q7 (satisfaction with group discussion) and Q9 (perceived task legitimacy).

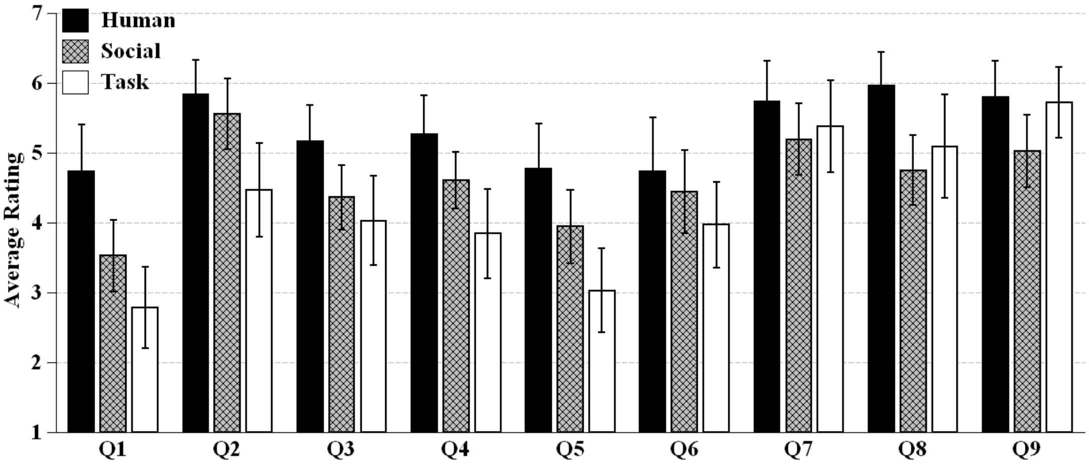


Figure 5.1: Average ratings for the Tutor (Q1-Q6) and the Learning Task (Q7-Q9)

Burke (1967) has shown that under conditions of high task legitimacy, differentiation between social and task role leaders (tutor) in the group does not happen. Assuming that effect of social behavior on task satisfaction is conditioned on role differentiation, we account for the variance due to perceived task legitimacy (Q9) by including it as a covariate in an ANCOVA to model task satisfaction (Q8) $F_{Q9}(1, 91) = 26.37$, $p < 0.001$. Using this model, we find that the only significant difference in task satisfaction among the three conditions is between the *Human* and *Task* conditions. The lower task satisfaction rating for the *social* condition is explained by the lower perceived task legitimacy in that condition.

5.3 Analysis of performed Social Behavior

In order to compare our automatic implementation of the social behaviors to the human tutors, we counted the instances of actual display of social behaviors by those tutors. The turns were classified as one of seven behaviors listed in Table 5.3 based on the social prompt closest to the turn. Table 5.3 also shows the average turn counts for the seven types of social behavior for the two types of tutors.

Behavior	Strategy	Social	Human
Doing Introductions	1a	2.67	3.80
Being Friendly	1b-1e	5.61	8.10
Doing Conclusions	1f	0.97	1.80
Trying to Release Tension	2a-2c	5.81	1.77
Agreeing	3a-3b	1.78	4.90
Pushing			0.57
Being Antagonist			1.23

Table 5.3: Average number of social behavior turns displayed by tutor

All the differences between the tutors shown in Table 5.3 are significant. We note that except the number of turns related to tension release strategies (2a, 2b, 2c), the human tutors performed significantly more social turns. Also, we note that the human tutors performed additional social behaviors that were not part of the social strategies implemented in our social tutors on some occasions. Both the Pushing and Being Antagonist behavior classify as negative socio-emotional interaction categories in Bales' IPA scheme.

Overall, we find that the human tutors performed significantly more social behavior than the automated tutors. This observation led us to investigate the effect of the amount of social behavior displayed by conversational agents. Experiment 2 and 3 explore the relationship between amount of social behavior and outcomes metrics within the context of collaborative learning application.

5.4 Analysis of effect of Social Behavior

We have observed that the use of social behavior in collaborative learning settings can help in improving learning outcomes (Kumar et. al., 2010a). This observation is not necessarily consistent with intuition. On the one hand, we could attribute this to the ability to manage the user's (student) attention towards the agent (tutor) by displaying the social behavior. At the same time, we could argue against the use of social behavior by considering it as a potential distraction from the task. As described in Experiment 2, we found that it is important to keep the amount of social behavior at an optimal level in order to observe the learning benefits of the social behavior and to avoid its distraction costs.

1	**Tutor**	Are you paying attention to this?
2	**S006**	of course
3	**S001**	yes
4	**Tutor**	What would happen if the stress in our Wrench is too high?
5	**S006**	fail
6	**S001**	it'd fail
7	**S014**	it would break.
8	**Tutor**	That's right. The wrench can break!
9	**Tutor**	A wrench that can break in normal use is not safe!
10	**S006**	:(
11	**Tutor**	In general, to keep a wrench safe, we want to keep the stress in the wrench high or low?
12	**S006**	lowwww
13	**S001**	low
14	**Tutor**	Right, stress should be kept low.
15	**Tutor**	Which material property determines when the stress is too high?
16	**S001**	yield strength?

Table 5.4: Excerpt of a Conceptual Tutoring Episode

In this section, we present a corpus analysis to discover an underlying model of how the social behavior affects learning outcomes. We used a corpus of collaborative learning interactions between teams of three or more students and a tutor. The interactions involve the teams performing a learning activity which lasts approximately 40 minutes. During this time, the tutors provide directions, hints and conceptual tutoring. Each team undergoes up to four conceptual tutoring episodes.

A conceptual tutoring episode is a system-initiated conversation during which the tutor leads the students through a directed line of reasoning to help them reflect upon a concept related to the learning activity. An excerpt of a tutoring episode discussing the relationship between stress and safety is shown in Table 5.4. No social behaviors are performed during such an episode. During the collaborative design activity, these episodes are the primary source of knowledge that the tests administered to the students measure. The tutor performance objective is to deliver the instructional content of the episodes as effectively and efficiently as possible. As discussed in section 5.4.2, we measure the performance of the tutor on these objectives using the test scores and the amount of time spent on the tutoring episodes.

5.4.1 Coding Tutoring Episodes

Each turn in all the tutoring episodes of the 32 interactions between a team of students and an automated tutor were annotated using a coding scheme described here. The tutor turns were categorized as either Respondable (TR) if the students were expected to respond to that tutor turn or Not Respondable (TU) otherwise. In Table 5.4, all the shaded turns are labeled as Respondable.

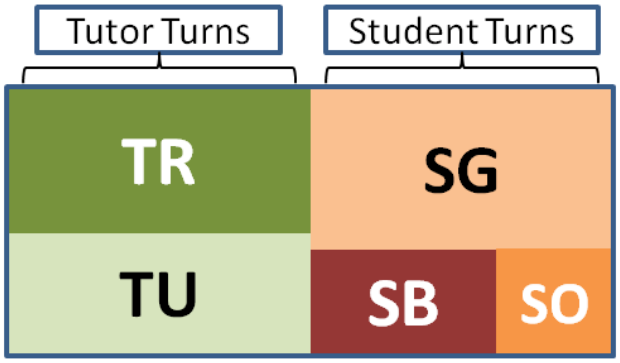


Figure 5.2: Venn Diagram of Episode Turn Annotations

Student turns are categorized into one of three categories. The Good turns (SG) label identifies turns where the students are showing attention to a respondable tutor turn (e.g. Turn 2 & 3 in Table 5.4) or the students are giving a correct or an incorrect response to a direct question by the tutor (e.g. Turns 5, 6, 7, 12, 13 & 16). Counterproductive (Bad) student turns (SB) include students abusing the tutor or ignoring the tutor (e.g. talking to another student when the students are expected to respond to a tutor turn). Student turns that are not categorized as Good or Bad are labeled as Other (SO). Turn 10 is an example of SO because it is a response to a tutor turn (9) where no student response is expected. Figure 5.2 shows a Venn diagram of the different annotations. All five categories are mutually exclusive.

5.4.2 Structural Equation Modeling

In order to discover an underlying model of how the use of social behavior affects student learning, we used a structural equation modeling (SEM) technique (Scheines et. al., 1994).

Data

To measure learning outcomes, our data comprised of scores from pre-test and post-test administered to 88 students who were part of the 32 teams whose data was annotated for this analysis. We included the count of number of good and bad responses from the students as measures of interaction characteristics of each student in our dataset. Total number of social turns performed by the tutor in each interaction was included as a characteristic of social behavior displayed by the tutor. Finally, the total amount of time (in seconds) that the students spent on the tutoring episodes was included as a characteristic of the interaction efficiency during the tutoring episodes.

Prior Knowledge

The only prior knowledge input to the model stated that the pre-test occurs before the post-test.

Discovered Models

We used Tetrad IV to discover a structural equation model in the data comprising of 6 fields (*PreTest*, *PostTest*, *GoodResponses*, *BadResponses*, *SocialTurns*, *EpisodeTime*) for each of the 88 students. Figure 5.3 shows the structural equation model discovered by Tetrad using the dataset described above. p-Value of 0.38 for this model confirms the hypothesis used by Tetrad for its statistical analysis i.e. the model was not discovered randomly. Note that unlike other statistical tests, SEM models built using Tetrad are evaluated as significant if the p-Value is greater than 0.05. The numbers on the arrows are correlation coefficients and the numbers on the boxes indicate mean values for each variable.

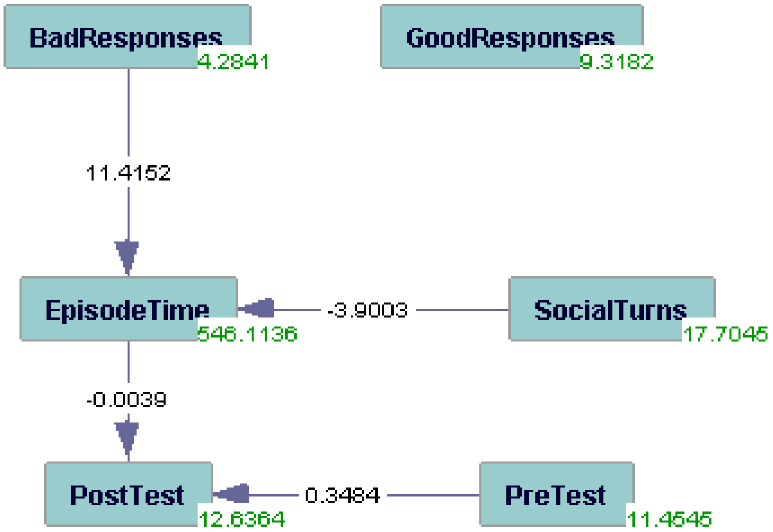


Figure 5.3: SEM discovered using all 6 variables in our dataset

Besides the expected causal effect of *PreTest* score on *PostTest* score, we find that as the duration of the tutoring episodes (*EpisodeTime*) increases, the learning outcomes deteriorate. We notice that an increase in the number of Bad responses by the students increases *EpisodeTime* indicating that students who abuse or ignore the tutor are likely to not pay attention to the instructional content presented during the tutoring episodes, hence prolonging the tutoring episode as the tutor tries to get the students through the instructional content. While it may seem that reducing the episode duration should hurt test scores because it reduces the time spent on learning. However, this is not the case because of the way tutoring episodes are conducted in our implementation. Reducing the episode duration does not reduce the instructional content being delivered. It only indicates a higher efficiency of delivery as same amount of content is being delivered in a smaller amount of time.

We observe that social behavior helps in counteracting the negative learning effect of Bad interaction behaviors of the students by reducing the *EpisodeTime*.

Tutors that perform social behavior are capable of managing the student's attention and get the students through the tutoring episode faster. Also, we note that tutor's use of social behavior does not directly reduce the amount of dysfunctional (bad) behavior of the students.

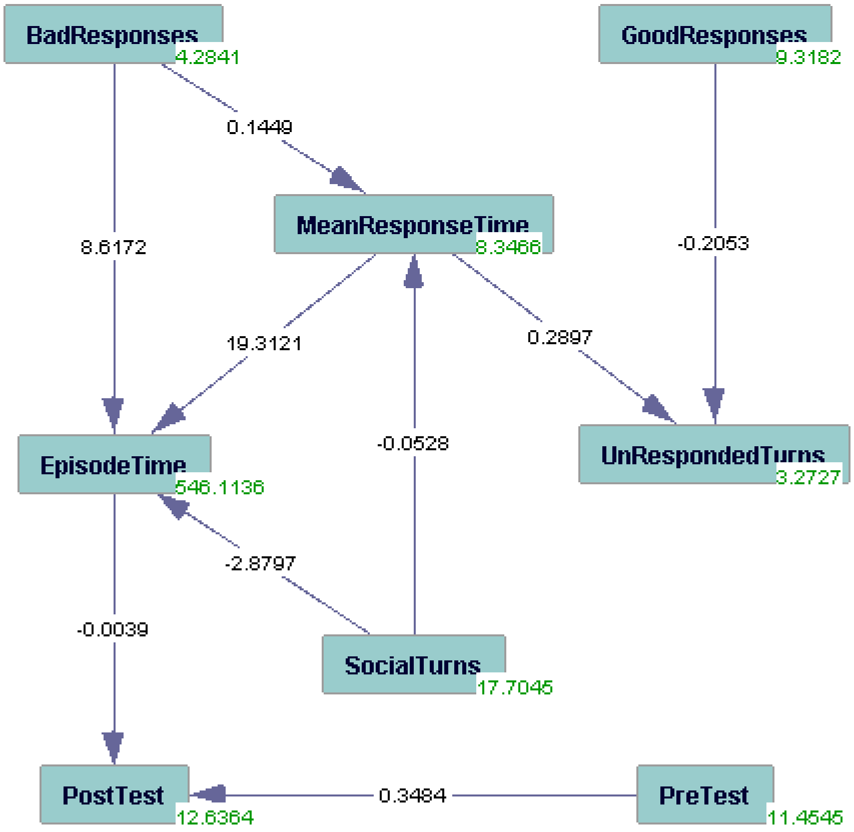


Figure 5.4: SEM including the *MeanResponseTime* and *UnrespondedTurns* variables

In order to verify this further, we experimented with two other configurations of the structural equation model. One of these configurations prohibited the relationship between *SocialTurns* and *EpisodeTime* and the other explicitly mentioned a relationship between *SocialTurns* and *BadResponses*. We found neither of the models discovered using these configurations to be statistically significant. While the relationships between those variables may become statistically significant with the availability of more data, the explanation of the effect of social behavior leading from Figure 5.3 seems to be more.

Also, we experimented with two additional variables (*MeanResponseTime* and *UnrespondedTurns*) which measured how quickly and how often the students responded to a direct question by the tutor. They act as additional measures of student attention and engagement. Figure 5.4 shows the structural equation model discovered using these variables (p=0.5264). We see that like *EpisodeTime*, *MeanResponseTime* is decreased by the use of social behavior and bad behavior by the students increases it. Also, as expected, increase in *MeanResponseTime* contributes to increasing *EpisodeTime*.

Note that the coefficients on the arrows in Figure 5.3 cannot be used to compare the magnitude of effect of the different variables on each other because of the different means and range of variation of each of the variables. We normalized the six variables used in that SEM to have a mean of 0 and a standard deviation of 1. Figure 5.5 shows the SEM using the normalized variables. The magnitude of the coefficient can be used to estimate the relative effect of two variables. For example, we see that a *PreTest* has a larger effect on *PostTest* than *EpisodeTime*.

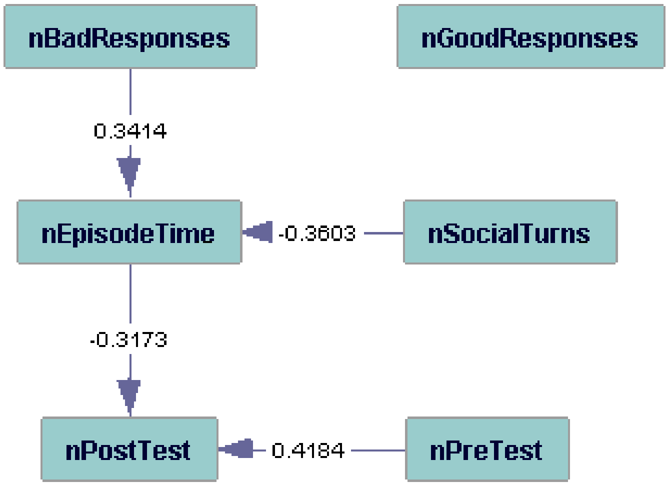


Figure 5.5: SEM with normalized variables

5.4.3 Interpretation

The SEM analysis discussed in the previous section helps us better understand the relationship between the use of social behavior and student learning in a collaborative learning setting. If we consider the duration of the tutoring episodes as an indicator of the students' attention to the tutor (higher duration⇒lower attention). We see that social behavior helps in managing the students' attention, which may be affected negatively by counterproductive/bad interaction behavior from the students.

Besides suggesting that social behavior could be a useful strategy for directing student attention, it also suggests that social behavior may not serve this function where counterproductive student behavior is not present or where it does not occur enough to negatively impact task behavior. This is because a minimum amount of time needs to be spent on each tutoring episode to deliver the instructional of the concept being discussed. In the absence of counterproductive student behavior, episode duration may be close to that minimum.

5.5 Experiment 2: Amount of Social Behavior

From experiment 1, we notice that the human capability outperformed our implementation of the social-capable automated tutor on most measures. One of the possible reasons for the shortcomings of the automated tutors on the performance and perception metrics could be attributed to insufficient display of social behavior by those tutors. Table 5.5 shows an excerpt from an interaction between a team of students and a Human tutor. Names are changed to preserve anonymity. A comparable example of a similar interaction is shown in Table 5.6 where a team of students is interacting with an automated tutor. In both cases, one of the members of the team responds (turns 65 and 82 respectively) to the tutor checking about the student's progress. The human tutor uses this event as an opportunity to encourage (Strategy 1e) the other group members to contribute (turn 66, 67) unlike the automated tutor.

The experiment described in this section attempts to investigate the effect of amount of social behavior on performance and perception outcomes in a collaborative learning situation. The effect of the amount of social behavior displayed on outcomes is our first choice for an experiment as it is one of the easiest dimensions to manipulate given our existing implementation of a socially-capable tutor. Further, if this hypothesis was shown to be valid, it has the potential to cover the most ground towards matching human social capabilities. Next, we describe how our automated tutors were modified to vary the amount of social behavior they displayed.

63	**Tutor**	Did you manage to calculate the moment already?
65	**S66**	600?
66	**Tutor**	Talk to your teammates about the design. Discussions are helpful!
67	**Tutor**	Does anyone want to respond to Timmy?
70	**S73**	yes it is. x2
71	**S73**	everything is x2
72	**Tutor**	Thank you Bobby
79	**S73**	you are welcome avis

Table 5.5: Excerpt of an interaction between a team of students and a Human tutor

80	**Tutor**	Now let's calculate the maximum stress in the Wrench.
81	**Tutor**	Note: Stress increases proportionately with the Moment.
82	**S99**	So just 8750x2?
83	**Tutor**	Since the moment doubled, the maximum stress in design 1 should be doubled too, i.e., 12800lb/in^2

Table 5.6: Excerpt of an interaction between a team of students and an automated tutor

5.5.1 Agent Implementation

The component network (20 components and 40 connections) of the CycleTalk tutor agent is shown in Figure 5.6. We can see that this component network is very similar to the component network shown in Figure 4.1. However the *GoalManager*, *GoalActor* components are new and the *ProgressDetector* component has been removed. The first two are used for performing a behavior specific to learning activity performed by students while interacting with this agent. The *ProgressDetector* is not used in this agent because none of the steps of this agent's plan depend on the detection of phrases indicative of the students' progress on the learning activity. The rest of this section describes the implementation details of this agent.

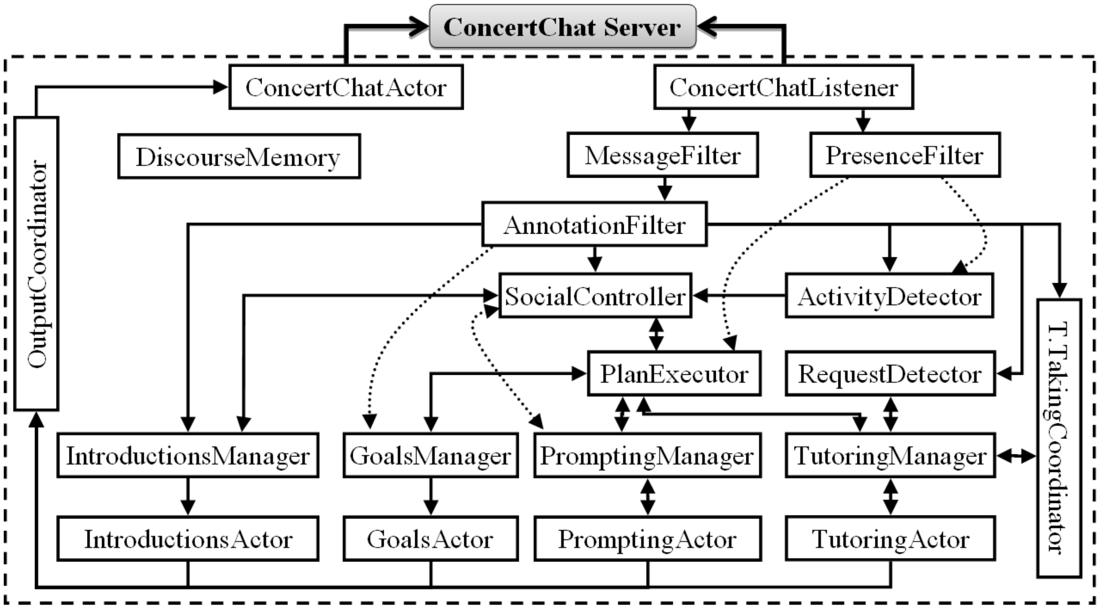


Figure 5.6: Component Network of the CycleTalk Tutor used in Experiment 2

The *ConcertChatListener* and *ConcertChatActor* components provide connectivity to the ConcertChat environment and isolate the components of the agent to allow easy integration with other environments if required. Text messages from the students are propagated through the component network after being annotated with semantic categories by the *AnnotationFilter*.

The interactive behaviors of the agent that are directly observable by the students are implemented by 4 manager-actor dyads. For example, the *IntroductionsManager-IntroductionsActor* dyad implements the introductions behavior that is performed when the Social Interaction Strategy 1a is triggered. The *PlanExecutor* and the *SocialController* trigger relevant task-related and social behavior respectively.

The *PlanExecutor* executes the tutor's task-related interaction plan comprised of 14 steps (including 4 tutorial dialogs) some of which may be skipped in the interest of time. One of the steps includes asking the students about their design goals which is performed by the *GoalsManager-GoalsActor* dyad. Based on a configuration parameter, the *TutoringManager* can favor one of the goals (or remain neutral) by choosing corresponding versions of the 4 tutorial dialogs when they are triggered.

The *SocialController* implements eight of the social strategies listed in Table 4.3. We did not use the three tension release strategies (2a, 2b, 2c) because we observed that in Experiment 1, the human tutor used these strategies very rarely. The strategies are triggered by rules based on the most recent plan step (for strategy 1a, 1d, 1f), semantic categories of the most recent student turns (for strategies 1b, 1c, 3a, 3b) and inactivity events by the *ActivityDetector* (for strategy 1e). In additional to these rules, the amount of social behavior is regulated using a Social Ratio parameter that specifies the percentage of all tutor turns that can be generated by the *SocialController*. For example, Social Ratio of 20% (as used in Experiment 1) limits the tutor to perform at most 20 turns generated by the *SocialController* for every 100 turns by the tutor.

For Experiment 2, we use two versions of this tutor with different values of the social ratio parameter. The tutor that generates lower amounts of social behavior (Low) is configured at 15% social ratio. The other version (High) is configured at 30% social ratio which is comparable to the percentage of social turns displayed by the human tutors in our earlier experiment.

5.5.2 Experimental Design

We conducted an experiment to evaluate the effect of amount of social behavior displayed by the automated tutors on performance and perception metrics. The experiment was part of a sophomore Thermodynamics lab project. 106 students enrolled in a sophomore Mechanical engineering course participated in the experiment. The students worked in teams of two to design a Rankine cycle. The experiment was conducted over 3 consecutive days of the same week. Two sessions were held each day. So, different students participated in the six different sessions. Within each session, students were randomly assigned to groups and conditions.

The procedure for the lab was divided into eight phases.

1. Students were led through a tutorial on using a thermodynamics cycle simulator called CyclePad (Forbus et. al., 1999).
2. Students read through written material on the subject of Rankine Cycle and green engineering.

3. Students used the CyclePad software to analyze the response of the cycle in terms of its efficiency, net power, waste heat and steam quality with respect to various system properties like temperature and pressure. During this phase, students followed along with our lab coordinator.

4. Following the tutorial, students filled out a motivation questionnaire (5 items) and a pre-test (30 items).

5. Students were given a tutorial on the ConcertChat collaboration software which they used in the next phase.

6. Next, the students logged into private ConcertChat rooms of their respective teams and started interacting with their teammate and an automated tutor. The students were asked to design a new Rankine cycle by choosing a set of values for the system properties in order to find an optimal output on the response variables. They were told that teams with the best designs will receive gift cards worth $20 as an additional incentive besides class credit which all participants received. To guide their design and to enable systematic interaction with the tutors, the students were asked to follow a worksheet (shown in Appendix E) which was designed to guide the students through every system property while considering its effect on each of the responses.

7. After the collaborative design phase, a post-test (29 items) was administered. They students also responded to the perception survey shown in Appendix B.

8. Finally, the students implemented the designs they came up with during the design phase individually using CyclePad. They were allowed to make further modifications to the design based on the observed responses from the simulator.

Our experimental manipulation was part of a larger experiment with multiple independent variables. The manipulation we are concerned with here is with regards to the amount of social behavior (social ratio) the tutors employed in phase (iv) were allowed to display. The student teams (dyads) were randomly assigned to one of three conditions i.e. None (0%), Low (15%) and High (30%). The corresponding values of social ratio for each of these conditions are shown in parenthesis. Conditions were evenly distributed among the teams across sessions. Each team spent the same amount of time on the collaborative design activity (35 minutes).

5.5.3 Learning Outcomes

The pre-test had one additional question than the post-test which was added to make the pre-test and post-test slightly different. This question was not used for calculating pre-test scores. Also, one of the questions on the tests was not used in calculating the test scores as it was very open-ended. Among the remaining 28 questions, 22 were objective (multiple choice questions) and 6 were subjective (brief explanation questions). The tests were graded by two graders without any information about the condition assigned to each student.

Using an ANOVA with the condition as an independent variable, we were unable to find any significant difference between the conditions on the total pre-test scores. This was also the case for the scores on the subjective questions and the objective questions. There was a significant improvement in all test scores (total, subjective and objective) between the pre-test and the post-test in all conditions, which shows that in general, the collaborative design activity was beneficial to all students. With respect to the pre-test, the relative effect sizes were 0.79 standard deviations (σ) for the total score, 0.69σ for the objective scores and 0.73σ for the subjective scores. All scores for both the pre and the post tests are shown below in Table 5.7.

Condition	**Pre-Test**			**Post-Test**		
	Total	**Objective**	**Subjective**	**Total**	**Objective**	**Subjective**
None (0%)	13.94	11.28	2.67	17.72	13.33	4.39
	(4.53)	*(2.91)*	*(2.23)*	*(4.09)*	*(2.47)*	*(2.04)*
Low (15%)	14.00	11.38	2.62	18.59	14.77	3.82
	(6.15)	*(4.16)*	*(2.54)*	*(4.72)*	*(3.43)*	*(1.74)*
High (30%)	14.08	12.03	2.06	17.72	13.75	3.97
	(4.46)	*(3.13)*	*(1.88)*	*(3.77)*	*(3.07)*	*(1.72)*

Table 5.7: Average Pre & Post test scores for each condition
(Standard deviation in paranthesis)

Using three different ANCOVA models for the three types of scores that used corresponding pre-test scores as a covariate and condition and session as independent variables, we found no significant differences between the three conditions (None, Low and High) on the total as well as the subjective scores. However, there was a significant effect of the condition variable on the objective scores F(2, 97)=3.48, $p < 0.05$. A pairwise Tukey test post-hoc analysis showed that the Low (15%) social ratio condition was marginally ($p < 0.07$) better than both None (effect size = 0.69σ) and High (effect size = 0.55σ) social ratio conditions. The difference between the None and the High conditions was not significant.

We find a similar effect on one of the learning performance metrics as reported in our previous experiment by an automated tutor with a comparable social ratio (20%). The hypothesis that performance gap between human and automated social tutors can be bridged by performing more social behavior like the human tutors does not hold in the case of learning metrics. Further, we think that the lack of significant differences on the subjective questions is because the tests were very long and the students might have focused more on the objective questions to complete most of the test. This is reflected in the relatively high scores on the objective questions (mean = 13.93) compared to a maximum of 22. In the case of the subjective questions (mean = 4.07), the maximum possible score was 11.

5.5.4 Survey Outcomes

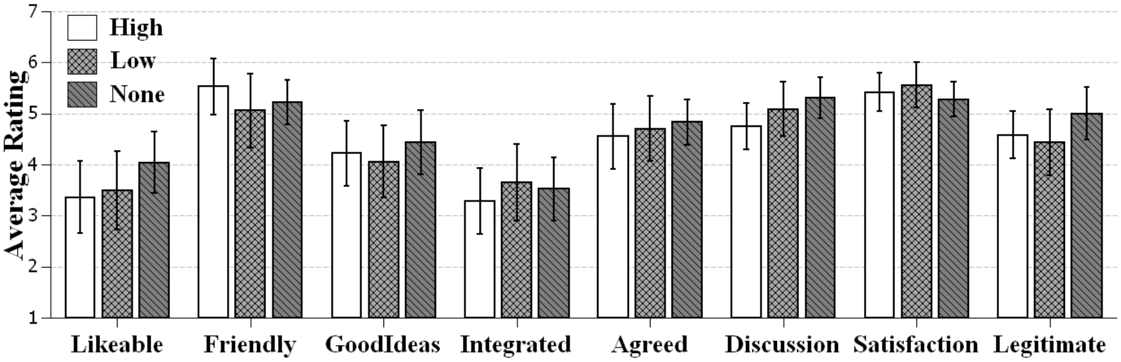


Figure 5.7: Average ratings for the Tutor and the Learning Task

We used the survey shown in Appendix B to elicit ratings about the tutor and the task from the students. However, the survey item about tension release was not used because we did not use the tension release strategies in this experiment (as mentioned in Section 5.5.1).

Figure 5.7 shows the average rating for the three types of tutors used in our manipulation. None of the differences between the three types of tutors were statistically significant for these perception measures. Once again, we note that the hypothesis that suggests performing a higher amount of social behavior to create human-like tutors does not hold for these measures.

5.5.5 Exposure Effect with Tutors

An additional analysis we were able to perform with the data available from this study was the effect of multiple exposures to automated tutors. Since our studies with engineering students span multiple years and classes, we were able to determine that 27 of our 106 participants had participated in a pilot study in a previous semester. The pilot study employed interaction with automated tutors (with no social capabilities) to teach the students about freshmen mechanical engineering concepts like relationships between forces, moments and stress. By including prior exposure as a binary (yes, no) pseudo-independent variable in the ANCOVA used to model learning outcomes on the objective questions (as described in Section 5.5.3), we found a significant interaction between the condition and the prior exposure variables ($F(2, 94) = 3.68$, $p < 0.05$). Figure 5.8 shows the interaction plot for the two variables.

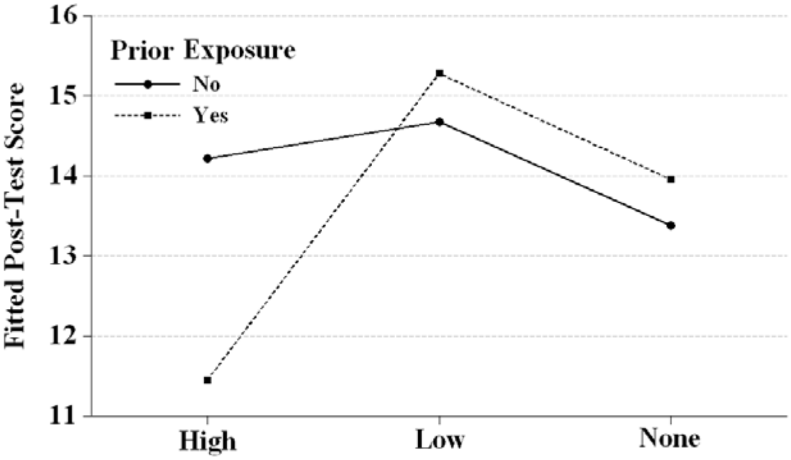


Figure 5.8: Interaction between our Experimental manipulation and Prior Exposure to Tutors

We note that tutors that display high amounts of social behavior lead to significantly poor performance for students who have had prior exposure to automated tutors. Relative to the students who do not have prior exposure to such tutors, the effect size is 1σ. This analysis suggest that it becomes increasingly important to choose the right amount of social behavior when creating conversational agents for repeated use or for user who may have prior exposure to such agents.

5.5.6 Estimating the Optimal Amount of Social Behavior

Up until here, we find in general that high (30%) social ratio tutors are not significantly different than tutors with no social capabilities (None). Also, in the case of students with prior exposure to automated tutors, these (High) tutors were significantly worse.

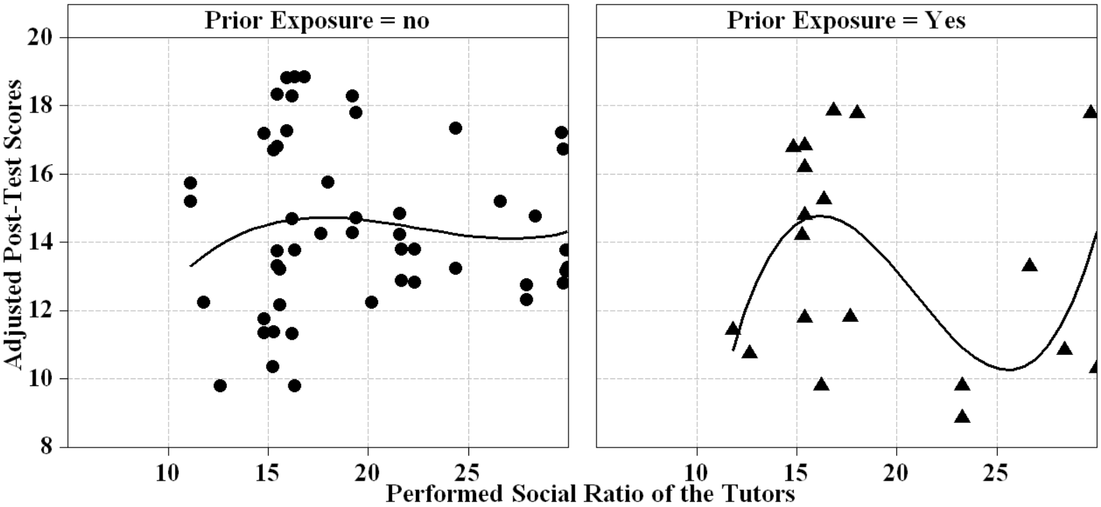


Figure 5.9: Scatter plot between Adjusted Post-Test scores and Social Ratio of the tutors in High and Low conditions

So, why do the High tutors lead to poor learning? We found that there was a significant effect of condition on the number of tutorial dialog turns the tutor performed F(2, 98) = 5.01, $p < 0.01$. A pairwise Tukey test post-hoc analysis showed that in the High condition (Mean=76.56, s.d.=9.03) the tutor performed significantly fewer task-related turns compared to the None condition (Mean=82.42, s.d.=4.67). The dialog turns performed by the tutors in Low condition was not significantly different from either High or None conditions (Mean=80.59, s.d.=11.59). We believe that this was because the high amount of social behavior was distracting the students from the learning activity and causing delays in their progress. Fewer dialog turns led to lower coverage of domain relevant material during the learning activity, which in turn led to poor performance on the tests.

The above observations suggest the relationship between learning performance and the amount of social behavior displayed by the tutor is non-monotonic. Figure 5.9 shows cubic polynomial regressions between the adjusted post-test scores and the percentage of social turns performed by the corresponding tutors for each student. Students with and without prior exposure to automated tutors are shown separately. We see that both in the case of students with or without prior exposure to automated tutors, a maxima in performance can be found around 16% performed social ratio.

5.5.7 Summary of Experiment 2

To summarize, we find that the tutors with low social ratio (15%) perform better than the high social ratio (30%) tutors and tutors with no social capabilities on learning outcomes. On perception metrics, these tutors are not significantly different from each other. Both these observations invalidate the hypothesis that matching the display of social behavior with human tutors in quantity will lead to human-like outcomes. Further, the learning effect between in the *Low* and the *None* conditions is consistent with the corresponding results from Experiment 1 (*Task* vs. *Social*).

The poor performance of High social ratio tutors suggests that the right amount of social interaction benefits the learning activity by keeping the group's instrumental and expressive needs fulfilled, excessive social interaction becomes a distraction and hinders the task-related interaction (dialogs about lessons in this case). This is consistent with the work of Bales (1953) with human small groups which finds that groups strive to achieve equilibrium between the instrumental (task-related) and expressive (Social-Emotional) interaction processes.

A separate analysis (Hua et. al., 2010) on the interactions between students and tutor during this experiment found that the number of abusive/negative comment made by the students about the tutor during the interaction were significantly higher in the High condition. This is indicative of the distractive effect of social behavior in the High condition. We have reported empirical values for the optimal amount of social behavior suitable for automated tutors in collaborative learning situations.

Having shown that automated tutors cannot match the performance of human tutors merely by matching the amount of social behavior displayed by the human tutors, we turn our investigation to other aspects of human social behavior display. Among the many options as next steps in improving the social capabilities of tutors, we think closer attention needs to be paid to circumstances under which human tutors choose to employ various social strategies and how the display of these strategies is intertwined with task based interaction. For example, in the excerpt shown in Table 5.5, the decision to elicit participation from other students may be relevant only if Timmy’s contribution to recent discussion outweighed contributions of the other students. Another aspect that can be potentially useful in modeling good social behavior by tutors is the study of student responses (or lack of responses) in the data we have collected from recent studies. In Table 5.5, turn 79 suggests that the tutor’s social behavior in turn 72 (Thanking Bobby) was appropriate.

Chapter 6

Triggering Policy for Social Behavior

The results of Experiment 1 demonstrate that the use of social interaction strategies help the agents participating in collaborative learning interactions achieve higher learning outcomes. The eleven social interaction strategies utilized by the tutor agents in that experiment were automatically triggered using a set of rules that are based on features of the discourse such as the lexical content of recent student turns, the most recent state of the agent's discourse planner, activity levels of the individual students as well as the group as a whole, and the number of social turns displayed by the agent per 100 contributions (Kumar et. al., 2010b).

These rules together form the agent's triggering policy for the social interaction strategies. The effectiveness of this triggering policy was compared to a gold-standard where human tutors were asked to choose, customize and trigger the prompts associated with each of the social interaction strategies (Kumar et. al., 2010a). Results showed that the groups with the human tutor had larger learning gains with respect to a non-social baseline. However, we were unable to find a significant difference between test scores of the students who interacted either with the human tutor or the rule-based agent. On metrics computed from a perception questionnaire, the human tutors were consistently rated better than the rule-based agents.

This observation suggests scope for further improvement and leads us to the motivation for the work presented in this chapter. We hypothesize that an agent equipped with a human-like triggering policy will be able to perform better on both performance outcomes like learning effectiveness as well as perception outcomes, which may be crucial for incorporating such agents in situations involving long term or recurring interactions with users in various day to day multi-party interactive situations.

In this chapter, we will first describe our approach and efforts towards building a human-like triggering policy as the first step towards verifying the hypothesis mentioned earlier. We employ a data-driven approach to learn a triggering policy from a corpus of tutoring interactions where human tutors triggered the social behaviors. Second, we will describe a classroom experiment conducted to verify the hypothesis mentioned in the previous paragraph.

6.1 Modeling Human Social Behavior

6.1.1 Data

As described in Chapter 5, our work is situated in the area of collaborative discussion in synchronous chat where a facilitator is managing or regulating the interaction. Specifically, in this case groups of three or more students work together on a collaborative design exercise under the supervision of a tutor. The data we use for this work consists of a collection of ten such transcripts. The students were all freshmen enrolled in a *Fundamentals of Mechanical Engineering* course. The collaborative design exercise involved designing a *better* wrench by changing the dimensions and materials of a wrench specification initially provided to them. The students were asked to take into account concepts of force, moment, stress, safety and cost while working on their design.

As mentioned, a tutor played a facilitating role in each team's chatroom. The tutor performed two primary functions. First, it provided instructions and brought up relevant concepts as the students worked on the design exercise. This function was fully automated. Second, the tutor performed social behaviors (Kumar et. al., 2010) mentioned earlier. The display of these social behaviors was controlled by a human tutor who was asked to select prompts corresponding to various social behaviors, modify the prompts if need be, and insert the prompts into the chat interaction. In the chatroom, the students saw the messages corresponding to both functions as messages from the same person (i.e. the tutor).

Annotation

In order to generate labels that identify the function of each of the tutor turns in the data, we collected annotations for each of the tutor turns using *Amazon Mechanical Turk*. Appendix I shows a screenshot of the annotation interface. The details of this annotation task are described next.

The annotation task was presented as a task involving classification of tutor turns. Each tutor turn to be classified was displayed along with a history of up to seven previous turns from the interaction. The turn to be classified was highlighted in a different color. The annotators were asked to evaluate the highlighted tutor turn in terms of whether it fit any of the following five categories.

Being Friendly (F)	Social
Relieving tension (TR)	
Agreeing (A)	
Showing Social/Emotional problems (N)	
Helping the students learn (T)	

Table 6.1: Labeling Categories

Description of the various ways behaviors corresponding to each of these categories could be realized was provided along with the list of categories. Categories {F, TR, A, N} correspond to social behaviors, and T category corresponds to the instructional behaviors of the tutor. The annotators were allowed to select as many of the five categories as appropriate including none. Each tutor turn was annotated by at least five different annotators. We collected a total of 6688 annotations for 1335 tutor turns.

The multiple (n) annotations $a_k^{t_i}$ for the same tutor turn (t_i) were combined to get a single label for each tutor turn using equation (6.1). 1[x] is an indicator function that evaluates to 1 when $x \neq 0$. C_i^l, the confidence of label l for turn t_i is calculated as the fraction of annotators of t_i that selected the category label l. We choose the category with the highest confidence to label the tutor turn with the condition that the label should be above a threshold of Θ. If the highest confidence category has a confidence of less than Θ, the label defaults to task (T) (i.e. non-social).

$$\text{Label}(t_i) = \begin{cases} l_j, & C_i^{l_j} \geq \Theta \\ default, & otherwise \end{cases} \tag{6.1}$$

$$l_j = \underset{l \in \{categories\}}{\operatorname{argmax}} \left(C_i^l\right) \tag{6.2}$$

$$C_i^l = \frac{\sum_{k=0}^{n} 1\left[a_k^{t_i}(l)\right]}{n} \tag{6.3}$$

Using this method, we can compute both the label as well as a confidence measure of the label. We computed two types of labels for each tutor turn. The first type of label maps the tutor turn to one of five possible categories {F,TR,A,N,T}. We refer to this type of label as the 5-class label. The second type of label combines the social behavior categories into a single category(S) mapping each tutor turn to one of two possible categories {S, T}. We refer to the second type of label as 2-class label.

In order to measure the quality of labels obtained by combining the multiple annotations collected via Mechanical Turk, we compared the labels to annotations provided by one expert. Table 6.2 shows the confusion matrix for the 5-class labels with respect to the expert annotations (Θ=0.65).

		Expert Annotation					
		F	**TR**	**A**	**N**	**T**	
Mechanical Turk	**F**	114	18	17	6	38	**Social**
	TR	2	0	0	1	1	
	A	3	1	23	0	25	
	N	1	0	0	1	1	
	T	62	11	14	14	982	

Table 6.2: Confusion matrix for 5-class labels (Θ=0.65)

Using Θ=0.65, Cohen's kappa for the 5-class type of label is 0.53. For the 2-class labels, kappa is 0.62. For now, we will only focus on the problem of predicting when to trigger a social behavior and ignore the issue of which social behavior should be performed at the time of triggering. Hence, we will only use the 2-class labels.

The choice of Θ=0.65 is based on a compromise between the mean confidence of the labels and the number of positive examples available to the learning algorithm. As we increase Θ, the number of tutor turns that we will be labeled as Social will reduce because of the elimination criteria in equation (6.1). This will reduce the number of positive examples available to our learning algorithm. As shown in Figure 6.1, we choose the value of Θ to maximize the number of positive examples while ensuring that the mean confidence of all our labels is above 0.8. Higher mean label confidence helps us avoid the mislabeling of instructional tutor turns such as positive feedback as social turns, which some annotators may otherwise see as Agreeing or Friendly turns.

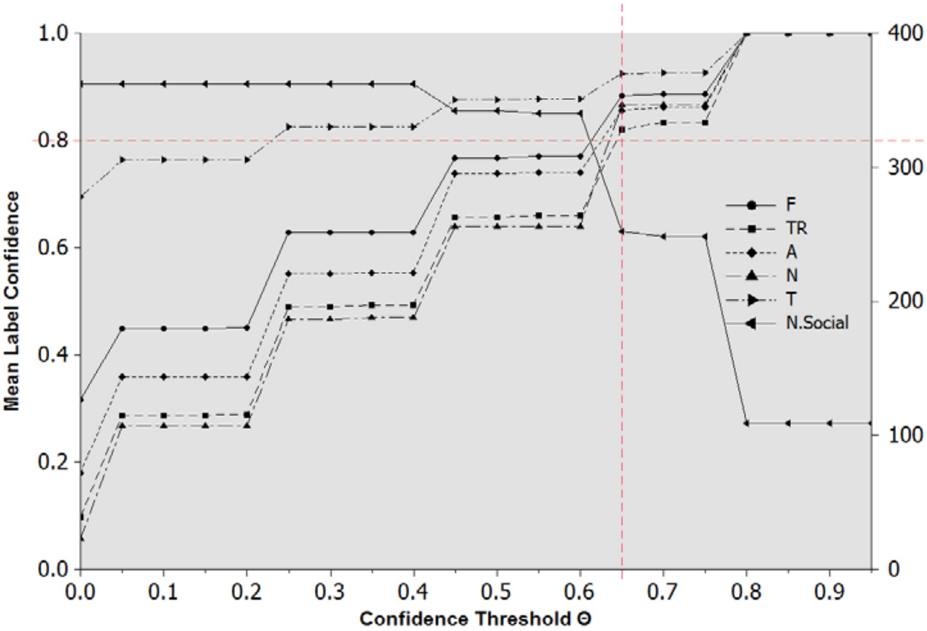


Figure 6.1: Mean Label Confidence &
No. of Social Turns for different values of Confidence Threshold Θ

To summarize the dataset discussed in this section, we have 10 transcripts of interactions between a team of students and a tutor. The transcripts comprise of a total of 2939 turns of which 1335 are from the tutor. We have labels for each of the tutor turns that indicate if the tutor turn was social. Using Θ=0.65 gives us 252 tutor turns that are social (positive examples) and 1083 task-related turns (negative examples).

6.1.2 Learning Problem

Given the data set discussed in the previous section, our objective is to learn a triggering policy that could predict when the human tutor would perform a social behavior. We consider this problem in an event-driven framework where the triggering policy has to provide a binary decision of whether to perform a social behavior at every occurrence of an event. The event-driven framework follows from the architecture (Kumar and Rosé, 2011) we use to build our agents.

In our current approach, we consider every turn in the interaction as an event. So after every turn in the interaction, the triggering policy has to consider information from the discourse up until that event (represented by features **X** discussed in Section 6.1.4) to make its decision.

$$\Psi(\,\mathbf{X}[\mathrm{e_i}]\,) \rightarrow d_i\,,\;\; d_i \in \{0,1\} \tag{6.4}$$

Formally, a triggering policy is a function Ψ that maps events ($\mathrm{e_i}$) to decisions (d_i). When executed over a sequence of events (e.g. a transcript), it produces a sequence of binary decisions $\boldsymbol{D} = \langle d_1, d_2, d_3, \dots, d_i, \dots, d_M \rangle$ of the same length such that d_i=1 if the policy decides to trigger a social behavior. In the rest of this section, we will discuss metrics for comparing triggering policies and features used to represent each event.

6.1.3 Metrics

Evaluating a triggering policy is a sequence comparison problem where we compare a reference sequence **R** with a hypothesized sequence **H**. Other problems in the field of natural language processing have developed evaluation metrics for such sequence comparisons. Research on topic/discourse segmentation (Eisenstein and Barzilay, 2008) has used a collection of metrics to compare binary sequences where 1 indicates the presence of a segment boundary. Other sequence comparison metrics such as word error rate use dynamic programming based alignment algorithm to align multi-class sequences and measure their dissimilarities (such as insertions, deletions, modifications).

Since the triggering policies we discuss here generate binary sequences of decisions, we will use the metrics used by earlier work in the area of topic segmentation. Here we will review various topic segmentation metrics that provide partial credit for near-miss segment boundaries by comparing several corresponding sub-sequences (windows) in the reference and hypothesis sequences.

$\mathbf{P_k}$ (Beeferman et. al., 1999) is the most commonly used metric in this evaluation setting. It is defined as the fraction of corresponding sub-sequences of length *k* where **R** and **H** do not share the presence or absence of a segment boundary. For example, if two binary sequences of length ten each are being compared using a sub-sequence window of length five, then $\mathbf{P_k}$ compares six difference sub-sequences. Corresponding sub-sequences are considered to be identical if both or neither have a segment boundary irrespective of the position of the boundary within each of sub-sequences. So, if three of the six sub-sequences are identical, $\mathbf{P_k}$ is equal to 0.5. This metric gives partial credit for near-miss cases within distance *k*, by allowing a fraction of the sub-sequences to be identical.

Another metric, WindowDiff **WD** (Pevzner and Hearst, 2002) measures the fraction of sub-sequences in **R** and **H** that do not share the same number of boundaries. Lower values of $\mathbf{P_k}$ and **WD** indicate better match between **R** and **H**.

Recently, Niekrasz and Moore (2010) identified biases inherent in the $\mathbf{P_k}$ and **WD** metrics by conducting formal and empirical analysis of these metrics. They suggested three new metrics: kKappa (***k*-κ**), kPrecision (***k*-prec**) and kRecall (***k*-rec**). They suggest that kKappa is an unbiased metric of evaluation in the coarse-grained binary sequence comparison case.

Furthermore, the paper suggests some diagnostic metrics that can be used to identify clumping and edge-bias in sequences. *Clumping* is a property of a sequence that has several positive decisions (triggers or segment boundaries) placed closed to one another. *Edge-bias* is another property of a sequence that places several positive decisions at the edge of a discourse. $\mathbf{P_k}$ and **WD** favor sequences that have clumping and/or edge-bias. For our application, a triggering policy that displays clumping or edge-bias is undesirable. Clumping leads to excessive social behavior being triggered and edge-bias prevents social behavior from being triggered when required. Since the ***k*-κ** metric does not favor triggering policies that display these problems, we will use the ***k*-κ** as our primary metric for evaluating triggering policies.

In this chapter, we will report results in terms of three of the new metrics (***k*-κ, *k*-prec,** ${}_\mathbf{k}$**-rec**). All of these metrics will be computed using *k*=5 which is approximately half the mean number of turns between two social turns by the tutor in our dataset. We will also report $\mathbf{P_k}$ and a metric (**ΔB**) that indicates the absolute difference in the amount of social behaviors triggered in **R** and **H**. It is important to trigger the right amount of social behavior during group interaction to achieve balance between task and social processes (Kumar and Rosé, 2010c). So, lower values of **ΔB** are desirable.

6.1.4 Features

Each event in the interaction is represented by a set of features that capture the information present in the discourse up until the event. In our current work, we are using five types of features.

Lexical features capture the content in the most recent student and tutor turns. We use a window of the most recent three student turns as well three tutor turns. Binary unigram and bigram features computed over the student and tutor turn windows were used as lexical features. No stemming or stop word removal was applied.

We used 57 *sentiment features* that were applied to the lexical content of the student turn window. These features were computed using General Inquirer (Stone, 1966) dictionaries with vocabulary size larger than 100 words. *Semantic features* indicate the presence of special phrases in the student window. These phrases were used by the rules which were used for automatic social behavior triggering (Kumar and Rosé, 2010b). We used 13 semantic features that indicate student contributions containing idea contributions, positivity, teasing, etc. Both the *sentiment* and the *semantic* features map the discourse of the students to a low dimensional space and attempt to capture social-emotional signals from the discourse.

State features represent the state of the discourse plan. These features help in capturing the task-specific characteristics of when social behavior should be performed. The 37 steps in the tutor's interaction plan are represented as binary features (Bohus et. al., 2006) indicating which step of the interaction is being executed. We included the information about the dialog state before and after the event using this feature representation. Additionally, we used a binary feature that indicates if the dialog state changed at the event.

Finally we included some *special purpose features* that capture the activity levels of the participants in the interaction. These features measure the number of chat contributions in the last 5 minutes from the tutor, the most active student and the least active student, as well as the range of the activity levels.

With these features representing the events, we now have a dataset of 10 transcripts each comprising of multiple events. For the rest of the section, we use the following notation for dataset. ${}^{t}e_i$ is the ith event of transcript t. ${}^{t}\boldsymbol{X}_i$ are the features for that event. ${}^{t}d_i$ is the reference decision label and ${}^{t}y_i$ is the confidence of social behavior at ${}^{t}e_i$.

$$DataSet = \{ (\, {}^{t}\boldsymbol{X}_i \, , \langle \, {}^{t}y_i \mid {}^{t}d_i \rangle) \} \tag{6.5}$$

Here, $${}^{t}\boldsymbol{X}_i = \text{features}(\, {}^{t}e_i) \tag{6.6}$$

$${}^{t}y_i = \, {}^{t}C_i^{social} \tag{6.7}$$

$${}^{t}d_i = \begin{cases} 1, & {}^{t}y_i \geq \Theta \\ 0, & otherwise \end{cases} \tag{6.8}$$

Given this dataset, we can formulate the learning problem in two ways. First is a binary classification problem i.e. learning Ψ as in equation (6.4). Second is a regression problem where we learn a function Ψ that maps events to a confidence of triggering social behavior.

$$\Psi(\boldsymbol{X}_i) \rightarrow y_i \tag{6.9}$$

6.1.5 Generating Social Behaviors

The focus of Section 6.1 is to learn a triggering policy that determines when an agent should perform a social behavior. However, in practical use, besides triggering the social behavior, we need to determine which one of the eleven social interaction strategies should be performed. Ideally, the triggering policy should decide this too. However, as we can see in Table 6.2, our dataset has very few examples for learning a policy at the level of granularity of individual strategies. For example, we only have four positive instances where the *Tension Release* strategies are being used.

Our implementation of the agent that uses the learnt trigger model provides a continuous stream of scores for each of the eleven strategies. The scores are computed using hand-crafted functions that use the same features used in our rule-based triggering policy (Kumar et. al., 2010b). When a social behavior is triggered, a roulette wheel selection is used to determine the strategy to be performed. Score of each strategy is used to determine its share of the circumference of the wheel. If the score of all the strategies is zero, a generic social prompt is performed.

6.1.6 Baseline Experiments

In this section, we will establish four baseline results before we present our proposed large margin learning algorithm in Section 6.1.7. These four baselines evaluate four different triggering policies using the metrics discussed in Section 6.1.3.

The first baseline uses a random (***Rndm***) triggering policy. This policy makes the triggering decision in two steps. The first step randomly generates a confidence for triggering social behavior. The second step converts the random confidence to a decision by applying a threshold ($\Theta=0.65$).

The second triggering policy (***Rules***) uses the same set of rules (Appendix J) which were used by the tutor agent in our rule-based implementation of the social interaction strategies described in Section 4.5.

Two other triggering policies are learnt from our dataset. The first of the learnt triggering policies (***Logit***) learns a classification function of the form shown in equation (6.4). We used a binary logistic regression algorithm to learn this policy.

The second learnt policy (***Linear***) makes the triggering decision in two steps similar to the (***Rndm***) policy. However, the first step uses a linear combination of the features to predict the confidence of the social trigger. This needs a function of the form shown in equation (6.9). We used a boosted (*AdditiveRegression*) linear regression algorithm (Hall et. al., 2009) to learn the weight parameters of a linear function that combines the features to obtain the predicted confidence.

	Rndm	**Rules**	**Logit**	**Linear**
k-κ	0.01	-0.09	**0.05**	0.00
k-prec	0.36	0.33	**0.45**	0.12
k-rec	**0.86**	0.35	0.24	0.01
P_k	0.60	0.52	**0.42**	**0.39**
ΔB	70.70	**3.10**	**5.80**	26.30

Table 6.3: Summary of Baseline Results

We employ a leave-one-transcript-out cross-validation approach that trains a policy on 9 transcripts and tests the policy on the 10th held-out transcript. Results reported in Table 6.3 are averaged over 10 test sets corresponding to each fold. For all the learnt policies, we use correlation-based feature selection and keep only the top 500 features computed separately over the training set for each fold.

Table 6.3 shows the results for these four baselines. The best results are highlighted in bold. All the best results are significantly better ($p < 0.05$) than the rest. On the conventional metrics $\mathbf{P_k}$, the learnt policies (***Logit*** and ***Linear***) perform significantly better than ***Rules*** and ***Random***. The learnt policies are significantly better than the ***Rules*** in terms of the unbiased metric **k-κ**. But they are not a significant improvement over ***Random***. ***Linear*** performs significantly worse than the rest on the precision metric (**k-prec**). ***Rules*** score high on recall by generating too many triggers. Both ***Rules*** and the learnt policy ***Logit*** get closest to generating the right number of triggers. However the triggers generated by ***Rules*** are quite misplaced w.r.t the reference **R** as indicated by **k-κ** and $\mathbf{P_k}$. Overall, we find that the learnt policy, ***Logit***, performs best on 4 out of 5 metrics.

6.1.7 Proposed Algorithm

Now we will present the motivation and the implementation details of our proposed learning algorithm. This algorithm learns a triggering policy similar to the ***Linear*** baselines. The first step uses the features ${}^{t}\boldsymbol{X}_i$ of the event to predict the confidence using a regression. The learning algorithm learns the features weights from the data as described in the rest of this section. The second step uses a thresholding filter like the baselines. In Section 6.1.8, we will also present a third filtering step, which we use to regulate the number of generated triggers.

Large Margin Learner

We have based our proposed learning algorithm on the online large-margin learning algorithms of Crammer and Singer (2003). We chose to use these algorithms because of the flexibility they provide to optimize the learnt regression weights over an entire discourse using one of the metrics presented in Section 6.1.3. In contrast to these algorithms, conventional regression learning algorithms optimize a loss function over individual data instances. In the case of our problem, near missed triggers are acceptable if that leads to an improvement in the metrics that evaluate the entire sequences of decisions in our discourse.

$$\Psi(\boldsymbol{X}_i) = \Phi\left(w^0 + \sum_{j=1}^{\mathrm{size}(\boldsymbol{X})} w^j \boldsymbol{X}_i^j\right) \rightarrow y_i \qquad (6.10)$$

$$T = \left\{ \left(\, {}^{t}\boldsymbol{X}_i \, , \, \langle \, {}^{t}y_i \mid {}^{t}d_i \, \rangle \right)_{i=1}^{n_t} \right\}_{t=1}^{m}$$

$$V = \left\{ \left(\, {}^{v}\boldsymbol{X}_j \, , \, \langle \, {}^{v}y_j \mid {}^{v}d_j \, \rangle \right)_{j=1}^{n_v} \right\}$$

```
1. $\mathbf{w_0} = 0$, $\mathbf{v} = 0$, $c = 0$
2. for each iteration: $k$: 1 … $K$
3.     for each training transcript: $t$: 1 … $m$
4.         for each event in t: i: 1 … $n_t$
5.             $\mathbf{w}$ = Change $\mathbf{w_c}$ using $T_i^t$
6.             $\mathbf{w_{c+1}}$ = Update $\mathbf{w_c}$ using $\mathbf{w}$ and $V$
7.             $\mathbf{v}$ += $\mathbf{w_{c+1}}$
8.             $c$++
9. $\mathbf{w_{final}} = \mathbf{v} / c$
```

Figure 6.2: Pseudo-code of our Large-Margin Learner

Since the function learnt by the large-margin learner performs the first step of our triggering policy, it takes the form of equation (6.9). Equation (6.10) shows the form of Ψ we will use in this work. We will use two different functions for Φ. First, $\Phi(x)=x$ corresponds to linear regression (Ψ^{Linear}). Second, we can use the logistic function for Φ which corresponds to a logistic regression (Ψ^{Logit}). In both cases, the large margin learning algorithm learns a set of weights (**w**) based on the training and validation data. The pseudo-code of the algorithm is shown in Figure 6.2. Training data T comprises of m transcripts. Each transcript t contains n_t events. Validation data V includes only one transcript which has n_v events.

This is similar to the pseudo-code for a generic online learning algorithm (McDonald et. al., 2005). The algorithm performs multiple iterations over each event in the training data and updates the weights being learnt. The two main operations of this algorithm are the **Change** and the **Update** operations in steps 5 and 6 respectively.

At each iteration, the **Change** operation discovers new weights $\mathbf{w}$ using the ith training instance T_i^t from t^{th} training transcript as a potential improvement over the current weights $\mathbf{w_c}$. The new weights are discovered by solving the quadratic program shown in equations (6.11) - (6.15).

${}^t\boldsymbol{X}_+$ and ${}^t\boldsymbol{X}_-$ are the weighted centroids of the positive $\left({}^t d_i = 1\right)$ and the negative $\left({}^t d_i = 0\right)$ examples respectively in the training set t. We use ${}^t y_i$ as the weight for positive examples and $1 - {}^t y_i$ as the weight for the negative examples. $\boldsymbol{\delta}^{\text{☑}}$ calculates the differences in the feature values of a training instance and its matching centroid. Similarly, $\boldsymbol{\delta}^{\text{☒}}$ calculates the differences in the feature values of a training instance and the other centroid. $\boldsymbol{\epsilon}$ is the difference between the predicted and the actual values of confidence for a training instance. α and β are parameters of the learning algorithm. They can be used to tighten or loosen the constraints.

$$\min \|\mathbf{w} - \mathbf{w_c}\| \tag{6.11}$$

s. t.

$$\mathbf{w}.\boldsymbol{\delta}^{\text{☑}} \leq \alpha M_1 \tag{6.12}$$

$$\mathbf{w}.\boldsymbol{\delta}^{\text{☒}} \geq 1 - \alpha M_1 \tag{6.13}$$

$$\boldsymbol{\epsilon} \leq \beta M_2 \tag{6.14}$$

$$\mathbf{w}.\ {}^t\boldsymbol{X}_i \geq 0 \tag{6.15}$$

$$\boldsymbol{\delta}^{\text{☑}} = \begin{cases} {}^t\boldsymbol{X}_+ - {}^t\boldsymbol{X}_i\,, & {}^t d_i = 1 \\ {}^t\boldsymbol{X}_i - {}^t\boldsymbol{X}_-\,, & {}^t d_i = 0 \end{cases} \tag{6.16}$$

$$\boldsymbol{\delta}^{\text{☒}} = \begin{cases} {}^t\boldsymbol{X}_i - {}^t\boldsymbol{X}_-\,, & {}^t d_i = 1 \\ {}^t\boldsymbol{X}_+ - {}^t\boldsymbol{X}_i\,, & {}^t d_i = 0 \end{cases} \tag{6.17}$$

$$\boldsymbol{\epsilon} = \begin{cases} {}^t y_i - \mathbf{w}.\ {}^t\boldsymbol{X}_i\,, & {}^t d_i = 1 \\ \mathbf{w}.\ {}^t\boldsymbol{X}_i - {}^t y_i, & {}^t d_i = 0 \end{cases} \tag{6.18}$$

We used an off the shelf solver[2] to solve this optimization problem. This solver was used because of its compliance with standard quadratic programming formalism and ease of integration with the rest of our learning algorithm implementation.

The four constraints shown in equations (6.12) – (6.15) guide the discovery of new weights in step 5 of the algorithm. Constraint (6.12) tries to bring the confidence of a training instance within a αM_1 margin of its matching centroid's confidence. Constraint (6.13) tries to keep the margin between the confidence and of a training instance and the other centroid at least as large as $(1 - \alpha M_1)$. Together, these two constraints push the weights so as to separate the positive examples from the negative examples by a margin of 1 in the confidence space.

Constraint (6.14) tries to bring the predicted confidence of a training instance within a margin of βM_2 of its true confidence (${}^{t}y_i$). Finally, constraint (6.15) keeps the predicted confidence of a training instance above zero (negative confidences are meaningless).

The **Update** operation in step 6 of the algorithm incorporates the newly discovered weights from step 5 into the current weights based on their performance on the validation set in terms of a desirable metric (M_3) as well as the confidence of the training instance. Currently, we use the following update rule:

$$\mathbf{w_{c+1}} = \lambda \mathbf{w} + (1 - \lambda)\mathbf{w_c} \tag{6.19}$$

where $$\lambda = \left(\Theta - {}^{t}y_i\right) + \gamma M_3 \tag{6.20}$$

In Section 6.1.9, we will report results using this algorithm for both Ψ^{Linear} and Ψ^{Logit}. In all the experiments reported in this chapter, we use $K = 2$ (number of iterations), $\alpha = 0.15$, $\beta = 0.30$, $\gamma = 1$,

$$M_1 = \sum_{t=1}^{m} P_k(\mathbf{w_c}, T^t)/m$$

---

[2] ojAlgo: http://ojalgo.org/

$$M_2 = \sqrt{\frac{\sum_{t=1}^{m} \frac{\sum_{i=1}^{n_t} \left(\left(w_c^0 + \sum_{j=1}^{\mathrm{size}(\boldsymbol{X})} w_c^j \; {}^{t}\boldsymbol{X}_i^j \right) - \; {}^{t}y_i \right)^2}{n_t}}{m}}$$

and M_3 = $P_k(\mathbf{w}, V)$. In the experiments Ψ^{Logit} is used, constraint (6.15) is not applied.

6.1.8 Social Ratio Filtering

As mentioned earlier, we used an additional filtering step in our experimental triggering policy in order to keep the amount of social behavior at appropriate levels. We measure the level of social behavior using **social ratio** which is the fraction of turns that correspond to social behavior.

Empirical studies from small group communication suggest that functional human groups have a social ratio of around 0.20 (Bales, 1950) over the entire course of the interaction. However, the actual social ratio changes based on what the group members are talking about. For example, it is reasonable to have a higher social ratio at the start of an interaction to help with the formative processes of the group.

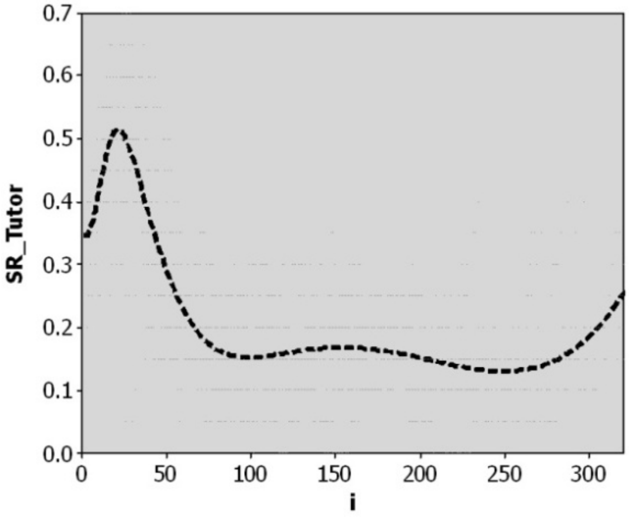


Figure 6.3: Estimated function for SRTutor

Also, the triggering policy can only control the contribution of the tutor's social turn to the social ratio. In order to capture the temporal variation of the tutor's contribution to social ratio, we built a non-linear regression model using our entire dataset. At any given turn, the tutor's contribution to social ratio (SR^{Tutor}) was computed as the fraction of social tutor turns in the tutor's last 20 turns.

We modeled the change in SR^{Tutor} as a combination of four Gaussians. Figure 6.3 shows a plot of the estimated function which we will use to regulate the number of triggers generated by our experimental triggering policies.

Instead of the using social ratio as we feature in our triggering policy learning algorithm, the use of a separate filter component for regulating the amount of social behavior was motivated by multiple practical and theoretical reasons. First of all, based on prior research in human small group communication as well as the results of Experiment 2, we are aware of the importance of performing the right amount of social behavior. The social ratio filter explicitly incorporates this knowledge into our model of social behavior triggering.

Second, the use of social ratio as a feature demands online updates to that feature during training which although not prohibitive, needs additional computation. Furthermore, the effect of the social ratio feature on the triggering decision depends on the weight assigned to that feature by the learning algorithm. While the most discriminative features are likely to be assigned sizeable weights, the fate of incorporation of this important knowledge in the learnt triggering model is dependent on the ability of the learning algorithm to discover evidence for this from our rather limited dataset. Preliminary experiments using social ratio as a feature without online updates did not lead to any improvements. Due to these reasons, we decided to use an explicitly filter to regulate the amount of social behavior.

6.1.9 Results

Table 6.4 shows results for four triggering policies with the best results from our non random baselines. As in the case of Table 6.3, best results are highlighted. The best results are significantly better than the rest.

Both the filtered large-margin learnt policies outperform everything else on the unbiased metric **k-κ**. In terms of precision, the large margin models are not significantly better than the best baseline (***Logit***). All of these models are significantly better than the ***Rules*** and ***Random*** baselines. The $\Psi^{Logit}_{Filtered}$ policy performs best on recall. However it is not significantly better than ***Rules***.

On the conventional metric $\mathbf{P_k}$, all of the learnt models are significantly better than ***Rules*** and ***Random*** baselines. Finally, $\boldsymbol{\Psi^{Logit}_{Filtered}}$ model performs as close as the rules in terms of generating the right number of social behaviors. Overall, we find that $\boldsymbol{\Psi^{Logit}_{Filtered}}$ is the best triggering policy in terms of all the five metrics we have used.

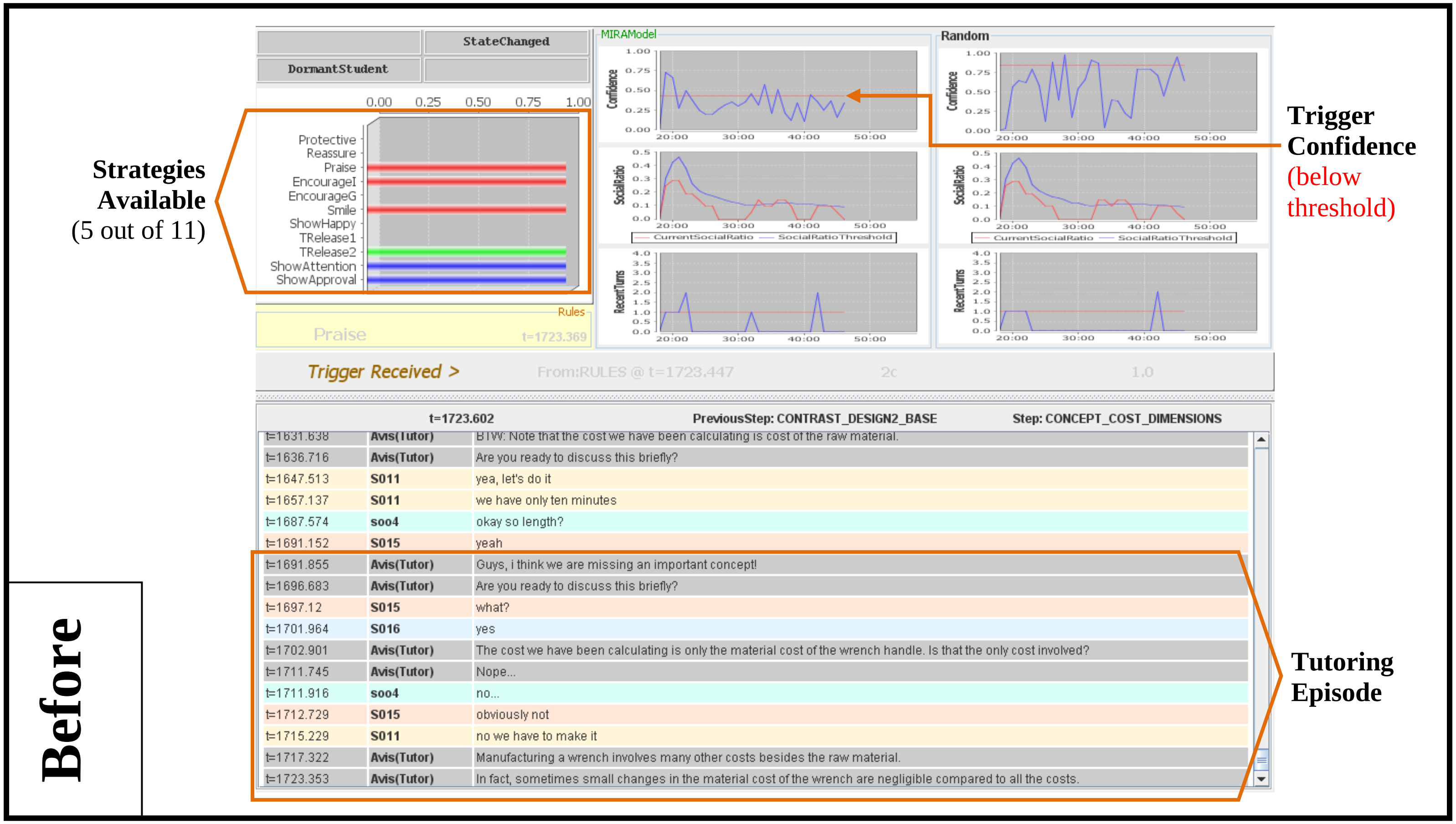
	Baseline	$\boldsymbol{\Psi^{Linear}}$	$\boldsymbol{\Psi^{Linear}_{Filtered}}$	$\boldsymbol{\Psi^{Logit}}$	$\boldsymbol{\Psi^{Logit}_{Filtered}}$
k-κ	0.05	0.08	**0.10**	0.08	**0.13**
k-prec	0.45	0.48	0.50	0.48	0.49
k-rec	0.35	0.30	0.29	0.33	**0.42**
$\mathbf{P_k}$	0.39	0.41	0.39	0.41	0.41
ΔB	**3.10**	12.57	13.13	14.38	**6.56**

Table 6.4: Evaluation results of our proposed Triggering Policies

Figure 6.4 and Figure 6.5 show an example of the use of the learnt triggering policy, the social ratio filter and the rule based scoring of each strategy (as discussed in section 6.1.5) to generate social behavior. In Figure 6.4 we see that the triggering confidence is below the allowed threshold as the interaction approaches the end of a tutoring episode. Also, we notice that five out of the eleven strategies are applicable at this time.

At the end of the tutoring episode, as we see in Figure 6.5, the learnt triggering policy generates a trigger with a confidence that is above the threshold and the social ratio filter approves of this trigger. The rules select strategy 1b (compliment/praise) and it is realized as shown in the highlighted line in the chat in Figure 6.5.

Given that we now have a triggering policy that mimics human triggering of social behavior to greater extent than our existing triggering policies, the next section describes a user study conducted to verify the hypothesis stated at the beginning of this chapter.



Strategies Available
(5 out of 11)

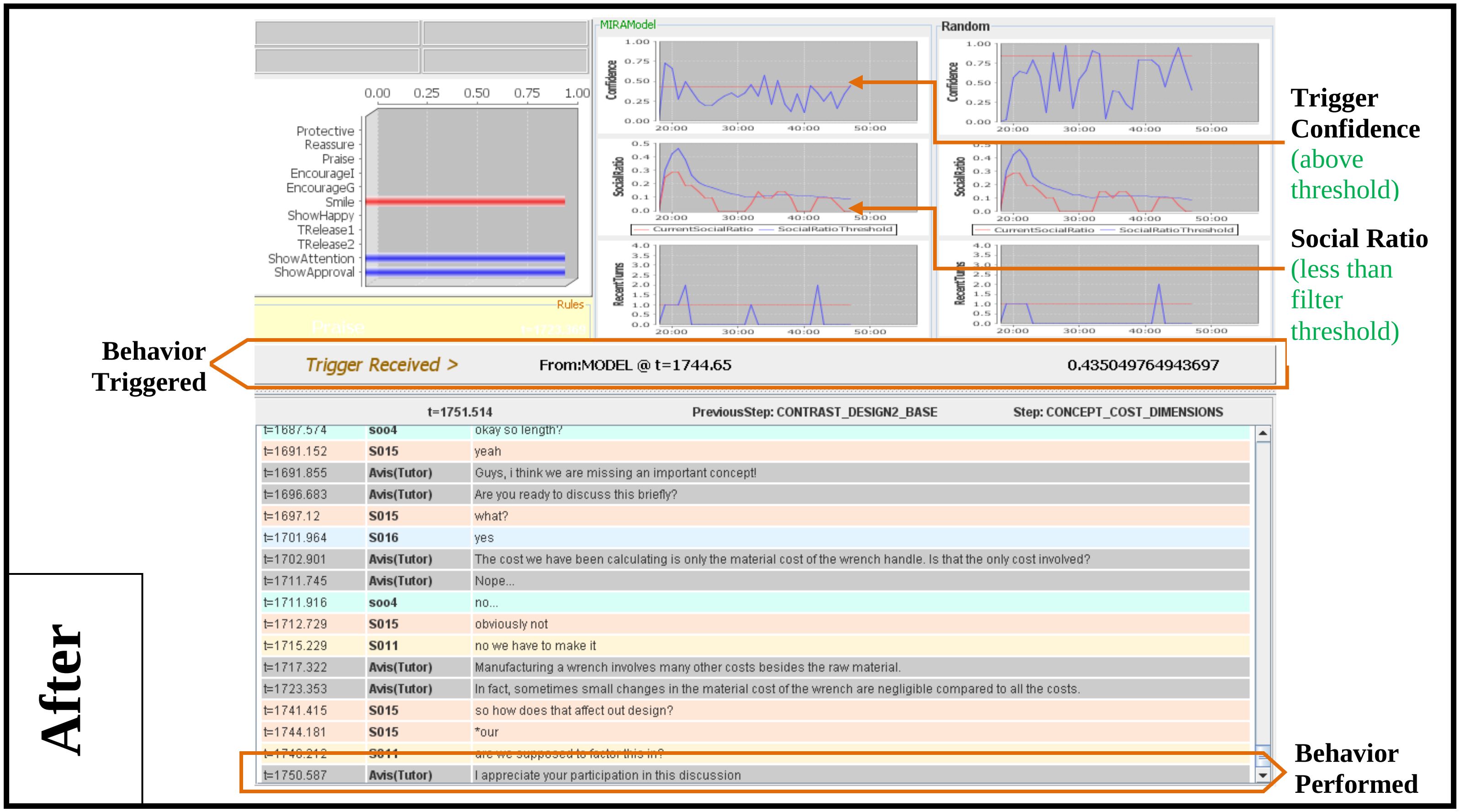
Trigger Confidence
(below threshold)

Tutoring Episode

t=1723.602 PreviousStep: CONTRAST_DESIGN2_BASE Step: CONCEPT_COST_DIMENSIONS

t=1631.638	**Avis(Tutor)**	BTW: Note that the cost we have been calculating is cost of the raw material.
t=1636.716	**Avis(Tutor)**	Are you ready to discuss this briefly?
t=1647.513	**S011**	yea, let's do it
t=1657.137	**S011**	we have only ten minutes
t=1687.574	**soo4**	okay so length?
t=1691.152	**S015**	yeah
t=1691.855	**Avis(Tutor)**	Guys, i think we are missing an important concept!
t=1696.683	**Avis(Tutor)**	Are you ready to discuss this briefly?
t=1697.12	**S015**	what?
t=1701.964	**S016**	yes
t=1702.901	**Avis(Tutor)**	The cost we have been calculating is only the material cost of the wrench handle. Is that the only cost involved?
t=1711.745	**Avis(Tutor)**	Nope...
t=1711.916	**soo4**	no...
t=1712.729	**S015**	obviously not
t=1715.229	**S011**	no we have to make it
t=1717.322	**Avis(Tutor)**	Manufacturing a wrench involves many other costs besides the raw material.
t=1723.353	**Avis(Tutor)**	In fact, sometimes small changes in the material cost of the wrench are negligible compared to all the costs.

Figure 6.4: Example of Social Behavior being generated by the Learnt Model (1)



Trigger Received > From:MODEL @ t=1744.65 0.435049764943697

t=1751.514 PreviousStep: CONTRAST_DESIGN2_BASE Step: CONCEPT_COST_DIMENSIONS

t=1687.574	**soo4**	okay so length?
t=1691.152	**S015**	yeah
t=1691.855	**Avis(Tutor)**	Guys, i think we are missing an important concept!
t=1696.683	**Avis(Tutor)**	Are you ready to discuss this briefly?
t=1697.12	**S015**	what?
t=1701.964	**S016**	yes
t=1702.901	**Avis(Tutor)**	The cost we have been calculating is only the material cost of the wrench handle. Is that the only cost involved?
t=1711.745	**Avis(Tutor)**	Nope...
t=1711.916	**soo4**	no...
t=1712.729	**S015**	obviously not
t=1715.229	**S011**	no we have to make it
t=1717.322	**Avis(Tutor)**	Manufacturing a wrench involves many other costs besides the raw material.
t=1723.353	**Avis(Tutor)**	In fact, sometimes small changes in the material cost of the wrench are negligible compared to all the costs.
t=1741.415	**S015**	so how does that affect out design?
t=1744.181	**S015**	*our
t=1748.212	**S011**	are we supposed to factor this in?
t=1750.587	**Avis(Tutor)**	I appreciate your participation in this discussion

Figure 6.5: Example of Social Behavior being generated by the Learnt Model (2)

6.2 Experiment 3: Evaluating a Human-like Triggering Policy

Here we will present an experiment we conducted to evaluate the effectiveness of various ways to trigger social behavior discussed in the previous section. This experiment is a step towards verifying the hypothesis that a human-like triggering policy could outperform a rule-based triggering policy that was used in our earlier experiments (Kumar et. al., 2010a).

We use the same interactive situation for the experiment presented here as in our earlier work. Freshmen mechanical engineering students enrolled at an American university participate in a computer-aided engineering lab that is divided into three parts, i.e., Computer-Aided Design (CAD), Computer-Aided Analysis (CAA) and Computer-Aided Manufacturing (CAM). Students practice the use of various engineering software packages for all three parts as they design, analyze and manufacture an Aluminum wrench. Our experiment is conducted during the second part (CAA) of the lab.

6.2.1 Procedure & Materials

The procedures and materials described in this section are similar to those employed in Experiment 1 (Section 5.1). The Computer-Aided Analysis lab comprises of two activities. The first activity involves analyzing a wrench design given to the students by specifying certain loading conditions and simulating the stresses and deformations in the wrench. Students are led by a teaching assistant during this activity. They spend approximately 25 minutes performing this activity. At the end of the analysis activity, the students see a simulation of the stress distribution in the body of the wrench.

After the analysis activity, a pre-test is administered. Each student spends 10 minutes working on the pre-test individually. The pre-test comprises of 11 questions, 8 of which are multiple-choice questions and the other 3 are short essay type questions.

The second activity of the CAA lab is a collaborative design activity. During this activity, students work in teams of three. Student in the same team are seated in separate parts of the lab and can only communicate using a text-based chatroom application (Mühlpfordt and Wessner, 2005). The chatroom application also provides a shared workspace in the form of a whiteboard.

After the pre-test, students are given written instructions describing the collaborative design activity. The instructions ask the students to design a better wrench in terms of ease of use, cost of materials and safety compared to the wrench they analyzed earlier. The students are expected to come up with three new designs in 40 minutes by varying parameters like dimensions and materials of the wrench. The instructions also include various formulae and data that the students might need to use for their designs. Besides course credit, the instructions mention an additional giftcard for the team that comes up with the best design ($10 for each member of the winning team).

Students are asked to log in to their respective team's chatroom. They spend the next 40 minutes working on the collaborative design activity. Besides the three students, the chatroom for each team includes an automated tutor. The tutor guides the students through the first two designs suggesting potential choices for dimension and materials for each design. As the design activity progresses, the tutor initiates four conceptual tutoring episodes to help the students reflect upon underlying mechanical engineering concepts like stress, force, moment, safety, etc., that are relevant to the design activity.

Our experimental manipulation happens during this 40 minute segment. The tutor in each team's chatroom is configured to perform social behavior using different triggering policies as specified by the condition assigned to the team. The conditions are discussed in the next section. Irrespective of the condition, each team receives the 4 conceptual tutoring episodes. Every student performs all the steps of this procedure like all other students.

At the end of the collaborative design activity, a post-test and a survey are administered. Students are asked to spend 15 minutes to first complete the test and then the survey. The post-test is the same test used for pre-test. The survey comprises of 15 items shown in Appendix B. The students are asked to rate each item on a 7-point Likert scale ranging from Strongly Disagree (1) to Strongly Agree (7). The 15 items on the survey include 11 items eliciting their perception of the tutor. 9 of the 11 items state positive aspects of the tutor (e.g. *…tutor was friendly…*). The other 2 items stated negative aspects about the tutor (e.g. *…tutor's responses got in the way…*). Besides the items about the tutor, 2 items elicited the student's rating about the collaborative design activity. The last 2 items were about the student's satisfaction with their performance on the design task.

In total, both the activities that are part of the CAA lab take approximately 1 hour 40 minutes.

6.2.2 Experimental Design

The teams participating in the experiment described here were divided into six conditions. These conditions determined the triggering policy and the amount of social behavior performed by the automated tutors. Tutors in the **None** condition did not perform any social behavior. Tutors in the **Rules** condition used the same hand crafted rule-based triggering policy (Appendix J) employed in our earlier experiment (Section 5.2). Following the results from another experiment (Kumar & Rosé, 2010c), the automated tutors in the Rules condition performed a moderate amount of social behavior (atmost 20% of all tutor turns). On average, the Rules policy triggered 25 social turns per interaction. This corresponds to the same amount of social behavior as the two Low conditions described ahead.

The **RandomLow** and **RandomHigh** conditions used a random triggering policy with a social ratio filter to regulate the amount of social behavior. In both the random conditions, the tutor would trigger social behavior using a random number generator to generate the confidence of triggering a social behavior after every turn (by a student or a tutor). In the RandomLow condition, a trigger was generated if the confidence was above 0.91. In the RandomHigh condition, a trigger was generated if the confidence was above 0.85. The triggers were filtered using the social ratio filter before generating social behavior. On average, the RandomLow condition had 23 behaviors triggered per interaction. About 37 behaviors were triggered in the RandomHigh condition.

The **LearntLow** and **LearntHigh** conditions used the best triggering policy learnt from a corpus of human triggering of social behavior as discussed in Section 6.1.9. The same social ratio filter used in the random conditions was used in these two conditions also. As in the case with RandomLow and RandomHigh, different values of a confidence parameter were used for the LearntLow and LearntHigh conditions to control the number of social behaviors triggered. On average, the LearntLow condition had 22 triggers and the LearntHigh condition had 28 triggers.

6.2.3 Results

126 students enrolled in an introductory mechanical engineering course at an Carnegie Mellon University participated in the experiment described in this section. The course was selected based on the suitability of the collaborative wrench design

activity as a curricular exercise for this course. On each day, four sessions of the Computer-Aided Analysis lab were conducted, and students attended only one assigned session. Session assignment was made based on an alphabetic split. The 126 students were divided into 42 teams. 20 teams participated on the first day of the experiment. They were evenly split into four conditions (None, Rules, RandomHigh & LearntHigh). The remaining 22 teams participated on the second day. Out of these, 5 teams each were assigned to the None and RandomLow condition. 6 teams each were assigned to the Rules and LearntLow conditions.

The rest of this section presents detailed results and analysis of this experiment. To summarize, we found that out of the six evaluated policies only the LearntLow policy that uses a triggering model learnt from human triggering data and generates a moderate amount of social behavior is consistently better than the other policies in terms of both performance as well as perception outcomes. Also, the LearntLow policy is found to be most efficient at delivering the instructional content as indicated by the smallest *EpisodeDuration* in Table 6.7.

Learning Outcomes

The learning outcomes analysis presented here shows the advantage of using a triggering policy learnt from a corpus of human triggering behavior along with a filtering technique that regulates the amount of social behavior as shown in Table 6.5.

First of all, we found no significant difference between the six conditions on the pre-test scores. As in the case of previous experiments using this learning activity, we saw that the learning activity was pedagogically beneficial to the students irrespective of the condition. There was a significant improvement in test scores between pre-test and post-test { $p < 0.0001$, $F(1,250) = 26.01$, effect-size = 0.58σ }.

The primary objective of the experiment described here was to verify the hypothesis that a human-like triggering policy could outperform a rule-based triggering policy. We used an ANCOVA analysis to compare the conditions that employed either a rule-based (*Rules*) or learnt triggering policy (*LearntLow* and *LearntHigh*). Using the post-test score as an dependent variable, the pre-test score as a covariate and the condition as the independent variable, we found a significant effect of condition { $p = 0.01$, $F(2,62) = 4.98$ } on the scores from the short-essay type questions on the tests.

A post-hoc analysis showed that the LearntLow condition was significantly better than the LearntHigh condition and the LearntLow condition was marginally better than the Rules condition { $p \approx 0.08$, effect-size = 0.84σ }. We observe that a triggering policy learnt from human triggering behavior can achieve a marginal improvement on learning outcomes compared to our existing rule-based triggering policy. This is consistent with our hypothesis.

	Mean	**St.Dev.**
LearntLow	5.12	0.54
RandomLow	5.06	0.67
None	4.75	1.13
RandomHigh	4.59	1.09
Rules	4.38	0.89
LearntHigh	3.98	1.74

Table 6.5: Mean and Standard Deviation of Adjusted Post Test Scores for Short Essay Type Questions

To further investigate the effects of other types of triggering policies, we repeated the ANCOVA described above with the data from all the six conditions of our experiment. We found no significant effect of the condition assigned to each team on the total test scores. However, there was a significant effect on the test scores of short-essay type questions using the pre-test score as a covariate and the condition as a factor { $p < 0.05$, $F(5, 119) = 2.88$ }. The adjusted post test scores for the short essay type questions and their standard deviations are shown in Table 6.5. Post-hoc analysis showed that the LearntLow condition was significantly better than LearntHigh condition { effect-size = 0.65σ }. Also, RandomLow condition was marginally better than LearntHigh condition { $p < 0.07$, effect-size = 0.62σ }.

This result further supports the observation from our earlier experiment (Kumar & Rosé, 2010c) which demonstrated that importance of performing the right amount of social behavior. Both RandomLow and LearntLow conditions employ the non-linear social ratio filter.

Perception Ratings

We averaged the student’s rating for the 11 items about the tutor into a single tutor rating measure used here. Rating on the two negative statements about the tutor were inverted (7→1, 6→2, and so on) for this calculation. We found a significant effect of condition on the tutor ratings { $p < 0.01$, $F(5,120) = 3.83$ }. Table 6.6 shows the mean and standard deviations of tutor ratings for each condition. Post-hoc analysis showed that only the Rules condition was significantly better than the RandomLow condition. Also, we found that Rules was marginally better than LearntHigh condition { $p < 0.08$ } and both Learnt-Low and None conditions was marginally better than RandomLow condition { $p < 0.08$ }.

	Mean	**St.Dev.**
Rules	4.74	1.45
LearntLow	4.56	1.58
None	4.42	1.49
RandomHigh	3.74	1.63
LearntHigh	3.55	1.26
RandomLow	3.18	0.91

Table 6.6: Mean and Standard Deviation of Tutor Ratings

While we did not see a significant improvement in perception due the use of a learnt triggering policy when compared to a rule-based triggering policy, we find an advantage over using a random triggering policy (RandomLow) which was as good as a learnt policy on the learning outcomes. The results from the tutor's perception ratings further support the importance of timing and regulating the amount of social behavior.

We did not find any significant effect of condition on the ratings about the design activity or student's task satisfaction.

6.2.4 Analysis of Tutoring Episodes

In comparison to Experiments 1 and 2, we find that the differences between the conditions in this experiment are milder in comparison. In order to further understand the results from the experiment 3, we applied the structural equation model discussed earlier (Figure 5.3) to the data collected from experiment 3.

Figure 6.6 shows the parameters of the structural equation model for our current experiment (p=0.4492). Only four variables were used because the annotations of good and bad student behavior are not available.

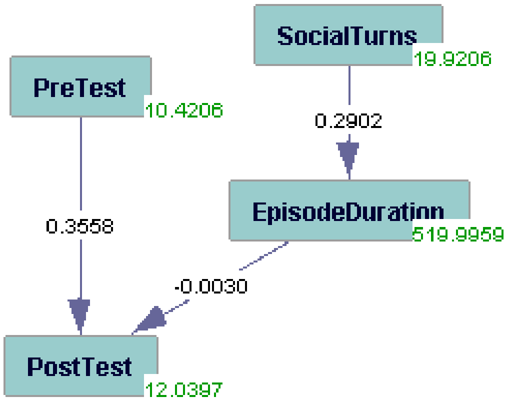


Figure 6.6: SEM applied to data from this experiment

	Mean	**St.Dev.**	$\textbf{Mean}^{\textbf{Experiment 1}}$
RandomHigh	540.80	49.50	
LearntHigh	534.80	61.00	
None	523.88	41.54	619.31 (*Task*)
Rules	519.80	102.70	543.12 (*Social*)
RandomLow	519.20	74.40	
LearntLow	484.00	69.80	
			483.89 (*Human*)

Table 6.7: Mean and Standard Deviation of Duration of Tutoring Episodes

We see that most of the model parameters (p-Value, means & correlations) are similar to parameters for the model shown in Figure 5.3. However there are two differences. First, we note that the mean of *EpisodeDuration* is smaller compared to that in Figure 5.3 which indicates that lesser counterproductive behavior was displayed by the students in this experiment. Eventhough the same learning activity was used in these two experiments; they were conducted at different times of the year. Experiment 1 was conducted in the first semester of freshmen year whereas experiment 3 was conducted in the second semester.

Nonetheless, the lower episode duration indicated that the conceptual tutoring episodes are operating closer to the minimum episode duration which leaves a smaller room for improvement by the use of social interaction strategies. As discussed in Section 5.4, we expect the social behavior to have a smaller effect on reducing episode duration in this case.

This is confirmed by the second difference between Figure 5.3 and Figure 6.6. The correlation between *SocialTurns* and *EpisodeDuration* is much smaller in magnitude compared to Figure 5.3 (-3.9). Figure 6.7 shows a structural equation model (p=0.6488) that combines that data from experiment 1 and experiment 3. The relationship between SocialTurns and EpisodeTurns in this figure is similar to Figure 5.3.

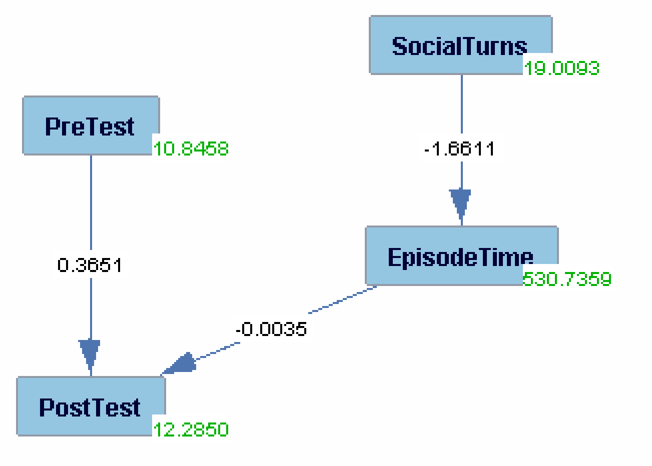


Figure 6.7: SEM from Meta-Analysis of Experiment 1 and Experiment 3

The milder differences between conditions in this experiment can be further explained using the Table 6.7 which shows the mean and standard deviations of the duration of tutoring episodes for each condition from experiment 3. The table also shows the mean episode duration from the corresponding conditions (*Task* → None, *Social* → Rules, *Human*) available in experiment 1. Note that the human condition was not used in experiment 3.

We see that in experiment 1, the attention of the group of students without any effect of social behaviors (*Task*) by the tutors was much worse than in the case of experiment 3 (*None*) as indicated by the higher episode duration for the *Task* condition. Using social behavior, our experimental tutors in the *Social* condition were able to reduce the mean episode duration by 76.19 seconds during experiment 1. However, the episode duration in the None condition in experiment 3 was already less than the episode duration for the social condition in experiment1, the rules were unable to reduce it further by much, which explains the lack of a significant learning effect between the None and the Rules conditions like experiment 1.

Table 6.7 also shows two additional interesting statistics. First, we see that both the RandomHigh and LearntHigh condition conditions which performed more than an optimal amount of social behavior added to further distracting the students. It is because of the higher episode duration for these conditions compared to the None condition. Second, we see that our learnt triggering policy with the right amount of social behavior used in the LearntLow condition was able to reduce the episode duration to almost the same values as the Human condition from experiment 1. This indicates that the LearntLow triggering policy was as good as the human tutors at maintaining student attention during the interaction.

6.2.5 Discussion

Prior work in the field of human-human interaction and human-machine interaction in the form of dialog systems has emphasized the importance of timing the display of behavior to achieve natural and/or productive interactions. In general, timing of interactive behaviors (verbal as well as non-verbal) has been studied in the context of joint activities being performed by the participants. Behaviors are timed to achieve and maintain coordination between the participants (Clark, 2005). Specifically, among other topics, timing of low-level (signal) interaction like turn-taking has been the subject of several investigations (Raux & Eskenazi, 2008; Takeuchi et. al., 2004).

On the other hand, the use of social behavior by conversational agents to support students has been proposed (Veletsianos et. al., 2009; Gulz et. al., 2010). Work in the area of affective computing and its application to tutorial dialog has focused on identification of student's emotional states and using those to improve choice of behavior performed by tutors (D'Mello et. al., 2005). Our prior work (Kumar et. al., 2010; Kumar et. al., 2007) has shown that social behavior motivated from empirical research in small group communication (Bales, 1950) can help in effectively supporting students in collaborative learning settings. Use of social interaction in other applications of conversational agents besides education has been investigated (Bickmore et. al., 2009; Dybala et. al., 2009; Dohsaka et. al., 2009).

The experiments presented here bridges these two tracks of research specifically proposing a solution to the challenge of timing social behavior in the context of a supporting collaborative learning. Compared to the work on timing signal-level joint activities like turn-taking, this work focuses on the timing of joint activities at the conversation level. The success of our algorithm at learning a model of timing conversational behaviors in the context of an interactive task could potentially offer a general approach for realizing such behaviors in other conversational agents.

6.2.6 Scope for Improvement: Error Analysis

In this section, we will discuss some of the common errors made by our triggering policy which can help in improving the approach to learn a triggering policy.

First of all, a quantitative analysis of the number of times social behavior was triggering when none of the social interaction strategies were applicable, i.e., number of times a generic social prompt is generated showed a significant negative correlation with the tutor's perception rating { coefficient = -0.20, $p < 0.05$ }. A closer integration between the triggering policy and the strategies selection mechanism would help in avoiding the generation of these behaviors. One way of addressing this problem may be to incorporate the scores of each of these strategies, computed at each turn during the interaction, as features in the learning algorithm.

Second, we notice few instances of social behavior being triggered too late making it irrelevant in the discourse context. An example of this is shown in the excerpt in the Table 6.8. Strategy 2a (Expression of feeling better) is becomes applicable at the end of second tutor turn when a significant calculation in the learning activity is completed. However, the triggering for this strategy is delayed by 72.3 second (515.9 - 443.6). While in some cases this may not be a problem if the group is still on the same topic. However, as we see in Table 6.9, the prompt is out of place after 25.3 seconds (296.9 - 271.6) when the discussion moves on to the next topic.

t=438.3	**Tutor**	So the moment achieved by the new 12" wrench is 600lb.in.
t=439.6	**S026**	wait 50% isnt 12
t=443.6	**Tutor**	That's double of what we had with the original design.
t=446.2	**S026**	half of 6 is three
t=451.0	**S020**	no 50% of 6 is 3
t=459.2	**S026**	exactly
t=465.6	**S026**	so it is 100
t=468.3	**S020**	yes
t=480.5	**S026**	cool, so what is our moment?
t=484.5	**S034**	double
t=496.3	**S020**	600
t=503.0	**S020**	moment = force times distance
t=508.5	**S020**	force = 50, distance = 12
t=515.9	**Tutor**	That's pretty good :-)

Table 6.8: Excerpt showing a social behavior being triggered too late (1)

t=251.3	**Tutor**	So, using a longer wrench, you can get a higher moment for the same effort. That makes the wrench easy to use, right?
t=260.7	**S078**	yes
t=266.1	**S071**	yes
t=271.6	**Tutor**	Let's apply this concept to the design challenge.
t=288.4	**Tutor**	To make the wrench easy to use, let's increase the length by 100%, i.e., to 12"
t=296.9	**Tutor**	It's good to have your contributions to the discussion :-)

Table 6.9: Excerpt showing a social behavior being triggered too late (2)

Finally, we observe that using the social ratio filter brings the amount of the social behavior generated by our policy closer to the amount of social behavior displayed by the human tutors. However, the social ratio filter also ensure that generated behavior is less than (or equal) to amount of human social behavior. From Table 6.4 we note that on average our filtered policy generates 6.56 fewer social turns. While we do not want to perform excessive social behavior, it might be beneficial to maintain the same level of social behavior as human tutors.

Our current implementation of the social ratio filter does not facilitate this matching in the amount of social behavior. One way of addressing this would be to dynamically shift the confidence threshold (shown in Figure 6.4) based on the gap between the current social ratio and the social ratio allowed by the filter. This would allow more triggers to be generated, when the policy is performing less than the expected number of social behaviors and vice versa.

6.2.7 Summary

In this chapter, we presented an experiment that compared the effectiveness of several social behavior triggering policies. Specifically, we compared a triggering policy learnt from a corpus of human triggering behavior to a rule-based policy which has previously been shown to be successful at triggering effective social behavior in a collaborative learning activity.

The presented experiment provides further evidence in support of the intuition that timing of social behavior and regulating the amount of social behavior are critical to improving performance and perception outcomes. A triggering policy based on human-like timing in combination with a filter that attempts to keep amount of social behavior at the same level as human tutors was shown to be marginally better than the rule-based policy on learning outcomes. Also, on perception measures, we found that the human-like policy is marginally better than a random triggering policy which uses the same filter to control the amount of social behavior. Only the learned model provides a win both on learning and on perception measures.

In order to better understand the effect of use of social behavior by automated tutors on student's learning outcomes, we presented a structured model which suggests that social behavior helps in achieving higher learning outcomes by allowing the tutor to better manage the student's attention (measured indirectly using episode duration). Following this model, we saw that a human-like triggering policy is able to achieve higher student attention as indicated by the smaller duration of tutoring episodes. Furthermore, the episode duration is comparable to that achieved by human tutors during the same learning activity in experiment 1.

Chapter 7

Application: Group Decision Making

The three experiments described in Chapter 5 and Chapter 6 study the benefits and appropriate use of social behavior by Conversational Agents for Collaborative Learning application. While we find strong results in favor of the use of social behavior for this application, we are also interested in investigating the generalizability of these benefits to other multi-party interactive situations.

In this chapter, we will describe our work on developing a socially capable agent that supports a group decision making activity. We will present an experiment conducted to verify the generalizability of the effects that socially capable agents can have on task success and perception.

7.1 Non-Combatant Evacuation Operation

7.1.1 Red Cross Rescue Scenario

In this section, we will describe an interactive situation involving a group of participants working on a decision making task. We use this situation to investigate the effects of socially capable agents. The Non-Combatant Evacuation Operation (NEO) task is a common military operation conducted to rescue, extract and evacuate non-combatant personnel under various circumstances including natural disasters, threat of hostile enemy, political uncertainty, etc. We have chosen to use an instantiation of a NEO. The Red Cross Rescue scenario (Warner et. al., 2003) was developed for the Office of Naval Research Collaboration and Knowledge Interoperability program in order to facilitate research on team collaboration and problem solving.

In this scenario, a team of participants work together to plan a course of action (COA) to rescue three Red Cross workers trapped on a pacific island in a war between the guerilla forces and the local military. The participants are assigned expert roles such as Weapons expert, Environmental expert and Intelligence expert. They are provided with extensive information regarding the scenario, the island, the guerilla forces, the local military, the condition of the Red Cross workers, their mission objective and knowledge related to their assigned expert role. The participants are asked to come up with a realistic COA that meets all the mission's objectives.

7.1.2 Procedure

A lab experiment that involves a team of participants working on the Red Cross rescue scenario is comprised of the following steps.

1. Participants complete a consent form and questionnaire that collects demographic information such as age, gender, occupation and military experience.
2. Participants are given instructions about the group decision making task. They are asked to communicate only through an Instant Messaging application. The instructions walk them through the Red Cross rescue scenario and give them information such as amount of time available to them and deliverables that are part of the COA.
3. The participants spend 12 minutes reading through the information about the island, guerilla and local military, their expert knowledge, etc.
4. Participants are given instructions to log into their instant messaging chat room.
5. The participants spend 50 minutes communicating with each other and an automated agent playing the role of an administrator to plan the COA for the scenario given to them. By the end of the 50 minutes, the participants come up with a plan comprising of six elements.

    - List of US personnel involved in the rescue
    - Any transportation they will need to get to, from and around the island
    - Any weapons they will need

- Critical timings for the mission
- A detailed plan describing the COA
- Any additional comments

6. After the group activity is completed, the participants individually complete a survey that elicits their perception about each other, the administrator agent and the task (Appendix G).
7. Finally, the participants complete a knowledge test that assesses their understanding of the mission objectives (Appendix H).

This procedure takes 90 minutes. During the entire time, participants are located in separate cubicles of the same room. They are not able to see each other. They are paid $15 for their time. They are also told that a performance reward is available to teams that come up with a high scoring plan.

7.1.3 Metrics

We use two direct metrics in this application scenario. The first one is a task success/performance metric that measures the quality of the COA generated by the team through their 50 minute interaction. The quality is measured using an evaluation rubric (Appendix F) that penalizes a COA for various types of mistakes and computes a quality score by subtracting the penalties from a maximum score of 100. The rubric includes five levels of penalties for mistakes that have different degrees of impact on the success of a mission. For example, an incorrect calculation with minimal impact carries a penalty of only 2 points. On the other hand, a calculation error with severe impact on mission success carries a penalty of 5 points.

The second metric is based on the perception survey shown in Appendix G. The metrics comprise of 18 items in total. 10 of these items elicit perception ratings of each teammate and of the agent. The other 8 items elicit ratings about the quality of discussion, task satisfaction, team cooperation, etc.

Note that a single performance metric is computed for the entire team. This is different from the metrics used in the Collaborative Learning application where the learning outcome was computed for each participant individually. On the other hand, the perception ratings are computed individually for each participant. This is similar to the case with the Collaborative Learning application.

7.2 Agent for supporting Group-Decision making

7.2.1 Agent Capabilities

neoagent	Hmmm ...
neoagent	According to your plan, are the rescuers likely to engage the enemy?
neo1	no
neo3	no
neoagent	Alright.
neo3	we will be as covert as possible
neoagent	BTW: Is this consistent with your mission objectives?
neo1	Yes
neo3	i think so, we dont want to destroy villages with missiles
neoagent	Hmmm... might help to think about how you can improve on this aspect.

Table 7.1: Excerpt of an interaction between Agent and Participants

We have developed a conversational agent that interacts with the team of participants during the 50 minutes they have for planning their COA for the Red cross rescue scenario. To situate the agent's presence in the team, the agent is assigned the role of an administrator. As the administrator, the agent's task is to provide necessary information and instructions to the team such as

1. Real time updates about the situation on the island while the team members move forward with their interaction. These updates are based on the dynamic information condition that was part of the design of the original Non-Combatant Evacuation Operation scenario (Warner et. al., 2003).
2. Reminders about the amount of time left to complete the COA

3. Reminders about taking the time to fill out the elements of the plan if the participants have not made significant progress on it.

In addition to delivering these necessary instructions to the participants, the agent tracks the decision making of the team to detect common mistakes such as not accounting for the medical needs of the Red Cross workers, using unavailable resources such as a translator, not being able to avoid detection by the enemy forces, etc. When such a mistake is detected by the agent, it brings up its concerns to the team by asking them a series of very general questions related to the concern.

The questions are framed in a very general form to allow generalizability of this capability of the agent to support planning of Non-Combatant Evacuation operation in general, without being specific to the Red cross rescue scenario. An excerpt of an interaction where the agent brings up a concern about engaging the enemy is shown in Table 7.1.

1. Showing Solidarity: Raises other's status, gives help, reward

1a. Do Introductions: Introduce and ask names of all participants

1b. Give Reassurance: When participant is discontent, asking for help

1c. Compliment / Praise: To acknowledge participant contributions

1d. Support Agreement: When teammates show approval towards each other

1e. Conclude Socially

2. Showing Tension Release: Jokes, laughs, shows satisfaction

2a. Be cheerful

2b. Highlight Disagreement: To encourage the team to address concerns that a participants might have

3. Agreeing: Shows passive acceptance, understands, concurs, complies

3a. Show attention / encouragement: To ideas from participants

3b. Show comprehension / approval: To opinions and orientations

Table 7.2: Social Interaction Strategies used by Agent
to support Group Decision Making

Besides these task-related behaviors the agent is also programmed with social capabilities similar to the social interaction strategies used for the tutor agents in Collaborative Learning applications. Table 7.2 lists the nine social interaction strategies used by the agent developed for this application. In addition to sharing many of the strategies listed in Table 4.3, we have implemented two additional strategies that are relevant to the group decision making application. Both of these strategies are triggered by presence of agreements or disagreements among the participants. Supporting agreement (Strategy 1d) and encouraging the participants to address the concerns that led to any potential disagreements (Strategy 2b) can help in regulating the social-emotional processes during the decision making interaction.

neo1	Okay, maybe we should break this problem down
neo1	let's tackle the first item
neoagent	Let's try that
neo3	re speed, checking. break down -- agreed.

neo1	We have night vision, they don't
neo1	Here, let me edit the plan real quick
neo3	sure
neoagent	Good to see that we are moving forward.
neo3	done editing for a bit?

neo3	I'm thinking we should cut our losses and come from the north. We might have casualties, but civilians won't get hurt
neo3	What do you think neo1
neoagent	It might help your plan to address any concerns your teammates have.
neo1	Ok
neo1	send everyone we have from different angles?

Table 7.3: Examples of Instantiation of Social Interaction Strategies during the NEO Group Decision Making Activity

Examples of the instantiation of some of these strategies during actual interactions with the participants performing the NEO activity in lab experiments are shown in Table 7.3. The first example shows the agent using strategy 3a to show encouragement towards a decisive act by one of the participants. In the second example, the agent demonstrates strategy 1d in response to the show of agreement/approval among the participants regarding using the tactical advantage of having night vision. Strategy 2b is exhibited in the last example in Table 7.3 where the agent highlights the concerns of one of the team members regarding civilian casualties.

7.2.2 Implementation

An agent with all the capabilities described in the previous section has been implemented using the Basilica architecture. Figure 7.1 shows the component network of this agent.

We can observe the similarities between the component network of this agent and that of other agents described in Chapter 3 and Chapter 4. The three user observable behaviors of the agent (doing introductions, prompting and short dialogs) are implemented as three filter-actor pairs. Input from the users is processed through a dictionary based annotator (*MessageAnnotator*).

Unlike the WrenchTalker agent that had two controller components that shared control, the NEO agent has three controller components. The *PlanExecutor* and the *SocialController* are similar to the corresponding components in the WrenchTalker agent. They control the execution of the interaction plan and the social behavior of the agent respectively. The triggering policy employed by the *SocialController* to trigger the social interaction strategies listed in Table 7.2 is based on a set of hand-crafted rules similar to those used by the WrenchTalker agent described in Chapter 4. The learnt triggering policy described in Chapter 6 would not generalize to be used in this scenario because it used features specific to the collaborative learning activities.

However, the NEO agent has a third controller component. The *DecisionTracker* component tracks the discussion of the users to maintain scores for the common mistakes that the team might be make during the decision making task. The *DecisionTracker* uses changes in these scores to generate events that launch dialogs that might help the users reevaluate and correct potential mistakes in their COA. An example of such a dialog is shown in Table 7.1. The list of common mistakes and the vocabularies used to track them were gradually developed during the development phase of our user studies described in the next section.

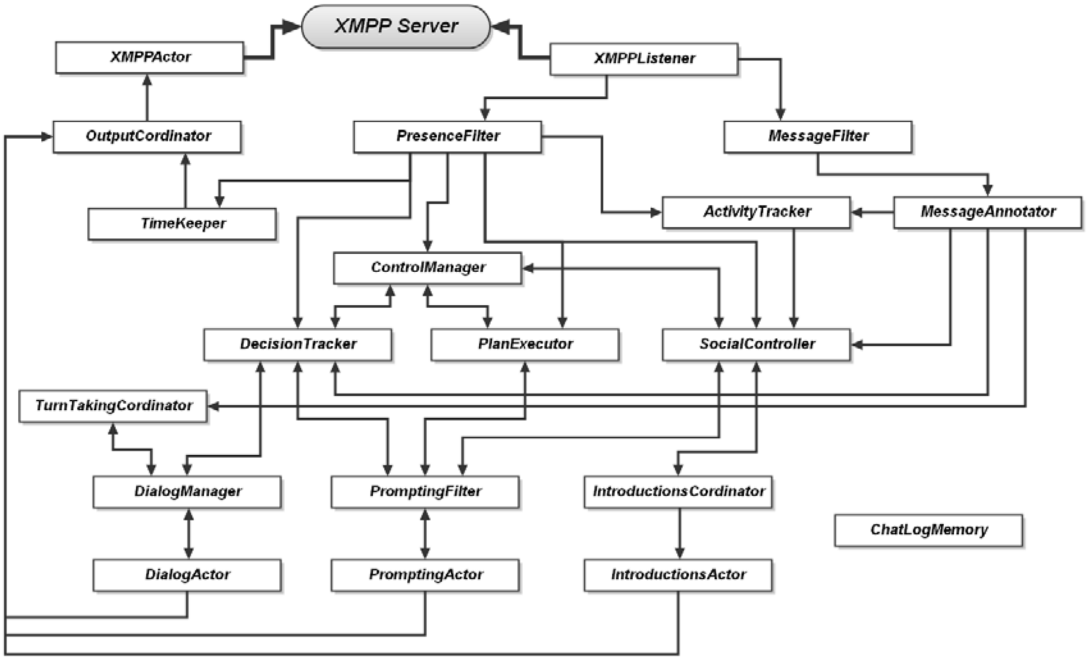
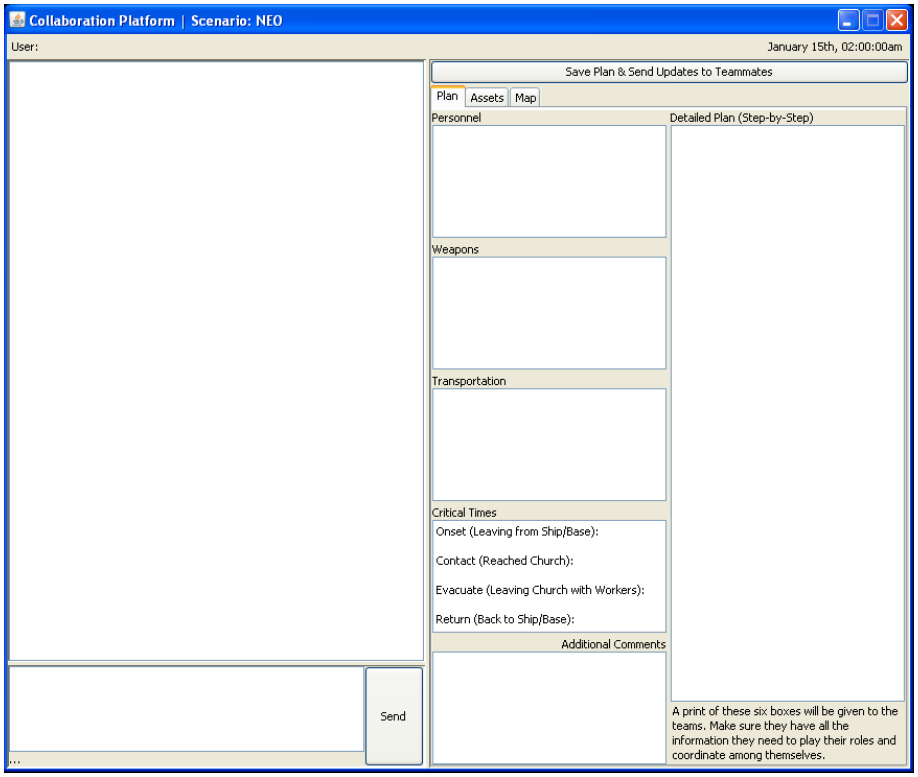


Figure 7.1: Component Network of the NEO Agent

Also, we can note that unlike the WrenchTalker agent where the two primary controllers (*PlanExecutor* and *SocialController*) shared control by alternately checking the need for control with each other, in the case of the NEO agent, control is managed through a dedicated *ControlManager* component. All the primary controllers (*DecisionTracker, PlanExecutor* and *SocialController*) request control from the *ControlManager* when they need to perform a behavior and relinquish the control as soon as their behavior is done. The *ControlManager* serves the requests for control as a queue. Furthermore, the *ControlManager* implements certain sanity checks into the control sharing mechanism by retracting control from components that might not relinquish it within a reasonable amount of time from being granted control. This serves as a mechanism for recovering from potential failures in one of the components and isolates the functionality of the rest of the component network from being affected by this failure. In general, this provides a scalable control sharing mechanism as the number of primary controllers in an agent grows.

We note that the environment listener and actor components have been replaced by *XMPPListener* and *XMPPActor* similar to the 9-1-1 Interpreter agent. We have developed a special purpose communication environment for the NEO group decision making activity that uses a server implementing the XMPP protocol as its communication backend. A screenshot of this communication environment is shown in Figure 7.2. On the left hand side, the environment provides a typical chat environment that allows the participants to communicate with each other and the agent using text messages. On the right hand side, the environment provides a shared workspace that allows the participants to collaboratively construct the course of action for the rescue. They can also refer to the list of available military assets and the island map within the other tabs on the right hand side of the environment.


Collaboration Platform | Scenario: NEO
User:
January 15th, 02:00:00am
Save Plan & Send Updates to Teammates
Plan
Assets
Map
Personnel
Detailed Plan (Step-by-Step)
Weapons
Transportation
Critical Times
Onset (Leaving from Ship/Base):
Contact (Reached Church):
Evacuate (Leaving Church with Workers):
Return (Back to Ship/Base):
Additional Comments
Send

A print of these six boxes will be given to the teams. Make sure they have all the information they need to play their roles and coordinate among themselves.

Figure 7.2: Communication Environment for NEO Group Decision Making Activity

7.3 Experiment 4: Supporting Group Decision Making

While earlier work with the NEO Red Cross rescue scenario has focused on studying communication and collaboration in teams comprising of only human participants (Letsky et. al., 2008), here we are interested in studying the benefits of using a socially capable conversational agent in addition to the human participants. In this section, we will describe an experiment similar to Experiment 1 discussed in Chapter 5.

7.3.1 Experimental Design

This controlled lab experiment compares the performance of human teams that are supported by a socially capable agent described in the previous section to that of teams supported by an agent that does not perform any of the social behavior listed in Table 7.2. We conducted a between subject experiment where each team of participants was randomly assigned to one of two conditions. The experiment followed the procedure described in Section 7.1.2. The experimental manipulation took place during step 5 of this procedure where the team of the participants interacted with each other as well as an automated agent for 50 minutes to decide on a course of action for the rescue operation.

The teams in the Social condition interacted with a socially capable agent whereas the teams in the Task condition interacted with an agent that performed only the task related functions.

7.3.2 Participants

The experiment was conducted in a closed lab with four cubicles that prevent the participants from seeing each other. Participants were recruited from the CMU Experiment Scheduling website. The website allows participants from the Pittsburgh community to enroll in experiments. An age restriction of 18 years – 36 years was applied and three participants were requested for each experimental session. If all three participants reported on time, they were randomly assigned to the three expert roles (Weapons, Environmental, Intelligence). If only two participants reported on time, the Environmental and Intelligence expert roles were combined and assigned to one of the participants and the other participant was assigned the Weapons expert role. The experiment session was canceled when only one or none of the participants reported on time.

The experiment sessions were conducted during 5 weeks spread over a duration of about four months. During this time, 37 sessions were conducted. 17 of these sessions had three participants and the remaining 20 had only two participants. Of these 37 sessions, only the last 20 sessions will be used in the analysis discussed in the next section. The two conditions that are part of our experimental design are evenly distributed between these 20 sessions. Two of the other 17 sessions were discarded because of connectivity problems that reduced the interaction time during step 5 of our procedure significantly. The other 15 were session held during the initial weeks of the experiment were discarded as development session which were used to test and fine tune the agent while it was being implemented. The agent development was frozen during the last 20 sessions and the same version of agents was used consistently through all of these sessions.

Of the 20 sessions used in our final analysis, 10 of the teams had 2 participants and the other 10 team had 3 participants. Teams of both sizes were evenly distributed between the two conditions. We found no significant differences between the distribution of age, gender, number of students and the number of people with military history (including ROTC) between the two conditions.

7.3.3 Results

As discussed in Section 7.1.3, we will analyze the results of this experiment in terms of two metrics.

Task Success / Performance Metrics

As a performance metric, we use the score that each team's course of action received based on the scoring rubric shown in Appendix F. The score is computed by subtracting the penalties assigned to the team's COA as per the rubric from a maximum score of 100. We can further subdivide the penalties into two categories. Coarse grained penalties (Error Type A or B) correspond to mistakes such as

- Ignoring to fill out a necessary element of the COA (e.g. timing, weapons, etc.)
- Choosing an unrealistic solution such as deploying tanks etc.

In contrast, the Fine grained penalties (Error Type C, D or E) correspond to minor mistakes such as a

- Calculation errors
- Spending more than 30 minutes on evacuating the Red Cross workers
- Damaging the village

Figure 7.3 shows a plot of the total score and each type of penalty for the teams in both the conditions. An ANOVA using each of these performance metrics as the dependent variable and condition as the independent variable showed that the Social condition is significantly better than the Task condition on the Total Score (higher score) as well as the fine-grained penalties (lower penalty). The difference between the two conditions for the coarse-grained penalties is not significant.

- Total Score: F(1,19)=9.21, $p < 0.01$
- Coarse-grained Penalty: F(1,19)=0.65, $p=0.431$
- Fine-grained Penalty: F(1,19)=14.82, $p<0.001$

This observation suggest that the participants make significantly fewer Type C, D and E errors when the NEO agent performs social behavior. Most of the dialogs that the agent brings up during the interaction correspond to mistakes that lead to such penalties. We believe the reason for this effect is that the participants in the NEO group decision making activity paid more attention to the agent's dialogs about these fine-grained mistakes and were able to rectify them within the time they had to come with their final COA. This reasoning follows from our analysis of the effect of social behavior on student attention towards tutorial dialogs in collaborative learning activities.

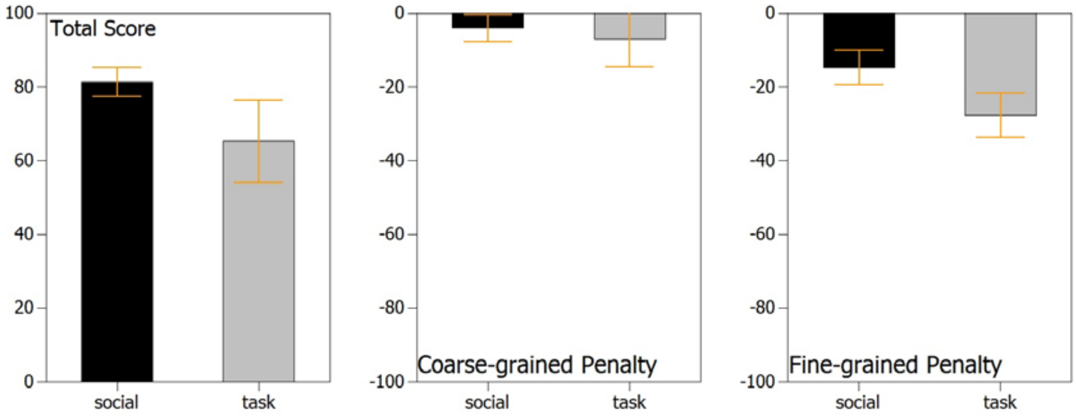


Figure 7.3: Plot of Total Score and Coarse & Fine grained Penalties

We found no significant effect of the size of the team on these metrics. In order to study the effect of the participant's understanding of the mission objectives on their performance as measured by the knowledge test (Appendix H), we performed an

ANCOVA using their individual test scores as a covariate to model their team's total score and penalties. Condition was used as the independent variable. We found no significant effect of their knowledge test scores on their team performance.

Perception Metrics

The perception metrics used in this experiment are based on a survey (Appendix G) administered to the participants after the group decision making activity. The first 10 items on the survey elicited ratings from each participant about the agent and their teammates on a 7-point Likert scale. We use the average score on these items as a participant's rating for the agent and his/her teammates. Ratings on questions 8 and 9 of the survey were inverted (7 → 1, 6 → 2, …) to match the polarity of all the questions. Figure 7.4 shows the mean agent and teammate rating for the participants in both of the conditions.

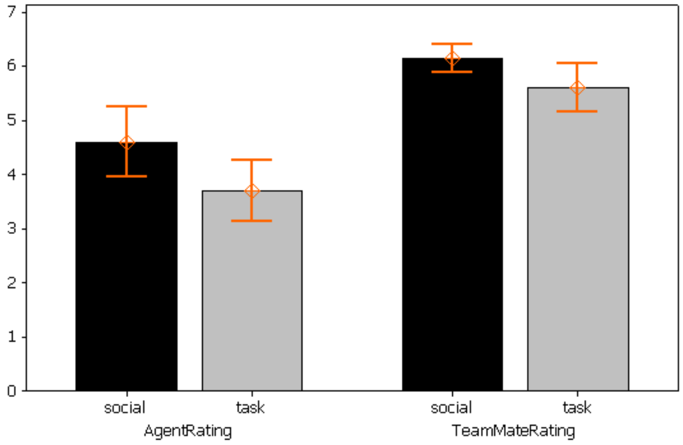


Figure 7.4: Mean Rating by the participants for the Agent and their Teammates

An ANOVA using the ratings as the dependent variable and the condition as the independent variable showed significant effect of condition on both agent as well as the teammate rating. The participants rated both their agent as well as their teammates significantly higher when the agents performed the social behaviors.

- Agent Rating: F(1,49)=4.62, $p < 0.05$
- Teammate Rating: F(1,49)=4.46, $p < 0.05$

Further, we performed an ANCOVA on the agent rating using teammate rating as a covariate and condition as an independent variable to check if the higher rating for the agent was because the participants were rating the agent relative to their teammates whom they rated higher in Social condition. We found no significant effect of teammate rating on agent rating.

While the higher rating for the agent could be explained by the use of social capabilities of the agent, we further investigated the reason for the higher ratings for the teammates using demographics and other available ratings as additional factors in the model. First of all, we found a significant gender effect on teammate rating. Male participants were significantly (F(1,49)=4.9, $p < 0.05$) more likely to rate their teammates higher (Average=6.11) compared to female participants (Average=5.55). We found no significant effect of the gender distribution of the teammates on the participant's ratings of their teammates. Second, we observed a significant positive correlation (r=0.566, $p < 0.001$) between the teammate rating and the cooperation rating given by the participant (Q15 on Appendix G). This suggests that teammates were rated higher when the participants perceived a higher degree of cooperation during the group decision making activity.

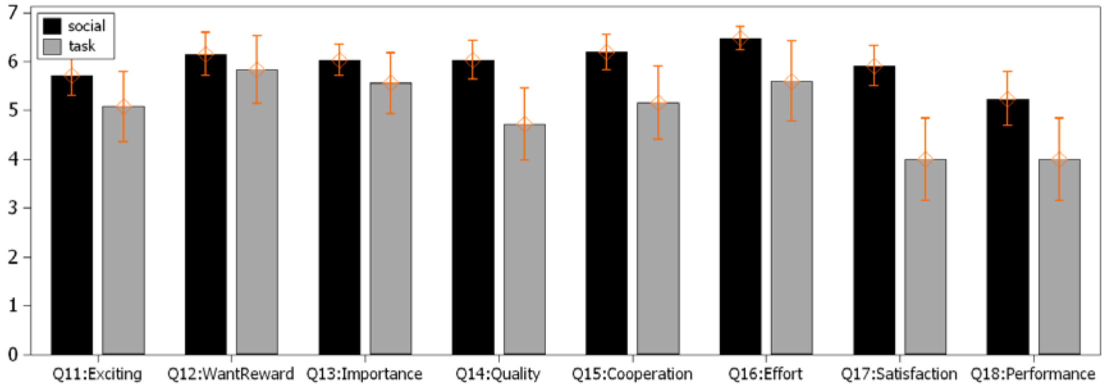


Figure 7.5: Average Ratings about Team, Task and Discussion

Figure 7.5 shows a plot of the average rating for the other items (Q11-Q18) on the survey shown in Appendix G. We observe significant effects in favor of the Social condition for the items 14 – 18 (Quality of discussion, Cooperation, Best Effort, Task Satisfaction, Team Performance).

Overall, we observe strong effects in favor of the Social condition for both the performance as well as the perception metrics. The use of socially capable agents can help in improving the quality of a team’s decisions and enrich their experience during group decision making activities.

Chapter 8

Conclusion

Interfaces such as Conversational Agents can offer effective support to users in a variety of interactive situations. The work presented in this thesis is motivated by a vision of agents that can participate and engage users in multi-party interactive situations while enhancing their productivity and enriching their experience.

Through this thesis, we have explored two problems that must be addressed to achieve this vision. The first of these problems involves the technical issues surrounding the implementation of agents that can participate in such situations. These problems arise from the assumptions (Chapter 2) implicit within the approaches and tools previously available for the development of Conversational Agents.

The second problem focuses on the communication skills that agents participating in multi-party situations must display. Specifically, we find the need for creating socially capable agents. In this work, we follow a model of human social behavior developed by research in small group communication to identify and implement the social capabilities that are relevant to the interactive situations investigated here.

This thesis contributes approaches, knowledge and software artifacts that address both of these problems. Experiments described in this thesis discuss the effectiveness and appropriate use of some of these contributions in the context of two multi-party application domains: Collaborative Learning and Group Decision Making.

In this chapter, we will list the specific contributions of this thesis in the context of these two problems. This will be followed by a discussion of the shortcomings and future directions leading from the conclusions derived from our experiments.

8.1 Thesis Contributions

8.1.1 Building Agents for Multi-Party Interactive Situations

Chapter 2 and Chapter 3 of this thesis describe our work on identifying and alleviating the problems involved in building agents for MPIS. We describe an approach for modeling conversation as an orchestration of triggering of behaviors. Development of agents based on this approach is supported by the Basilica architecture which is the primary technical contribution of this thesis. Basilica is an event-driven architecture that represents CAs as a network of behavioral components.

Following a programmatic approach (vs. an authoring approach), the behaviors of each of these components are specified using a high-level programming language (like Java). This approach provides us with a rich representational capability to build agents because the behaviors of these agents are not restricted to simple combinations of a limited set of conversational operations. This is different from many existing formalisms for representing agent behaviors which use very high level languages that provide only a small set of operators to specify agent behavior.

The use of an event-driven architecture and decomposition of the agent into a network of behavioral components provides the flexibility to model complex interaction dynamics that are often observed during multi-party interactive situations (e.g. Multi-Party turn taking in a collaborative learning situation). The loose-coupling of behavioral components that follows from the decomposition facilitates re-usability of these components across agents as demonstrated by the progression of agents discussed in Chapter 3.

As an extension of the benefits of this representational capability and decomposability, the architecture facilitates the development of hybrid agents that combine many existing techniques for modeling the individual conversational behaviors encapsulated within each of the behavioral components. Hence, in a way, the Basilica architecture functions as a meta-architecture.

Besides building the Basilica architecture, we have developed a collection of agents using this architecture for a variety of interactive tasks and a variety of agent roles to demonstrate the breadth of capabilities of the Basilica architecture. These agent implementations provide reusable components that can be used for developing other agents. Also, the designs of the component networks of these agents provide guidelines and examples for designing agents that can be implemented using the Basilica architecture.

8.1.2 Socially Capable Conversational Agents

This thesis experimentally explored the design space of socially capable agents. The relevant social capabilities of these agents are determined within the context of their application using the social-emotional interaction categories identified by research in Small Group Communication. Chapter 4 and Chapter 7 discuss the use of this approach to design and implement agents that support users in two applications scenarios, i.e., collaborative learning and group decision making.

The thesis describes a series of experiments conducted using these socially capable agents. The results and analysis of these experiments contribute the following insights into the use of socially capable conversational agents.

Agents with social capabilities can achieve significantly better task success and perception ratings compared to agents that perform the same task related behaviors but no social behaviors. Specifically, experiments 1 and 4 show this in the context of the two different applications listed above. While both the applications use similar metrics of agent perception, the metrics used for task success are very different. However, a similar effect of the use of social behavior is observed. Together, these results recommend the use agents with social capabilities to support users in multi-party interactive situations.

Structural equation modeling discussed in Section 5.4 contributes an insight into the mechanism through which the use of social behaviors by agents achieve the outcomes described above. We find that social behavior helps in regulating the attention of the user towards the information being delivered by the agent. Specifically, we notice that social behavior helps in counteracting the negative effects of dysfunction in groups on the attention of the users. The resulting higher attention of the user towards the agent has a direct positive effect on task success. While on the one hand this recommends the use of socially capable agents in applications frequently used by communicatively dysfunctional groups, we also note that the benefits of the use of these social capabilities may diminish or disappear in the case of interaction involving high functioning groups.

Experiment 2 and 3 investigate the issue of appropriate use of the social capabilities of the agent used in experiment 1. Experiment 2 compares three agents that perform different amount of social behavior to determine an appropriate amount of social behavior. Experiment 3 on the other hand compares six agents that use three different policies for timing social behavior.

First of all, both of these experiments emphasize the important of performing the right amount of social behavior. A simpler recommendation based on experiment 2 suggests using a moderate (15-20%) amount of social behavior on average over the course of a 35 minute interaction with a group of users. The fine grained social ratio filtering approach discussed in Section 6.1.8 suggests the use of model that capture that temporal dynamics of the appropriate amount of social behavior over the course of the 35 minutes interaction with the users. The model learnt using the filtering approach suggests that it may be appropriate to perform as much as 50% of social behavior in the formative/conclusive phases of group interaction and agrees with the 15% recommendation of the simpler model during the performance phase of the interaction.

Second, based on experiment 3, we find support for the use of a triggering policy learnt from data that captures a series of human-made decisions to perform social behavior in a collaborative learning application. Compared to random and rule-based triggering policies, the learnt policy used in combination with the social ratio filter mentioned above was found to be the only policy that performs highly on both the task success as well as the perception metrics. Besides recommending the choice of an appropriate triggering policy, this thesis also contributes an approach for modeling the timing of human social behavior. Specifically, this approach uses a large margin learning technique to learn a model that decides when a social behavior should be performed. The technique provides us the flexibility to design optimization constraints that direct the algorithm towards discovering a model that optimizes a metric (such a P_k or kKappa) of our choice.

Together, these experiments contributes knowledge that helps us in taking a step towards achieving the vision of creating effective and useable agents that can use social capabilities to improve the support they offer to users in multi-party interactive situations. In addition to this knowledge, the experiments described here and other experiments conducted using the various agents built using the Basilica architecture contribute data collected from over 1000 human subjects who make up over 300 small groups.

Above all, through this work, we have attempted to bridge the fields of Small group communication and Conversational Agents. As further discussed in the next section, both of these fields stand to gain from this bridge. While group communication offers insights into the design of appropriate communication skills for CAs, these agents offer a controlled technique for simulating and modeling the effects of specific behaviors on the outcomes of group interaction.

8.2 Directions

Before we discuss the next steps and promising future directions resulting from this thesis, four shortcomings of the analysis and techniques used in this work are listed below.

First, in our analysis of the effect of the use of social behaviors on learning, we only consider the social behavior performed by the agents. While this is the primary interest of this thesis, conventionally an analysis of Balesian equilibrium between instrumental and expressive processes in group interaction takes into account the expressive contributions of all the participants in the group. In our analysis, we have assumed invariance in the amount of social behavior performed by the users between the teams used in our analysis. As a next step towards improving our understanding of the mechanism through which social behaviors affect outcomes of our interest, further analysis should incorporate variables that measure the social behaviors performed by the users.

Second, the techniques used to learn a policy for triggering social behavior only helps the agents decide if they should perform a social behavior. As discussed in section 6.1.5, this decision is followed by a rule-based determination of which social behavior should be performed. Although rarely, this leads to the situations such as the triggering policy recommending that the agent should perform a social behavior while the rules do not find any of the social interaction strategies applicable at that time. In our current implementation, this is addressed by performing a default social behavior. This hybrid approach that combines learnt and rule based models to create a triggering policy is a suitable solution in our case considering the relatively small amount of data available to train the triggering policy. However, this directs our attention to a general problem of learning multi-class policies that have to choose between more than two options. As we suggest an approach that model agents as a collection of behavior one or more which may be triggered by learnt policies, it may be useful to investigate algorithms that can learn to choose between more than two actions associated with a behavior.

Related to the above mentioned shortcoming is the issue of coordination among various behaviors that may be triggered almost simultaneously. Our current implementations of CAs address this issue in different ways. For example, the *WrenchTalker* tutor agent described in section 4.5 achieves coordination between the task and social behaviors of the agent by using a control token that is alternately shared between the components that perform these behaviors. The component that does not need control, immediately relinquishes it to the other component. In contrast to this implementation, the NEO agent described in section 7.2 uses a *ControlManager* component to achieve coordination between the three behaviors of that agent. While currently the control manager uses a simple queue based approach to coordinate between the three behaviors, we find this to be a better approach as we move towards agents with large number of behaviors. Depending on the application and the design of the agent's component network, a more sophisticated coordination policy such as a priority queue can be implemented as an extension to this approach. In general, coordination among behaviors is a one of the challenges that must be investigated further to create highly complex CAs.

The fourth shortcoming relates to the two options offered by this thesis to help us choose the appropriate amount of social behavior that agents should perform. Both of these choices (fixed percentage, social ratio filter) use an interaction invariant model. However, it may be useful to change the amount of social behavior performed by the agents based on the characteristics of the group and its interaction. The structural equation model shown in Figure 5.3 offers two potential approaches to adaptively determine the right amount of social behavior for collaborative learning applications. First, a simple approach can use the magnitude of increase in the episode duration from an efficient minimum for each tutoring episode. Higher episode duration can be used as an indication of the need for higher amount of social behavior. A more complex approach could model the dysfunction in the groups by automatically monitoring the bad behavior of the students and use that to determine the right amount of social behavior appropriate for each group.

Besides the four shortcoming and the corresponding next steps discussed above, we have considered two promising research directions that could use the techniques and knowledge developed in this thesis.

First of all, we found that social behavior acts as a regulatory mechanism in group interaction. Specifically, it regulates user attention towards the agent measured as the reduction in the time it takes for the tutor to deliver its instructional content during tutoring episodes. Discovery and study of other regulatory mechanism which are crucial to functioning of small groups could help us create better agents for MPIS.

Second, the conversational agents are only one of many types of interfaces that support group interaction. Other interfaces such as chatrooms, multi-player video games, discussion forums, online social networks, etc. could potentially benefit from having relevant social capabilities alongside their existing task related functionality. In general, the display of multiple behaviors so as to maintain equilibrium between different objectives of the interactive situation is a problem that re-occurs in many of these modern interfaces. A common example of this is the display of advertisements on websites where designs and developers must choose the right amount of advertising for the content being presented to maintain equilibrium between the attention and the financial economies of the website.

8.2.1 Outlook

In contrast to the useful yet rather simplistic applications which were being investigated in the context of conversational agents and dialog systems until recently, we now see this technology being applied to richer and more complex interactive applications such as the two applications discussed in this thesis. Future work on CAs will focus on developing systems that can interact with many users using multiple modalities for input and output. The duration of an interactive session may extend from minutes to hours and days. The agents may need to interact with other interfaces including other agents. The range of application may vary from experience driven casual use to performance driven serious use. Many of these aspects are currently being investigated by researchers within the context of several application scenarios.

As we move towards realizing these agents, research on this technology must investigate not only novel approaches for representing and generating interactive capabilities, but also allow several approaches to work together to create highly interactive agents. While our use of Basilica as a meta-architecture enables this within the scope of the work discussed in this work, earlier in this section, we have discussed two shortcomings of behavior generation and coordination that must be investigated in the near future as we move towards these complex applications. Along these lines, few other directions are worth further investigation.

We must investigate approaches to generate novel behavior to keep the user engaged over long term interaction. The use of data-driven and crowd sourcing techniques can help us acquire agent behavior from the observation of human behavior. In addition to approaches for modeling agents the focus on adhering to a plan or reacting to local discourse events, we must develop approaches that allow agents to be opportunistic. Such agents will be able to effectively utilize opportunities to achieve secondary goals such as delivering recommendations to the user, eliciting user feedback, shaping user perspective towards the task or the agent, etc. The use of non-anthropomorphic interactive behaviors (e.g. beeps) can help in improving the communication efficiency in certain application. The lack of a model of the effects of such behaviors impedes their use and is worth future exploration.

Within the context of multi-party interactive situations, the issue of group formation is crucial. While in our applications ad-hoc groups were used, group functioning and productivity can be further improved by selectively grouping users that are likely to complement yet cooperate with each other. Initial or prior interaction of the users with agents can be used to systematically or opportunistically elicit user characteristics that can help in determining optimal grouping of users.

Appendix A

Test administered during Wrench Lab

Q1. Stress is **1 point**

a)	Force	b)	Force x Area
c)	Force / Area	d)	Area / Force

Q2. Does stress determine how easy it is to use a Wrench to turn a bolt? 1 point

a) Yes
b) No

Q3. Explain your answer to Q2 in the space below **2 points**

Q4. In general, while designing a Wrench, we want to **1 point**

a) Increase Stress
b) Decrease Stress
c) Not change Stress

Q5. Explain your answer to Q4 in the space below **2 points**

For questions 6 - 8, What happens to the following on increasing the length of the handle?

Q6. Stress	a)	Increases	**1 point**
	b)	Decreases	
	c)	Doesn't Change	
Q7. Ease of Use	a)	Improves	**1 point**
	b)	Degrades	
	c)	Doesn't Change	
Q8. Cost	a)	Increases	**1 point**
	b)	Decreases	
	c)	Doesn't Change	

Q9. Explain your answer to Q7 in the space below **2 points**

____

____

Q10. If the Yield Stress of the following material is as below **1 point**

Plastic = 4000 lb/in^2
Aluminum = 35000 lb/in^2
Steel = 50000 lb/in^2
Titanium = 80000 lb/in^2

From a safety point of view, which is the best material of choice for building the Wrench?

a) Plastic b) Aluminum
c) Steel d) Titanium

Q11. The maximum stress in the Aluminum Wrench you designed is about 5000 lb/in^2. If you double the length of the handle, which all of the following materials can you use safely to manufacture the wrench. Use a Safety Factor of 4.

2 points

a) Plastic b) Aluminum
c) Steel d) Titanium

Appendix B

Test administered during Thermodynamics Lab

For each of the following changes to cycle parameters, how is the efficiency changed. Pick only one option. (1 point each)

Q1. Decreasing Maximum Temperature at which heat is added to the Cycle

(a) increases (b) decreases (c) remains the same

Q2. Decreasing Minimum Temperature at which heat is rejected from the Cycle

(a) increases (b) decreases (c) remains the same

Q3. Why is excessive moisture or liquid water in steam undesirable in steam turbines? (1 point)

Q4. What limits the minimum pressure at which a condenser of a Rankine cycle can be operated at? (2 points)

Q5. If you were cared both about the power output and the environmental impact of a Rankine cycle, how would you achieve the best compromise between these two goals? (1 point)

Consider a simple ideal Rankine cycle with fixed turbine inlet temperature (Tmax) and condenser pressure (Pmin). For each of the following (Q6 – Q11), what is the effect of increasing the boiler pressure (Pmax) on: (1 point each)

Q6. Pump work input

(a) increases (b) decreases (c) remains the same

Q7. Turbine work output

(a) increases (b) decreases (c) remains the same

Q8. Heat supplied

(a) increases (b) decreases (c) remains the same

Q9. Heat rejected

(a) increases (b) decreases (c) remains the same

Q10. Cycle efficiency

(a) increases (b) decreases (c) remains the same

Q11. Liquid moisture content at turbine exit

(a) increases (b) decreases (c) remains the same

Q12. Explain your answer to Question 10 (2 points)

Consider a simple ideal Rankine cycle with fixed boiler and condenser pressures (Pmax and Pmin). For each of the following (Q13 – Q18) what is the effect of superheating the steam to a higher temperature on: (1 point each)

Q13. Pump work input

(a) increases (b) decreases (c) remains the same

Q14. Turbine work output

(a) increases (b) decreases (c) remains the same

Q15. Heat supplied
(a) increases (b) decreases (c) remains the same

Q16. Heat rejected
(a) increases (b) decreases (c) remains the same

Q17. Cycle efficiency
(a) increases (b) decreases (c) remains the same

Q18. Liquid moisture content at turbine exit
(a) increases (b) decreases (c) remains the same

Q19. Explain your answer to Question 16 (2 points)

Consider a simple ideal Rankine cycle with fixed turbine inlet conditions (Pmax and Tmax). For each of the following (Q20 – Q25) what is the effect of lowering the condenser pressure (Pmin) on: (1 point each)

Q20. Pump work input
(a) increases (b) decreases (c) remains the same

Q21. Turbine work output
(a) increases (b) decreases (c) remains the same

Q22. Heat supplied
(a) increases (b) decreases (c) remains the same

Q23. Heat rejected
(a) increases (b) decreases (c) remains the same

Q24. Cycle efficiency
(a) increases (b) decreases (c) remains the same

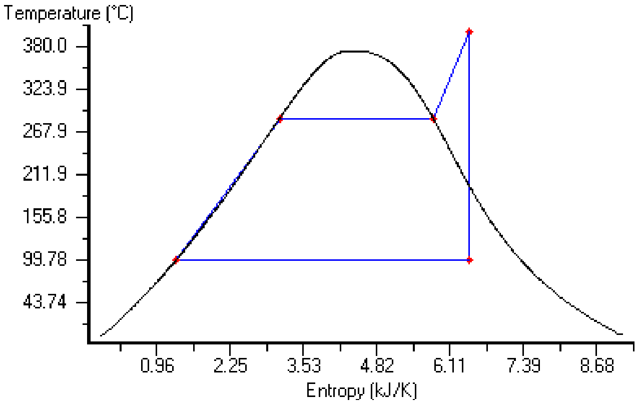
Q25. Liquid moisture content at turbine exit
(a) increases (b) decreases (c) remains the same

Q26. Explain your answer to Question 21 (2 points)

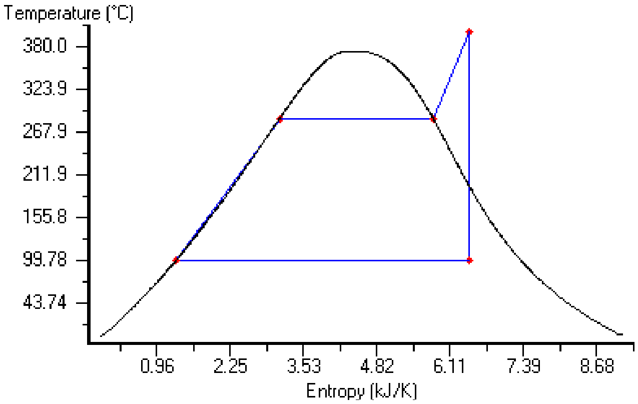
Q27. Which of the following energy sources is the best choice for improving Cycle efficiency? (1 point)

(a) Coal (b) Solar (c) Natural Gas (d) Nuclear

Q28. In the T-S diagram for a Rankine Cycle shown below, indicate the net amount of work done by the Rankine Cycle. (1 point)



Q29. Modify the T-S diagram for a Rankine Cycle shown below to indicate how the cycle changes when you increase the Maximum Pressure of the Cycle (2 points)



Appendix C

Collaborative Learning Perception Survey

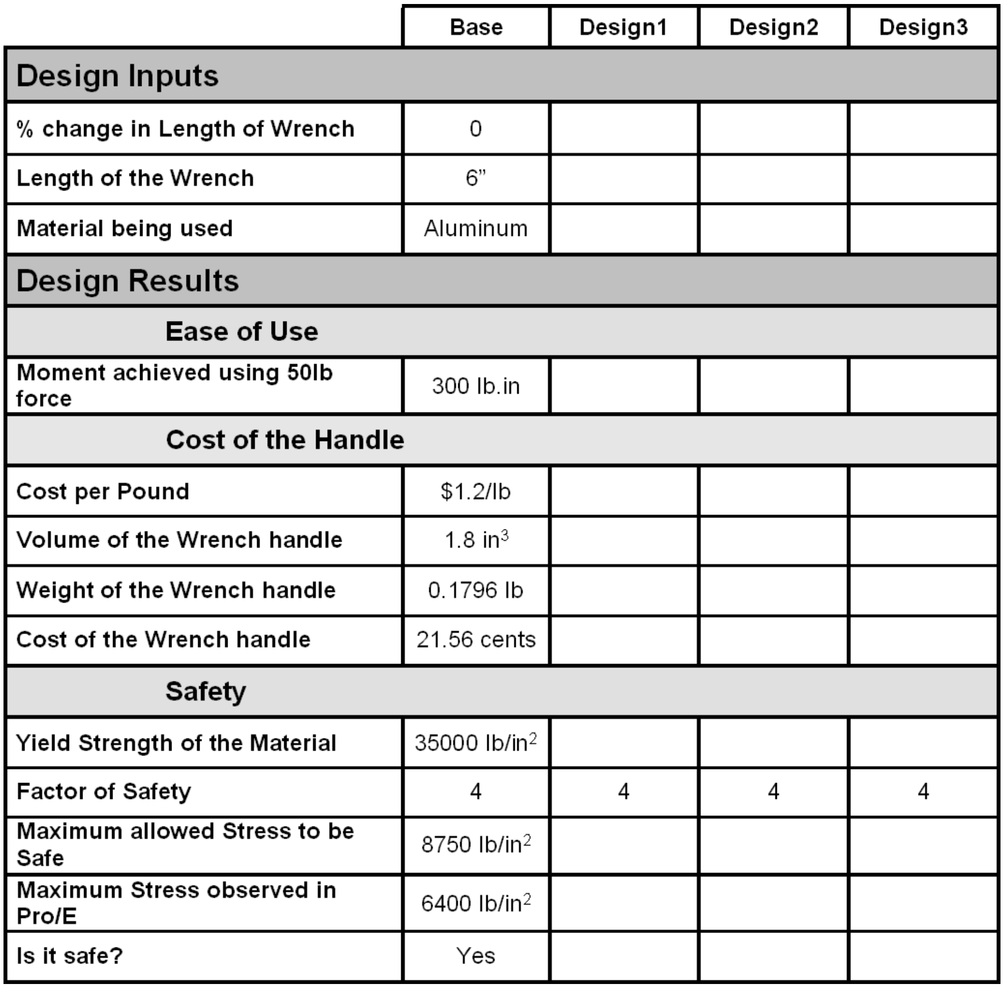
Using the following scale, Indicate to what extent you agree with each of the following items.

1	**2**	**3**	**4**	**5**	**6**	**7**
Strongly Disagree	Mostly Disagree	Somewhat Disagree	Neutral	Somewhat Agree	Mostly Agree	Strongly Agree

Q1	I liked the tutor very much.	1	2	3	4	5	6	7
Q2	The tutor was very cordial and friendly during the discussion	1	2	3	4	5	6	7
Q3	The tutor was providing very good ideas for the discussion	1	2	3	4	5	6	7
Q4	The tutor kept the discussion at a very comfortable level socially	1	2	3	4	5	6	7
Q5	The tutor was part of my team	1	2	3	4	5	6	7
Q6	The tutor received the ideas and suggestions I contributed to the discussion positively	1	2	3	4	5	6	7
Q7	I am happy with the discussion we had during the design challenge	1	2	3	4	5	6	7
Q8	My group was successful at meeting the goals of the design challenge	1	2	3	4	5	6	7
Q9	The design challenge was exciting and I did my best to come up with good designs	1	2	3	4	5	6	7

Appendix D

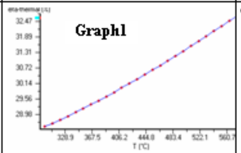
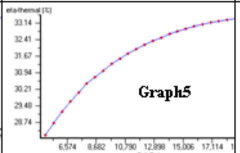
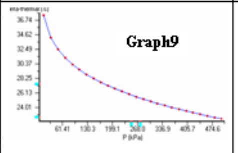
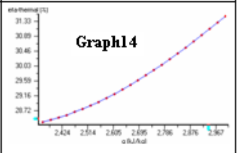
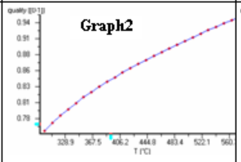
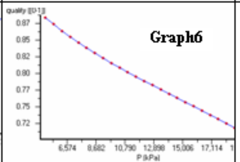
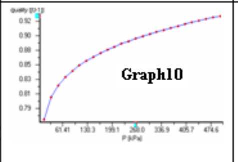
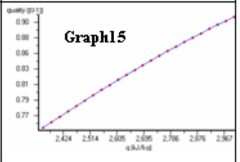
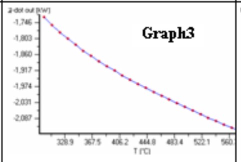
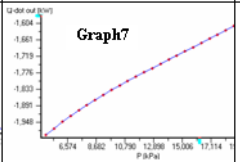
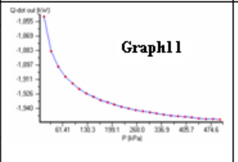
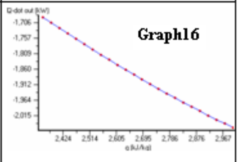
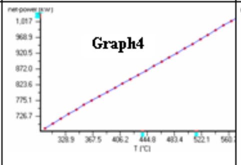
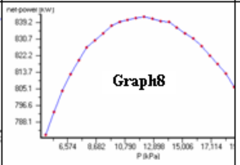
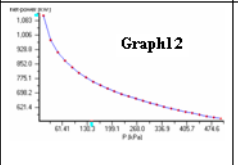
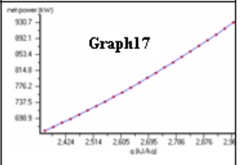
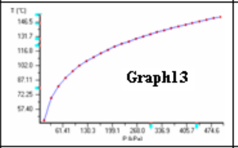
Design Sheet for Collaborative Wrench Design Activity



	Base	Design1	Design2	Design3
Design Inputs				
% change in Length of Wrench	0			
Length of the Wrench	6”			
Material being used	Aluminum			
Design Results				
Ease of Use				
Moment achieved using 50lb force	300 lb.in			
Cost of the Handle				
Cost per Pound	$1.2/lb			
Volume of the Wrench handle	1.8 in^3			
Weight of the Wrench handle	0.1796 lb			
Cost of the Wrench handle	21.56 cents			
Safety				
Yield Strength of the Material	35000 lb/in^2			
Factor of Safety	4	4	4	4
Maximum allowed Stress to be Safe	8750 lb/in^2			
Maximum Stress observed in Pro/E	6400 lb/in^2			
Is it safe?	Yes			

Appendix E

Design sheet for Collaborative Power Plant Design Activity

	Maximum Temperature Must be < 570°C	**Maximum Pressure** Must be < 20000kPa	**Minimum Pressure**	**Choice of Fuel**
Efficiency				
Steam Quality Cannot be less than 0.85				
Waste Heat				
Net Power Output You want this as high as possible				
Minimum Temperature Must be > 40°C				
Notes & Values you choose				

Appendix F

Scoring Rubric for Non-Combatant Evacuation Planning

Planning Card	Error	Error Type	Points Lost
Personnel	Omitting Personnel Card	A	20
Transportation	Omitting Transportation Card	A	20
	Using aircraft that requires in flight refueling without calculating refueling needs	C	5
	Calculation error rendering the solution impossible	C	5
	Using aircraft that requires in flight refueling (with correct calculations)	E	2
	Performing calculations incorrectly with minimal impact	E	2
	Using one aircraft that needs refueling, but others that don't.	E	2
Weapons	Omitting Weapons Card	A	20
	Using unavailable weapons	D	3
	Failing to address weapons used by selected military	D	3
Times	Omitting Times Card	A	20
	Failing to include required critical times (onset of operation, contact with workers, evacuation, return)	C	5
	Calculation error rendering the solution impossible	C	5
	Failing to account for tides or coral reef (if using sea approach)	D	3

Planning Card	Error	Error Type	Points Lost
	Rescuing workers during daylight hours	D	3
	Performing calculations incorrectly with minimal impact	E	2
	More than 0.5 hours spent in church	E	2
Plan	Omitting Plan Card	A	20
	Unrealistic solution (tanks, etc.)	B	10
	Neglecting to include all requirements of the plan as listed in the Mission Statement • Getting to the church • Evacuating the workers • Returning to the base or ship	C	5 for each
	Failing to address medical treatment a. Insulin, b. Broken leg	C	2.5 each
	Harming the enemy unnecessarily	C	5
	Damaging the village unnecessarily	C	5
	Failing to arrange for translator if need established	D	3
	Failing to address detection	D	3
	Failing to avoid detection if addressed	C	2
	Failing to avoid land mines	E	2
Miscellaneous	Other Type A error (e.g. omitting planning card)	A	20
	Other Type B error (e.g. serious violation of mission statement, unrealistic solution)	B	10
	Other Type C error (e.g. moderate violation of mission statement, calculation error with serious impact)	C	5
	Other Type D error (e.g. minor violation of mission statement)	D	3
	Other Type E error (e.g. calculation error with minimal impact)	E	2

Appendix G

Group Decision Making Perception Survey

1 Strongly Disagree	2 Mostly Disagree	3 Somewhat Disagree	4 Neutral	5 Somewhat Agree	6 Mostly Agree	7 Strongly Agree

Using the above scale,
Indicate to what extent you agree with each of the following statements for each participant in your team. Ignore the field about yourself.

Q1	This participant provided good ideas for our task.	**Admin**	1	2	3	4	5	6	7
		Weapons	1	2	3	4	5	6	7
		Environmental	1	2	3	4	5	6	7
		Intelligence	1	2	3	4	5	6	7

Q2	This participant received my contributions positively.	**Admin**	1	2	3	4	5	6	7
		Weapons	1	2	3	4	5	6	7
		Environmental	1	2	3	4	5	6	7
		Intelligence	1	2	3	4	5	6	7

Q3	This participant was friendly during the discussion.	**Admin**	1	2	3	4	5	6	7
		Weapons	1	2	3	4	5	6	7
		Environmental	1	2	3	4	5	6	7
		Intelligence	1	2	3	4	5	6	7

Q4	This participant responded to my contributions.	**Admin**	1	2	3	4	5	6	7
		Weapons	1	2	3	4	5	6	7
		Environmental	1	2	3	4	5	6	7
		Intelligence	1	2	3	4	5	6	7

Q5	This participant helped in lowering the tension in our team.	**Admin**	1	2	3	4	5	6	7
		Weapons	1	2	3	4	5	6	7
		Environmental	1	2	3	4	5	6	7
		Intelligence	1	2	3	4	5	6	7

Q6	This participant was paying attention to our conversation.	**Admin**	1	2	3	4	5	6	7
		Weapons	1	2	3	4	5	6	7
		Environmental	1	2	3	4	5	6	7
		Intelligence	1	2	3	4	5	6	7

Q7	Overall, I liked this participant very much.	**Admin**	1	2	3	4	5	6	7
		Weapons	1	2	3	4	5	6	7
		Environmental	1	2	3	4	5	6	7
		Intelligence	1	2	3	4	5	6	7

Q8	I often ignored what this participant was saying.	**Admin**	1	2	3	4	5	6	7
		Weapons	1	2	3	4	5	6	7
		Environmental	1	2	3	4	5	6	7
		Intelligence	1	2	3	4	5	6	7

Q9	This participant's contributions got in the way of our planning.	**Admin**	1	2	3	4	5	6	7
		Weapons	1	2	3	4	5	6	7
		Environmental	1	2	3	4	5	6	7
		Intelligence	1	2	3	4	5	6	7

Q10	This participant was an important part of my team.	**Admin**	1	2	3	4	5	6	7
		Weapons	1	2	3	4	5	6	7
		Environmental	1	2	3	4	5	6	7
		Intelligence	1	2	3	4	5	6	7

Using the same scale as above,
Indicate to what extent you agree with each of the following statements.

Q11	The mission was exciting	1	2	3	4	5	6	7
Q12	I wanted to win the extra reward	1	2	3	4	5	6	7
Q13	My group members felt that the discussion was important	1	2	3	4	5	6	7
Q14	I am happy with the discussion we had during this task	1	2	3	4	5	6	7

Q15	We worked together during the mission	1	2	3	4	5	6	7
Q16	We tried our best to come with the rescue plan	1	2	3	4	5	6	7
Q17	My team was successful at meeting all the objectives of our mission	1	2	3	4	5	6	7
Q18	The plan we came up with will qualify for the extra reward	1	2	3	4	5	6	7

Appendix H

Knowledge Test:
Non-Combatant Evacuation

Consider the following evacuation plan that one of the teams came up with:

Personnel:
Flight Crew for two helicopters
4 for seahawk, 2 for hornets
2 riflemen with rifles and grenades
1 riflemen with just rifle
1 radioman

Weapons:
Whatever is on the helicopters

Transportation:
1 US Navy SeaHawk
1 F-18 Hornet
1 C-130 with extra fuel

Comments:
Hornets for fire support

Timing:

1. Seahawk leaves USS Enterprise: 3am
2. Hornet leaves USS Enterprise: 4:45am
3. C-130 leaves base in time to refuel seahawk
4. Seahawk Arrive at Church: 5am
5. Hornet arrives at church 5mins after seahawk
6. Leave Island: 5:30am
7. Return to USS Enterprise: 9am

Details:

- Hover over church and airlift evacuees
- Two navy seals will reach down to church and secure area, stabilize medical situation and evacuate
- Fly over the north and enter the island from north
- Seahawk to be refuel on route back to USS Enterprise by C-130

Answer the following questions about the above plan:

Q1. **Do the rescuers have all the personnel they need to execute this plan?** YES NO

Q2. **Do the rescuers have sufficient weapons to execute this plan?** YES NO

Q3. **Do the rescuers have sufficient transportation to execute this plan?** YES NO

Q4. **Does the plan address all the medical care that must be provided to the evacuees?** YES NO

Q5. **According to this plan, are the rescuers likely to be detected or engage the enemy?** YES NO

Q6. **Does the plan involve potentially complicated operations like refueling, use of tanks, navigating through land mines, involving locals, etc.** YES NO

Q7. **On a scale of 1 (worst) to 10 (best), how would rate the above plan?** __________

Consider the following evacuation plan that one of the teams came up with:

Personnel:
7-man squad of Navy seals
6-man squad of Army special forces

Weapons:
As included on aircraft + /w team

Transportation:
3 Toyota Trucks
1 C-130
1 Zodiac
1 Blackhawk
5 F-18s

Timing:
- Onset: 2am
- Contact: 4:10am
- Evacuation: 4:20am
- Return to Base or Ship: 5:20am

Details:
- 2am: Contact local by radio for 3 toyota trucks to be waiting 50 miles south of church on the shore
- Fly C-130 in with 5 seals, 6 army, 6 miles from shore
- Meet trucks on shore + at the same time send in 3 F-18s on the east shore for diversion
- Drive to church in trucks
- 2 medics go into church w/water + insulin
- 4:20 black hawks will pick team up while F-18s defends us.
- Send the F-18s back

Answer the following questions about the above plan:

Q1. **Do the rescuers have all the personnel they need to execute this plan?** YES NO

Q2. **Do the rescuers have sufficient weapons to execute this plan?** YES NO

Q3. **Do the rescuers have sufficient transportation to execute this plan?** YES NO

Q4. **Does the plan address all the medical care that must be provided to the evacuees?** YES NO

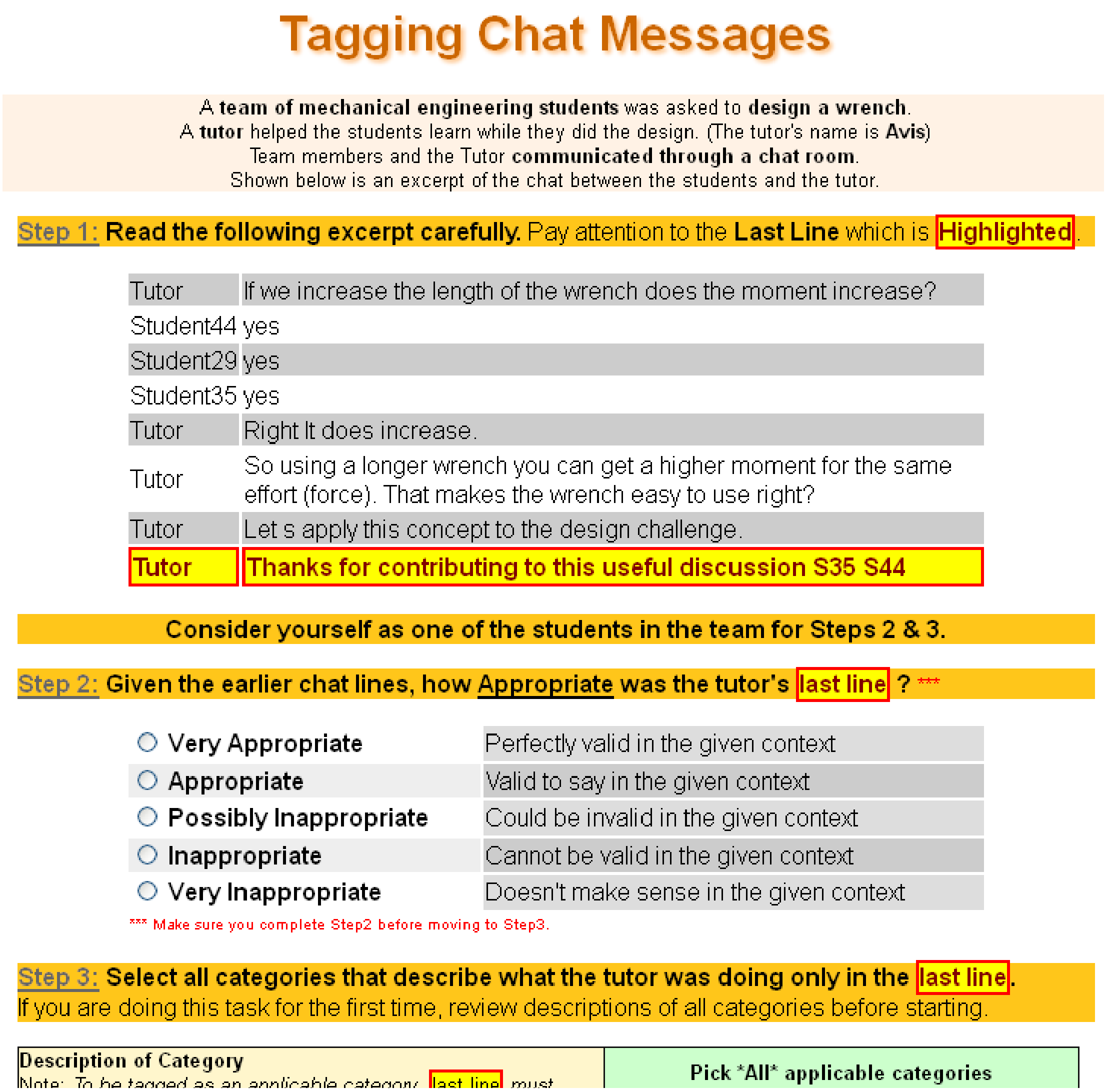
Q5. **According to this plan, are the rescuers likely to be detected or engage the enemy?** YES NO

Q6. **Does the plan involve potentially complicated operations like refueling, use of tanks, navigating through land mines, involving locals, etc.** YES NO

Q7. **On a scale of 1 (worst) to 10 (best), how would rate the above plan?** _________

Appendix I

Design of the Annotation Interface



Tagging Chat Messages

A **team of mechanical engineering students** was asked to **design a wrench**.
A **tutor** helped the students learn while they did the design. (The tutor's name is **Avis**)
Team members and the Tutor **communicated through a chat room**.
Shown below is an excerpt of the chat between the students and the tutor.

Step 1: **Read the following excerpt carefully.** Pay attention to the **Last Line** which is **Highlighted**.

Tutor	If we increase the length of the wrench does the moment increase?
Student44	yes
Student29	yes
Student35	yes
Tutor	Right It does increase.
Tutor	So using a longer wrench you can get a higher moment for the same effort (force). That makes the wrench easy to use right?
Tutor	Let s apply this concept to the design challenge.
Tutor	**Thanks for contributing to this useful discussion S35 S44**

Consider yourself as one of the students in the team for Steps 2 & 3.

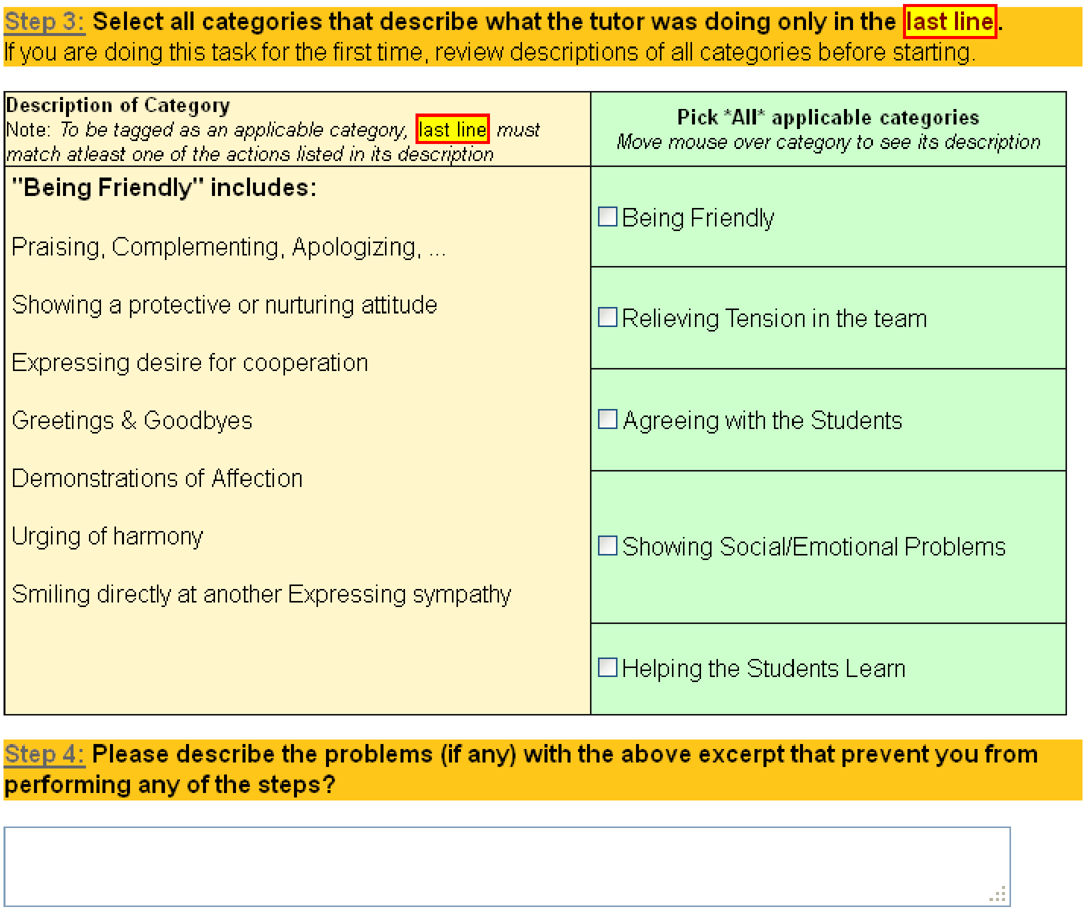
Step 2: **Given the earlier chat lines, how Appropriate was the tutor's last line ?** ***

○ **Very Appropriate**	Perfectly valid in the given context
○ **Appropriate**	Valid to say in the given context
○ **Possibly Inappropriate**	Could be invalid in the given context
○ **Inappropriate**	Cannot be valid in the given context
○ **Very Inappropriate**	Doesn't make sense in the given context

*** Make sure you complete Step2 before moving to Step3.

Step 3: **Select all categories that describe what the tutor was doing only in the last line.**
If you are doing this task for the first time, review descriptions of all categories before starting.

Description of Category *Note: To be tagged as an applicable category,* last line *must*	**Pick *All* applicable categories**



Step 3: **Select all categories that describe what the tutor was doing only in the last line.**
If you are doing this task for the first time, review descriptions of all categories before starting.

Description of Category Note: *To be tagged as an applicable category,* last line *must match atleast one of the actions listed in its description*	**Pick *All* applicable categories** *Move mouse over category to see its description*
"Being Friendly" includes: Praising, Complementing, Apologizing, ... Showing a protective or nurturing attitude Expressing desire for cooperation Greetings & Goodbyes Demonstrations of Affection Urging of harmony Smiling directly at another Expressing sympathy	☐ Being Friendly ☐ Relieving Tension in the team ☐ Agreeing with the Students ☐ Showing Social/Emotional Problems ☐ Helping the Students Learn

Step 4: **Please describe the problems (if any) with the above excerpt that prevent you from performing any of the steps?**

Appendix J

Rules for Triggering Social Behaviors

Only Triggered if Current Social Ratio < Threshold (0.2 in Experiment 1)		
Event Received	→	**Social Behavior Triggered**
DORMANT_GROUP When group is inactive		1e: Encourage Prompt targeted towards entire group
DORMANT_STUDENT When individual is inactive		1e: Encourage Prompt targeted inactive individual

Plan Step Completed		**Social Behavior Triggered**
DO_GREETINGS	→	1a: Do introductions
DO_CONCLUSION		1f: Conclude socially
{ Calculation Steps }		2a: Expression of feeling better Prompt is selected based on step completed
{ Design Review Steps }		2c: Express enthusiasm, elation, satisfaction Prompt is selected based on step completed

Plan Step Completed	→	**Social Behavior Triggered**
{ Tutoring Episode Steps }		1d: Compliment / Praise Praised students selected based on contributions

Only Triggered if Current Social Ratio < Threshold (0.2 in Experiment 1)		
Student Message Annotation	→	**Social Behavior Triggered**
TEASING && !TUTOR_REFERENCE		1b: Be protective & nurturing
DISCONTENT || HELP_NEEDED		1c: Give reassurance
SMILES || POSITIVITY		2b: Be cheerful
IDEA_CONTRIBUTION		3a: Show attention
GIVING_OPINION || GIVING_ORIENTATION		3b: Show compression / approval

Bibliography

(Ai et. al., 2010) Hua Ai, Rohit Kumar, Amrut Nagasunder, Carolyn P. Rosé, 2010, *Exploring the Effectiveness of Social Capabilities and Goal Alignment in Computer Supported Collaborative Learning*, Intelligent Tutoring Systems, Pittsburgh, PA

(Aist and Mostow, 2009) Gregory Aist and Jack Mostow, 2009, *Designing Spoken Tutorial Dialogue with Children to Elicit Predictable but Educationally Valuable Responses*, Interspeech 2009, Brighton, UK

(Aleven et. al., 2001) Vincent Aleven, Octav Popescu and Kenneth R. Koedinger, 2001, *Pedagogical Content Knowledge in a Tutorial Dialogue System to Support Self-Explanation*, AIED-2001 Workshop on Tutorial Dialogue Systems, San Antonio, Texas

(Allen et. al. 1996) James F. Allen, Bradford W. Miller, Eric K. Ringger, and Teresa Sikorski, *A Robust System for Natural Spoken Dialogue*, Proc. of the 1996 Annual Meeting of the Association for Computational Linguistics (ACL'96), June 1996. pp. 62-70

Amtrack Julie, 2003, http://www.networkworld.com/news/ 2003/0619julie.html

(Andre et. al., 2004) Elisabeth Andre, Matthias Rehm, Wolfgang Minker, and Dirk Buhler, 2004, *Endowing Spoken Language Dialogue Systems with Emotional Intelligence*, Proc. of Affective Dialogue Systems, 2004

(Arnott et. al., 2008) Elizabeth Arnott, Peter Hastings and David Allbritton, 2008, *Research Methods Tutor: Evaluation of a dialogue-based tutoring system in the classroom*, Behavior Research Methods, 40 (3), 694-698

A.L.I.C.E. Bot, http://www.alicebot.org/about.html

(Bales, 1950) Robert F. Bales, 1950, *Interaction process analysis: A method for the study of small groups*, Addison-Wesley, Cambridge, MA

(Bales, 1953) Robert F. Bales, 1953, *The Equilibrium Problem in Small Groups*, In: T. Pardone, R. F. Bales, E. A. Shils (Eds): Working Papers in the Theory of Action, Glencoe, IL, Free Press

(Bales, 1958) Robert F. Bales, 1958, *Task roles and social roles in problem-solving groups,* In: T. M. Newcomb and E. L. Hartley (Eds): Readings in Social Psychology, Hold, Rinhart and Winston, 1958

(Banerjee and Rudnicky, 2006) Satanjeev Banerjee and Alex Rudnicky, 2006, *SmartNotes: Implicit Labeling of Meeting Data through User Note-Taking and Browsing*, Proc. of the NAACL-HLT, New York, NY

Basilica Wiki, Architecture for building Conversational Agents, http://basilica.rohitkumar.net/wiki/

(Beeferman et. al., 1999) Doug Beeferman, Adam Berger and John D, Lafferty, 1999, *Statistical Models for Text Segmentation*, Machine Learning, 34 (1-3): 177-210

(Benuš, 2009) Štefan Benuš, 2009, *Are we 'in sync': Turn-taking in collaborative dialogues*, Proc. Interspeech 2009

(Bhatt et. al., 2004) Khelan Bhatt, Martha Evens, Shlomo Argamon, 2004, *Hedged responses and expressions of affect in human/human and human/computer tutorial interactions*, CogSci, Chicago, IL

(Bickmore and Cassell, 2001) Timothy Bickmore and Justine Cassell, 2001, *Relational agents: a model and implementation of building user trust*, Proc. of the SIGCHI Conference on Human Factors in Computing Systems, CHI '01. ACM, New York, NY, 396-403

(Bickmore et. al., 2009) Timothy Bickmore, Daniel Schulman and Langxuan Yin, 2009, *Engagement vs. Deceit: Virtual Humans with Human Autobiographies*, Proc. of Intelligent Virtual Agents, Amsterdam, Netherlands

(Bion, 1961) Wilfred R. Bion, 1961, *Experiences in groups: And other papers*. Basic Books, New York, NY

(Bohus, 2007) Dan Bohus, 2007, *Error Awareness and Recovery in Conversational Spoken Language Interfaces*, PhD dissertation, Computer Sci-ence Department, Carnegie Mellon University, Pittsburgh, PA

(Bohus et. al., 2007a) Dan Bohus, Sergio Grau, David Huggins-Daines, Venkatesh Keri, Gopala Krishna A., Rohit Kumar, Antoine Raux, and Stefanie Tomko, 2007, *Conquest - an Open-Source Dialog System for Conferences*, HLT-NAACL 2007, Rochester, NY

(Bohus et. al., 2007b) Dan Bohus, Antoine Raux, Thomas Harris, Maxine Eskenazi and Alex Rudnicky, 2007, *Olympus: an open-source framework for conversational* *spoken language interface research*, HLT-NAACL 2007, Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technology, Rochester, NY

(Bohus and Horovitz, 2009) Dan Bohus and Eric Horvitz, 2009, *Dialog in the Open World: Platform and Applications*, Proc. of the 11th International Conference on Multimodal Interfaces and the 6th Workshop on Machine Learning for Multimodal Interfaces, Cambridge, MA, USA' , pp. 31-38

(Bohus and Rudnicky, 2003) Dan Bohus and Alex Rudnicky, 2003, *RavenClaw: Dialog Management Using Hierarchical Task Decomposition and an Expectation Agenda*, Eurospeech 2003, Geneva, Switzerland

(Bos and Oka, 2003) Johan Bos and Tetsushi Oka, 2003, *Building Spoken Dialogue Systems for Believable Characters*, 7th workshop on the semantics & pragmatics of dialogue

(Burke, 1967) Peter J. Burke, 1967, *The development of Task and Social-Emotional Role Differentiation*, Sociometry, vol. 30 (4), pp. 379-392

(Callaway et. al., 2007) Charles Callaway, Myroslava Dzikovska, Elaine Farrow, Manuel Marques-pita, Colin Matheson and Johanna Moore, 2007, *The Beetle and BeeDiff tutoring systems*, SLaTE 2007, Farmington PA

(Clark, 2005) Herbert H. Clark, 2005, *Coordinating with each other in a material world*, Discourse Studies, 7 (4-5), 507-525

(Constantinides et. al., 1998) Paul C. Constantinides, Scott Hansma, Chris Tchou, Alexander I. Rudnicky, 1998, *A schema-based approach to dialog control*, Proc. of ICSL

(Cassell et.al., 1999) Justine Cassell, Timothy Bickmore, M. Billinghurst, L. Campbell, K. Chang, H. Vilhjálmsson, and H. Yan, 1999, *Embodiment in Conversational Interfaces: REA*, Proc. of the CHI'99 Conference, pp. 520-527. Pittsburgh, PA

(Chaudhuri et. al., 2008) Sourish Chaudhuri, Rohit Kumar, Carolyn P. Rosé, 2008, *It's not easy being green - Supporting Collaborative Green Design Learning*, ITS 2008, Montreal

(Chaudhuri et. al., 2009) Sourish Chaudhuri, Rohit Kumar, Iris Howley, Carolyn P. Rosé, 2009, *Engaging Collaborative Learners with Helping Agents*, Proc. of AI in Education

(Pearce and Conger, 2003) Craig L. Pearce and Jay Alden Conger (Eds.), 2003, *Shared leadership: Reframing the hows and whys of leadership,* Sage, Thousand Oaks, CA

(Crammer and Singer, 2003) Koby Crammer and Yoram Singer, 2003, *Ultraconservative online algorithms for multiclass problems,* Journal of Machine Learning Research, 3, 951-991

Dan Bohus, *A list of spoken language interfaces,* http://research.microsoft.com/en-us/um/people/dbohus/SDS

(DeVault et. al., 2009) David DeVault, Kenji Sagae and David Traum, 2009, *Can I finish? Learning when to respond to incremental interpretation results in interactive dialogue*, SIGDIAL 2009, London, UK

(Davis, 1969), James H. Davis, 1969, *Group performance*, Addison–Wesley, Reading, MA

(D'Mello et. al., 2008) Sidney D'Mello, Tanner Jackson, Scotty Craig, Brent Morgan, Patrick Chipman, Holly White, Natalie Person, Barry Kort, Rana el Kaliouby, Rosalid W. Picard and Arthur Graesser, 2008, *AutoTutor Detects and Responds to Learners Affective and Cognitive States*, Workshop on Emotional and Cognitive Issues, ITS 2008, Montreal

(Dohsaka et. al., 2009) Kohji Dohsaka, Ryoto Asai, Ryichiro Higashinaka, Yasuhiro Minami and Eisaku Maeda, 2009, Effects of Conversational Agents on Human Communication in Though Evoking Multi-Party dialogues, 10th Annual SigDial meeting on Discourse and Dialogue, London, UK

(Dybala et. al., 2009) Pawel Dybala, Michal Ptaszynski, Rafal Rzepka and Kenji Araki, *Humoroids: Conversational Agents that induce positive emotions with humor*, 2009, AAMAS, Budapest, Hungary

(Edlund et. al., 2004) Jens Edlund, Gabriel Skantze and Rolf Carlson, 2004, *Higgins - a spoken dialogue system for investigating error handling techniques*, Proc. of ICSLP, Jeju Island, Korea

(Eisenstein and Barzilay, 2008) Jacob Eisenstein and Regina Barzilay, 2008, *Bayesian unsupervised topic segmentation*, Proc. of EMNLP, Honolulu, HI

(Ferguson et. al., 2009) George Ferguson, James Allen, Lucian Galescu, Jill Quinn, Mary Swift, 2009, *CARDIAC: An Intelligent Conversational Assistant for Chronic Heart Failure Patient Heath Monitoring*, Proc. of the AAAI Fall Symposium on Virtual Healthcare Interaction, Arlington, VA

(Foner, 1997) Leonard N. Foner, 1997, *Entertaining agents: a sociological case study*, Proc. of the First international Conference on Autonomous Agents, AGENTS '97. ACM, New York, NY, 122-129

(Forbus et. al. 1999) Kenneth D. Forbus, Peter B. Whalley, John O. Everett, Leo Ureel, Mike Brokowski, Julie Baher, Sven E. Kuehne, 1999, *CyclePad: An Articulate Virtual Laboratory for Engineering Thermodynamics*, Artificial Intelligence, vol. 114 (1-2), pp. 297-347

(Freedman, 2000) Reva Freedman, 2000, *Plan-based dialogue management in a physics tutor*, Proc. 6th conference on Applied Natural Language, pp 52-59

(Gulz et. al., 2010) Agneta Gulz, Annika Silvervarg and Björn Sjödén, 2010, *Design for off-task interaction - Rethinking pedagogy in technology enhanced learning*, Intl. Conf. on Advanced Learning Technologies, Tunisia

(Graesser et. al., 2001) Arthur C. Graesser, Natalie K. Person. Derek Harter and The Tutoring Research Group, 2001, *Teaching tactics and dialog in AutoTutor,* Intl. journal of AI in Education. vol. 12(3), pp. 257-279

(Graesser et. al., 2003) Arthur C. Graesser, Kristen Moreno, Johana Marineau, Amy Adcock, Andrew Olney and Natalie Person, 2003, *AutoTutor Improves Deep Learning of Computer Literacy: Is it the Dialog or the Talking Head?* AIED 2003, Sydney, Australia

(Graesser et. al., 2005) Arthur C. Graesser, Patrick Chipman, Brian C. Haynes, and Andrew Olney, 2005, *AutoTutor: An Intelligent Tutoring System with Mixed-initiative Dialogue*, IEEE Transactions in Education, 48, 612-618

(Gockley et. al., 2005) Rachel Gockley, Allison Bruce, Jodi Forlizzi, Marek Michalowski, Anne Mundell, Stephanie Rosenthal, Brennan Sellner, Reid Simmons, Kevin Snipes, Alan C. Schultz and Jue Wang, 2005, *Designing Robots for Long-Term Social Interaction,* IROS 2005

(Gorin et. al., 1997) Allen L. Gorin, Giuseppe Riccardi, Jeremy H. Wright, 1997, *How may I help you?* Speech Communication (SPEECH) 23(1-2):113-127

(Gweon et. al., 2011) Gahgene Gweon, S. Jeon, J. Lee, and Carolyn P. Rosé, 2011, *A Framework for Assessment of Student Project Groups On-Line and Off-Line*, In S. Puntambekar, G. Erkens, C. Hmelo-Silver, C. (Eds.), Analyzing Collaborative Interactions in CSCL: Methods, Approaches and Issues, Springer

(Hall et. al., 2009) Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann and Ian H. Witten, 2009, T*he WEKA Data Mining Software: An Update*; SIGKDD Explorations, Volume 11, Issue 1

(Hanna et. al., 2007) Philip Hanna, Ian O'neill, Craig Wootton, Michael Mctear, 2007, *Promoting extension and reuse in a spoken dialog manager: An evaluation of the queen's communicator*, ACM Trans. Speech Language Processing 4, 3

(Harris and Rudnicky, 2007) Thomas K. Harris and Alex I. Rudnicky, 2007, *Teamtalk: a platform for multi-human-robot dialog research in coherent real and virtual spaces*, Association for the Advancement of Artificial Intelligence, Vancouver B.C., Canada

(Isbell et. al., 2001) Charles Isbell, Christian R. Shelton, Michael Kearns, Satinder Singh, Peter Stone, 2001, *A social reinforcement learning agent*, Proc. of the 5th International Conference on Autonomous Agents, AGENTS '01, Montreal, Canada

(Isbister et. al., 2000) Katherine Isbister, Hideyuki Nakanishi, Toru Ishida, Cliff Nass, 2000, *Helper Agent: Designing an assistant for human-human interaction in a virtual meeting space*, CHI 2000, 57-64, The Hague, Netherlands

(Isbister and Doyle, 2002) Katherine Isbister and Patrick Doyle, 2002, *Design and Evaluation of Embodied Conversational Agents: A Proposed Taxonomy*, AAMAS Workshop: Embodied Conversational Agents, Bologna, Italy

(Miller, 1976) Jean Baker Miller, 1976, *Toward a New Psychology of Women.* Beacon Press, Boston, MA

(Johnson, 2007) W. Lewis Johnson, 2007, *Serious use of a serious game for language learning*, In R. Luckin et al. (Eds.), Artificial Intelligence in Education, Amsterdam

(Jordan et. al., 2006) Pamela W. Jordan, Maxim Makatchev, Umarani Pappuswamy, Kurt VanLehn and Patricia Albacete, 2006, *A natural language tutorial dialogue system for physics*, Proc of the 19th International FLAIRS Conference. Menlo Park, CA

(Jordan et. al., 2007) Pamela Jordan, Brian Hall, Michael Ringenberg, Yue Cui, Carolyn P. Rosé, 2007, *Tools for Authoring a Dialogue Agent that Participates in Learning Studies*, AIED 2007

(Kumar et. al., 2011) Rohit Kumar, Jack L. Beuth and Carolyn P. Rosé, 2011, *Conversational Strategies that Support Idea Generation Productivity in Groups*, 9th Intl. Conf. on Computer Supported Collaborative Learning, Hong Kong

(Kumar and Rosé, 2010a) Rohit Kumar, Carolyn P. Rosé, 2010, *Conversational Tutors with Rich Interactive Behaviors that support Collaborative Learning*, Workshop on Opportunities for intelligent and adaptive behavior in collaborative learning systems, ITS 2010, Pittsburgh, PA

(Kumar et. al., 2010) Rohit Kumar, Hua Ai, Jack Beuth, Carolyn P. Rosé, 2010, *Socially-capable Conversational Tutors can be Effective in Collaborative-Learning situations*, Intelligent Tutoring Systems, Pittsburgh, PA

(Kumar and Rosé, 2010b) Rohit Kumar, Carolyn P. Rosé, 2010, *Engaging learning groups using Social Interaction Strategies*, NAACL-HLT, Los Angeles, CA

(Kumar et. al., 2007a) Rohit Kumar, Carolyn Rosé, Mahesh Joshi, Yi-Chia Wang, Yue Cui and Allen Robinson, 2007, *Tutorial Dialogue as Adaptive Collaborative Learning Support*, 13th AIED 2007, Los Angeles, California

(Kumar et. al., 2007b) Rohit Kumar, Gahgene Gweon, Mahesh Joshi, Yue Cui, Carolyn Rosé, *Supporting students working together on Math with Social Dialogue*, Workshop on Speech and Language Technology in Education, Farmington, PA, 2007

(Kumar and Rosé, 2009) Rohit Kumar, Carolyn Rosé, 2009, *Building Conversational Agents with Basilica*, Proc. of NAACL-HLT, Boulder, CO

(Kun et. al., 2007) Andrew Kun, Tim Paek and Zeljko Medenica, 2007, *The Effect of Speech Interface Accuracy on Driving Performance*, Interspeech 2007, Antwerp, Belgium

(Lane and VanLehn, 2004) H. Chad Lane and Kurt VanLehn, 2004, *A Dialogue-based tutoring system for beginning programming*, FLAIRS 2004, Miami Beach, FL

(Laughlin, 1980) Pattrick R. Laughlin, 1980, *Social combination processes of cooperative problem-solving groups on verbal intellective tasks*, In: M Fishbein (Ed.), Progress in Social Psychology (pp. 127–155). Hillsdale, NJ: Erlbaum

(Letsky et. al., 2008) Michael P. Letsky, Norman W. Warner, Stephen M. Fiore, and C. A. P. Smith, (Eds.)., 2008, *Macrocognition in teams: Theories and methodologies*, Ashgate

(Litman et. al., 2000) Diane Litman, Satinder Singh, Michael Kearns and Marilyn Walker, 2000, *NJFun: a reinforcement learning spoken dialogue system*, Proc. of the ANLP-NAACL 2000 Workshop on Conversational Systems, Morristown, NJ

(Litman and Silliman, 2004) Diane J. Litman and Scott Silliman, 2004, *ITSPOKE: An Intelligent Tutoring Spoken Dialogue System*, HLT-NAACL Demonstrations, Boston, MA

(Lison, 2011) Pierre Lison, 2011, *Multi-Policy Dialogue Management*, SigDial 2011, Portland OR

(Liu and Chee, 2004) Yi Liu and Yam San Chee, *Designing Interaction Models in a Multiparty 3D learning environment*, Intl. Conf. on Computers in Education, 2004

(McDonald et. al., 2005) Ryan McDonald, Koby Crammer and Fernando Pereira, 2005, *Online large-margin training of dependency parsers*, Proc. of ACL, 91-98, Ann Arbor, MI

(McGrath, 1984) Joseph E. McGrath, 1984, *Groups: Interaction and Performance*, Prentice-Hall, NJ

(Mühlpfordt and Wessner, 2005), Martin Mühlpfordt and Martin Wessner, 2005, *Explicit referencing in chat supports collaborative learning*, Proc. Computer Support for Collaborative Learning (CSCL)

(Murray et. al., 2001) R. Charles Murray, Kurt VanLehn and Jack Mostow, 2001, *A decision-theoretic architecture for selecting tutorial discourse actions*, AIED-2001 Workshop on Tutorial Dialogue Systems, San Antonio, Texas

(Nakano et. al., 2008) Mikio Nakano, Kotaro Funakoshi, Yuji Hasegawa, Hiroshi Tsujino, 2008, *A Framework for Building Conversational Agents Based on a Multi-Expert Model*, 9th SigDial Workshop on Discourse and Dialog, Columbus, Ohio

(Nallasamy et. al., 2008) Udhyakumar Nallasamy, Alan W Black, Tanja Schultz and Robert Frederking, 2008, *NineOneOne: Recognizing and Classifying Speech for Handling Minority Language Emergency Calls*, LREC 2008, Marrakech, Morocco

(Neikrasz and Moore, 2010) John Niekrasz and Johanna D. Moore, 2010, *Unbiased Discourse Segmentation Evaluation*, Proc. of SLT, Berkeley, CA

NoHold Instant Support, http://www.nohold.com

Nuance Café, http://cafe.bevocal.com/

(O'Neill et. al., 2003) Ian O'Neill, Philip Hanna, Xingkun Liu and Michael McTear, 2003, *The Queen's Communicator: A Object-Oriented Dialogue Manager*, Proc. Eurospeech 2003, pp 593-596

(Pakucs and Melin, 2001) Botond Pakucs and Håkan Melin, 2001, *PER: A speech based automated entrance receptionist*, In 13th Nordic Conference of Computational Linguistics, NoDaLiDa'01. Uppsala University, Uppsala

(Patel et. al., 2003) Niraj Patel, Michael Glass and Jung Hee Kim, 2003, *Data Collection Applications for the NC A&T State University Algebra Tutoring*, Fourteenth Midwest Artificial Intelligence and Cognitive Science Conference, Cincinnati, OH

(Pevzner and Hearst, 2002) Lev Pevzner and Marti A. Hearst, 2002, *A critique and improvement of an evaluation metric for text segmentation*, Computational Linguistics, 28(1):19–36

(Raux et. al. 2005) Antoine Raux, Brian Langner, Dan Bohus, Alan Black, and Maxine Eskenazi, 2005, *Let's Go Public! Taking a Spoken Dialog System to the Real World*, Interspeech 2005, Lisbon, Portugal

(Raux and Eskenazi, 2007) Antoine Raux and Maxine Eskenazi, 2007, *A Multi-Layer Architecture for Semi-Synchronous Event-Driven Dialogue Management*, ASRU 2007, Kyoto

(Raux, 2008) Antoine Raux, 2008, *Flexible turn-taking in Spoken Dialog Systems*, PhD Dissertation, Language Technologies Institute, Carnegie Mellon University, Pittsburgh, PA

(Rayner et. al., 2005) Manny Rayner, Beth Ann Hockey, Jean-Michel Renders, Nikos Chatzichrisafis and Kim Farrell, 2005, *Spoken Language Processing in the Clarissa Procedure Browser*, Natural Language Engineering 1 (1), pp. 1-28

(Rosé et. al., 2001a) Carolyn P. Rosé, Pamela Jordan, Michael Ringenberg, Stephanie Siler, Kurth VanLehn, Anders Weinstein, 2001, *Interactive Conceptual Tutoring in Atlas-Andes*, Proc. Artificial Intelligence in Education (AIED), San Antonio, TX

(Rosé et. al., 2001b) Carolyn P. Rosé, Johanna D. Moore, Kurt VanLehn, David Allbritton, 2001, *A Comparative Evaluation of Socratic versus Didactic Tutoring*, Proc of Cognitive Sciences Society

(Rosé et. al., 2003) Carolyn P. Rosé, D. Bhembe, S. Siler, R. Srivastava, and Kurt VanLehn, 2003, *Exploring the Effectiveness of Knowledge Construction Dialogues*, Proc. Artificial Intelligence in Education (AIED)

(Rosé et. al., 2006) Carolyn P. Rosé, Rohit Kumar, Vincent Aleven, Allen Robinson, Chih Wu, 2006, *CycleTalk: Data Driven Design of Support for Simulation Based*

Learning, Intl. Journal of AI in Education Special Issue on The Best of ITS '04 , 16, 195-223

(Rudnicky and Xu, 1999) Alex I. Rudnicky, and Wei Xu, 1999, *An agenda-based dialog management architecture for spoken language systems*, IEEE Automatic Speech Recognition and Understanding Workshop

(Scheines et. al., 1994) Richard Scheines, Peter Spirtes, Clark Glymour, and Christopher Meek, 1994, *TETRAD II: Tools for Discovery*, Lawrence Erlbaum Associates, Hillsdale, NJ

(Seneff et. al., 1998) Stephanie Seneff, Ed Hurley, Raymond Lau, Christine Pao, Philipp Schmid, and Victor Zue, 1998, GALAXY-II: A Reference Architecture for Conversational System Development, Proc. ICSLP 98, Sydney, Australia

Siri: Virtual Personal Assistant, http://siri.com/

(Skantze and Schlangen, 2009) Gabriel Skantze and David Schlangen, 2009, *Incremental dialogue processing in a Micro-Domain*, EACL. Athens, Greece

Staffan Larsson, *Dialogue systems and projects*, http://www.ling.gu.se/~sl/dialogue_links.html

(Stone, 1966) Phillip J. Stone, 1966, *The General Inquirer: A Computer Approach to Content Analysis*. The MIT Press

(Takeuchi et. al., 2004) Masashi Takeuchi, Norihide Kitaoka and Seiichi NakagawaM, 2004, *Timing detection for realtime dialog systems using prosodic and linguistic information*, Intl Conf. on Speech Prosody 2004, Nara, Japan

Tetrad IV, http://www.phil.cmu.edu/projects/tetrad/tetrad4.html

(Thelen, 1956) Herbert A. Thelen, 1956, *Emotionality and Work in Groups*, In L. D. White (ed.), The State of the Social Sciences, Chicago, IL: University of Chicago Press

(Traum and Rickel, 2002) David Traum, Jeff Rickel, 2002, *Embodied agents for multi-party dialogue in immersive virtual worlds*, AAMAS, Bologna, Italy

(Turunen et. al., 2004) Markku Turunen, Esa-Pekka Salonen, Mikko Hartikainen, Jakko Hakulinen, William Black, Allan Ramsay, Adam Funk, Andrew Conroy, Paul Thompson, Mark Stairmand, Kristiina Jokinen, Jyrki Rissanen, Kari Kanto, Antti Kerminen, Bjorn Gamback, Maria Cheadle, Fredrik Olsson and Magnus Sahlgren, 2004, *AthosMail - a Multilingual Adaptive Spoken Dialogue System for*

E-mail Domain, Workshop on Robust and Adaptive Information Processing for Mobile Speech Interfaces

(Turunen and Hakulinen, 2003) Markku Turunen and Jaakko Hakulinen, 2003, *Jaspis - An Architecture for Supporting Distributed Spoken Dialogues*, Eurospeech' 2003, Geneva, Switzerland

(VanLehn et. al., 2007) Kurt VanLehn, Arthur C. Graesser, G. Tanner Jackson, Pamela Jordan, Andrew Olney, Carolyn P. Rosé, 2007, *When Are Tutorial Dialogues More Effective Than Reading?* , Cognitive Science 31, pp. 3–62

(Veletsianos et. al., 2009) George Veletsianos, Charles Miller and Aaron Doering, Veletsianos, 2009, *EnALI: A Research and Design Framework for Virtual Characters and Pedagogical Agents,* Journal of Educational Computing Research, 41(2), 171-194

VoiceXML, http://www.w3.org/TR/voicexml21/, 2007

Voxeo Prophecy, http://www.voxeo.com/products/

(Wang and Johnson, 2008) Ning Wang and Lewis Johnson, 2008, *The Politeness Effect in an intelligent foreign language tutoring system*, Intelligent Tutoring Systems, Montreal, Canada

(Warner et. al., 2003) Norman W. Warner, Elizabeth M. Wroblewski and K. Shuck, 2003, *Noncombatant Evacuation Operation Scenario*, Naval Air Systems Command, Human Systems Department (4.6), Patuxent River, MD.

(Weizenbaum, 1966) Joseph Weizenbaum, 1966, *ELIZA — A Computer Program For the Study of Natural Language Communication Between Man And Machine*, Communications of the ACM 9 (1): 36–45

(Weusijana et. al., 2008) Baba Kofi A. Weusijana, Rohit Kumar, Carolyn P. Rosé, 2008. *MultiTalker: Building Conversational Agents in Second Life using Basilica*, Second Life Education Community Convention, Purple Strand: Educational Tools and Products, 2008, Tampa, FL

(Williams, 2007) Jason D. Williams, 2007, *Applying POMDPs to Dialog Systems in the Troubleshooting Domain*, Proc. HLT/NAACL Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technology, Rochester, NY

(Zheng et. al., 2005) Jun Zheng, Xiang Yuan, Yam San Chee, 2005, *Designing multiparty interaction support in Elva, an embodied tour guide*, AAMAS, Netherlands

http://thesis.rohitkumar.net/thesis.pdf