2 EXPLORING THE ROLE OF SOCIAL LEARNING ON TEAM MENTAL MODELS

Vishal Singh[*,a], **Andy Dong**[*,b] **and John S Gero**[†]

[*]*Design Lab, University of Sydney, Sydney, Australia. +61 2 9351 3031; +61 2 9351 2769.*
E-mails: [a]*vsin9057@mail.usyd.edu.au;* [b]*adong@arch.usyd.edu.au*
[†]*Krasnow Institute for Advanced Study, Virginia, USA and UTS, Sydney, Australia.*
E-mail: john@johngero.com

This paper presents a computational model developed to study the role of different modes of social learning on the formation of team mental models (TMM). In this model, the TMM is formed over time as team members learn about each other. Members can learn about other team members from (a) their personal interactions with the task and other team members (b) observing other members perform tasks and interact with a third member. The different modes of learning are discretely represented, allowing controlled analysis of the role of each of the learning modes in the formation of the TMM. The research framework, experiment design and simulation results are presented. Outcomes of this research have implications for the way design teams are put together, how they are managed, and how they can be studied using computational simulations.

Keywords: Team-Mental-Model, Team Expertise, Social Learning.

1. INTRODUCTION

Designing is increasingly a team activity. Knowledge about the tasks to be performed and the expertise is distributed across the team composed of individual experts. Cross[1] and LaFrance[2] suggest that experts possess both factual knowledge as well as tactical knowledge. A team is more than the sum of its parts and a mere collection of individual experts is not enough to produce an expert team.[3] While variability exists in the definition of team expertise, there is a general agreement that team expertise means: (1) team as a whole has the relevant domain knowledge; (2) team members have well-developed strategies for task performance and task handling, and (3) team members have well-developed knowledge of each others capabilities. This means the team members should have well-developed mental models. Team members form different mental models based on task, context, process, team membership and competence[4] to perform the task collectively. Studies have been conducted on the influence of mental models on team performance.[5,6] The research reported here is focussed on the simulation of the development of expertise based on a computational model of social learning as the basis for forming team mental models (TMM). The results have implications for the way design teams are put together and how they are managed.

A TMM is considered as an individual agent's knowledge of its own competence and the competence of all other agents in the team to perform the different tasks. If an agent cannot perform a specific task, a well-developed TMM allows it to allocate the task to the agent that is most competent in performing the given task. The importance of knowing the knowledge source in a distributed system has also been emphasized in research on transactive memory systems.[7]

Social learning is important in the formation of TMMs, influencing the team performance. Though separate studies have been conducted to investigate the relationships between TMMs and team performance, and different modes of social learning have been reported,[8,9] the contribution of the different modes of social learning on the development of TMMs and team expertise needs to be established. The

method taken is this research is a comparative analysis of the roles of each of these learning modes on the formation of a TMM.

This paper describes a computational framework and the experiment design to study the effects of different modes of social learning on the formation of a TMM, and, hence, team expertise. Computational models are recognized as a useful tool for studying and advancing theories on social and organizational behaviour.[10,11]

Simulations used to validate the framework are presented. Empirical data from experiments are presented to demonstrate the usefulness of the model in developing a better understanding of the role of social learning on design team expertise.

2. RESEARCH FRAMEWORK

Figure 1 is a schematic representation of the research framework. Social interaction in a team depends on the team structure (scope of interaction) and prior acquaintance (pre-existing mental models). Interaction among agents allows them to learn about each other. How much an agent learns about other agents depends on the agent's learning capabilities. In this model two dimensions of agents' learning capabilities are considered: (1) level of detail of the mental model; and (2) cognitive abilities that determine the modes of learning (what kinds of data can an agent sense from the environment). In this framework team structure and agent capabilities are parameters that influence the rate of formation of mental models of other agents.

The TMM is formed as agents form Agent-Mental-Models (AMM) of individual agents. Formation of TMM determines how the team uses its expertise to perform the task. Performance of a team is measured as the rate of task completion. Teams that have a higher rate of formation of team expertise should have a higher rate of task completion.

The effectiveness of the formation of TMM is measured by comparing the developed TMM against the actual TMM (as given in the form of expertise distribution). For each agent the TMM formed could be different based on their individual experience, because where an agent is (e.g. which sub-group), when and whom it interacts with (or observes), and what the interaction (or observation) is about, matters.[12] Hence, the overall TMM formed is calculated as the average of the TMMs formed by each of the team members.

2.1. Experiment Design

A matrix of experiments is proposed. Parameters varying along the two axes are: (1) agent parameters; and (2) team parameters.

Agent parameters are defined by an individual agent's learning capabilities. In a team environment, four kinds of social learning are identified: (a) learning from personal interaction with the task and other agents; (b) learning from explicit information exchange with other agents; (c) learning through observation of the tasks being performed by other agents; and (d) learning by observing interaction among other agents.

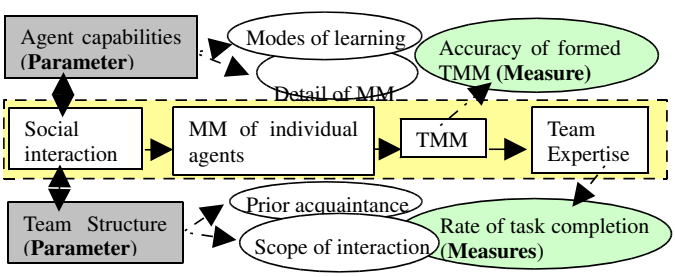

Figure 1. Schematic representation of the research framework.

Team parameters include: (a) prior acquaintance amongst team members; (b) team structure; and (c) the rate of attrition and acquisition. Prior acquaintance is implemented as pre-existing mental models of known agents. Team structure is implemented by constraining interactions that affect social learning amongst agents. The rate of attrition is implemented as the probability of removal or addition of a random agent from/into the team during the simulation. Combinations of these team parameters provide different team environments along the team axis.

Since this research is restricted to the development of a TMM, the domain knowledge, task-mental-model and task handling strategies, process-mental-model have been pre-coded into the agent's knowledge base and remain fixed.

3. COMPUTATIONAL MODEL

3.1. Modeling Decisions and Implementation

The computational model has been implemented in JADE (Java Agent Development Environment)[13] based on the following.

3.1.1. Agent Communication

Agent interaction in the team is in the form of message passing. Agents exchange messages based on FIPA (Foundation for Intelligent Physical Agents)[14] protocols, which are pre-agreed message exchange protocols for agent communication language messages to facilitate interoperability and standardization. Agents exchange messages to: (a) allocate tasks; (b) inform source agent that task is done; and (c) send refusal messages to convey their inability to perform the task.

3.1.2. Agent's Knowledge Base

An agent's domain knowledge is fixed, i.e. task mental model for agents is well developed. For a specific input and required task agents either know the solution or do not have any knowledge of the solution. In case the solution is not known, agents refuse the task. Agents know all the design steps i.e. if an agent can perform a task then it knows the next design step to be performed. The protocol for the task handling is also known to all the agents i.e. their process mental model is well-developed. But when the team is initially formed, apart from the knowledge of their own capabilities, agents do not know capabilities of other agents or "who knows what?" i.e. the TMM is not developed. Once the simulation is started agents develop the TMM through different modes of learning.

Implementation: when an agent is initialized it has a default AMM for all the agents in the agent population. This AMM consists of: (a) role identifier; (b) counters for N^P, the number of times the agent has performed the given task and N^A, the number of times a task has been allocated to the agent.

The ratio N^P/N^A gives an agent's competence value for a given role. For each agent there are as many such competence values as the total number of roles. When the agent is initialized, the default value for N^P is 1, and N^A is 2 because, with no prior information there is an equal chance that an agent may or may not have competence in the given role.

The AMM is represented as an M-dimensional vector of competence values of the M roles within the team. The TMM is represented as an M x N matrix $\mathbf{E} = \{e_{ij}\}$, where N is the total number of agents. Each element e_{ij} represents the competence c of the jth agent for the ith role, such that $0<c<1$.

Updating AMM and TMM: when an agent receives a positive feedback on an agent's competence both N^P and N^A are incremented by one. When a negative feedback is received then only the N^A is incremented by one.

3.1.3. Agent Learning

Agents learn as they interact with their environment that includes the task and the other agents. This interaction with the environment includes the observations that the agents make based on the sense data available to them. Agent's learning is limited to the TMM, which is primarily learning "who knows what?" Agents can learn from their personal interaction with other agents and through observations,

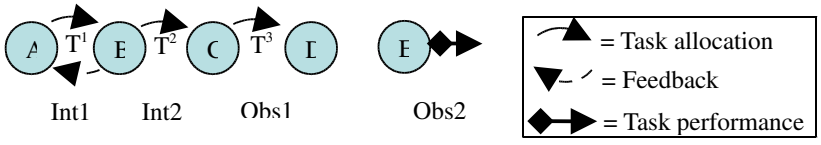

Figure 2. Learning opportunities in a team environment (symbols are defined in Table 1).

Table 1. Learning assumptions from the learning opportunities in the team environment.

Case	Condition (IF)	Deduction (THEN)
Int1 (interaction)	If an agent A allocates a task T^1 to another agent B If an agent A allocates a task T^1 to another agent B	then B knows that A does not have competence to perform task T^1 then task recipient B always gives a feedback to A on it's (B's) capability at T^1
Int2 (interaction)	If an agent C receives a task T^2 from another agent B If an agent B receives a task T^1 from A, and A has asked B to pass on the resulting next task T^2 to agent C	then C knows that B has the competence to perform the task preceding T^2 (i.e. T^1) in the design steps then B knows that: A knows that C has the competence to perform the task T^2. i.e. A knows something about the AMM of C in B
Obs1 (observation)	If an agent A observes another agent C allocating task T^3 to a third agent D	then A knows that: C does not have the competence to perform task T^3; D may have the competence to perform task T^3
Obs2 (observation)	If an agent A observes another agent E performing Task T^4	then A knows that E has the competence to perform the task T^4

Figure 2. The assumptions that an agent makes during the interaction with its environment are given in Table 1.

3.1.4. Task Handling

Using their AMMs agents decide which agent should be assigned the new task. Agents allocate the task to the agent with the highest competence value for the given role. Where multiple agents have the highest value of competence then an agent is randomly selected from the short-listed agents. New tasks are obtained by looking up the design steps as pre-coded in the agent's knowledge base.

3.1.5. Attrition and Member Acquisition

In a design team it is possible that agents may leave the team mid-way through the project. Exit of agents may necessitate the acquisition of new agents into the team to ensure that necessary domain expertise is maintained. This change in team composition may affect team performance. Hence, an agent attrition factor has been implemented.

Implementation: The rate of attrition is modelled as the probability that an agent or a given percentage of agents from the team will leave the team in a given cycle of the simulation run. The agent or agents to exit the team are chosen randomly. An attrition controller is implemented such that at any given time there is at least one agent for each role within the team. If the exit of an agent leaves the team without some expertise on a specific role, another agent with that expertise is immediately introduced into the team.

3.1.6. Team Structure

In general, design teams are either termed as "flat" or "hierarchical" based on their structure and interaction. Flat teams have no hierarchy and no sub-divisions. These kinds of teams are generally used in design exploration and early design phase. Experts are drawn from multiple disciplines and

there are no nominated leaders. A leader may emerge over time based on the interactions among team members.

In traditional organizational practice design teams are often organized into hierarchies. They have nominated leaders and are divided into expertise-based sub-teams. In such teams the design task is passed to the sub-teams with relevant expertise.

Implementation: Team structure has been implemented by constraining the interaction among agents and allocation of design tasks. In these simulations nominated leaders can be specified or a leader can be chosen at the run-time. Run-time leaders are chosen by the client-agent through a bidding process.

3.1.7. Busyness

In a design team agents can learn from the observations they make based on the sense data available to them. This observable sense data includes agent-task interactions and agent-agent interactions. But this learning is subject to their attention. If an agent is busy (may or may not be with the design task) when the observable data is available then the observation is not made in that instance. A "Busyness" factor is introduced for agent's attention to observable data.

Implementation: Busyness is implemented as the probability of an agent at any given cycle that it may not be able to sense the observable data available in that cycle.

3.1.8. Prior Acquaintance

When a new project team is formed it is possible that some of the team members may have a prior acquaintance. This prior-acquaintance means that agents have a partially-developed AMM of known agents at the start of the simulation.

3.2. Model Validation

Three types of validation are discussed in the literature,[15,16] which include: (1) comparing the observed simulation data to actual predictable data for known cases; (2) comparing the observed social behaviors to expected behaviors, which is typical of social groups; and (3) docking the implemented tool against a similar tool by comparing the observed behaviors from simulations using the two different tools.

This paper validates the model by comparing observed data against predictable data for known cases i.e. cases for which values can be theoretically calculated, and demonstrating observable behaviors comparable to typical social behavior.

In this study, the theoretical upper limit, L^{max}, for the number of messages exchanged between agents before task completion can be calculated. For internal validation of the model the observed number of messages should always be less than or equal to the L^{max}.

4. SIMULATIONS

4.1. Simulation Set-Up

A design task with M discrete sub-tasks is introduced to a team of N agents. Each agent has either full or no knowledge of a role corresponding to a specific sub-task, which means that if an agent is given a sub-task it either performs it or cannot do it at all. If an agent cannot perform a given sub-task it sends a refusal message, after which the sub-task is allocated to another agent. If the agent performs the sub-task it informs the sub-task allocator that the sub-task has been performed. The next sub-task to be performed is chosen and the cycle repeats until all the sub-tasks have been performed. If the allocator agent does not know any expert to perform a given sub-task it allocates the sub-task to a random agent and gradually builds a model of the agent's competence at the given role corresponding to the sub-task.

Different types of agents are used in different simulations. The difference in agents is only based on their learning capabilities. Agents of type A^1 are able to interact with other agents, and they learn only from their personal interactions. These agents are not able to observe other data from the environment.

Agents of type A^2 are not only capable of learning from social interactions but they can also observe the other interactions in the design team and learn from those observations.

The number of agents is represented as $N = N^b$ x b, where N = number of agents in the team, b = number of equal-sized task-groups of size N^b. The number of roles is represented as $M = M^1 + M^2 + \cdots + M^n$, where M is the total number of roles in the team, and M^i the number of roles to be performed by ith group. Expertise distribution is represented as $M^i(P^i)$, $M = \Sigma M^i$, where P^i is the number of agents in the team that can perform those roles. N need not equal $\Sigma\ P^i$, i.e., there may be more agents than the number of roles.

For a team where agents start with no prior knowledge of each other's competence the initial sub-task allocations are random until they identify experts with the corresponding role. Each time a sub-task is allocated two messages ("call for proposal" and feedback) are passed. Thus for a team with N agents, M roles, and P agents for each role, the theoretical upper limit on the number of messages exchanged before the task is complete is $(N - P + 1) \times M \times 2$.

Hence, using the calculated L^{max} for a team with given expertise distribution is

$$L^{\max-\text{cal}} = 2 \times \Sigma M^i(N - P^i + 1) \tag{1}$$

The following results have been obtained using Monte Carlo simulations with 120 runs.)[17] Table 2 summarizes the number of messages exchanged in the simulations, where O is the maximum value for the number of messages observed. Table 3 summarizes the simulation results when checking TMM formation. Two types of experiment conditions have been used in terms of team composition: (1) flat teams have no sub-groups; and (2) task-groups; and non-task-groups. For task-groups (TG) a team is divided into groups based on similarity and domain dependency of roles. These kinds of design teams are typical in large organizations. Non-task-groups (NTG) are flat for the purpose of task allocation. However, agents are biased towards observation based learning only within their own groups. These kinds of scenario can be found in geographically distributed project teams.

In some simulations, the factor of Busyness, B is introduced. The probability of agent being busy at any given time is given by B=1/3.

Calculating the value of TMM formed: All agents start with a default value of the competence of all agents on all possible roles. i.e. all agents start with a default value of the TMM. As they learn about the competence of themselves and team members the values for corresponding elements are updated. The value of TMM formed is the percentage of the default values that have been replaced by the actual competence values at the end of the simulation. The overall value of TMM formed is the average of the values of TMM formed for all the individual team members. SD is the standard deviation across the value of TMM formation among the agents, evaluated across the entire team.

4.2. Discussion of Simulation Results:

The number of messages observed (O) in all the test cases is below the $L^{\max-\text{cal}}$ validating the implementation of the computational model.

Table 2. Summary of the number of messages exchanged in simulations.

Agent	N	M	M^1 (P^1) M^2 (P^2)	Runs	$L^{\text{max-cal}}$	O
A^1/A^2	15	10	9(1)1(5)	30	292	229
A^1/A^2	$15 = 5 \times 3$	$10 = 4 + 3 + 3$	4(1)3(1)3(1)	30	100	73

Table 3. Summary of the TMM formation in the simulations.

Agent	Type	N	M	M^1 (P^1) M^2 (P^2)	TMM formed (%)	SD
A^2	TG, B (1/3)	$15 = 5 \times 3$	$10 = 4 + 3 + 3$	4(1)3(1)3(1)	6.90	0.71
A^2	NTG, B (1/3)	15	10	9(1)1(5)	42.49	6.29

As expected agents that can learn from their social observations in addition to their personal interactions have a higher value for TMM formation. What is of interest in this research is to compare the rate of learning in the different cases. A t-test conducted on data in Table 3 shows a confidence level of 95% and 92% respectively. Following trends can be observed and need to be investigated further. (1) As the team size increases the formation of TMM takes longer. (2) Dividing teams into smaller role-based task-groups facilitates the rate of task completion. But this may not necessarily lead to the formation of more developed TMMs. The reduced time for task completion may reduce the opportunity for interaction amongst team members, and hence the lower value of TMM formation. In task-group based teams, the out-group learning is negligible (i.e. AMM of agent's outside the group are not developed). (3) Teams where role-based task-groups are not formed but there are non-task-groups (e.g. by geographical division) leading to observation bias, the TMM models are relatively more developed. But, in this case the rate of task completion is slower as compared to task-group based teams. In flat teams that have non-task-groups the out-group learning is much lower than in-group learning, but it is still significant. (4) When a team that has already worked on one design cycle is chosen for the next design simulation then the rate of task performance is much higher. This is because the TMM is much better developed at the start of the new design cycle. These teams show "primacy", where agents that have already been identified as experts get the design task and other agents that could perform the same task are never explored.

The observations made so far suggest the following. (1) If a team performs well in the first project (in terms of rate of task completion) it may not necessarily perform better than another team that has taken much longer to perform the same task. This is because the other team that may have taken longer in the first project had longer time to develop TMM and may end up out-performing the first team in the next project. Hence, team expertise should be evaluated through longitudinal data and not from singular experiences. (2) To explore the expertise across the team, in the initial stages of team formation a rotation policy on task allocation may be useful. In such a scenario, if a member is already identified as expert in a given role then in the next cycle it may be useful to explore if there are other members who are experts on the same role, rather than allocating the task to the already identified expert. Such measures will help formation of better TMMs and may be useful in case of attrition, though that may negatively affect team performance in the early stages of team formation. Effects of such policies in different scenarios will be tested in future experiments. (3) Preliminary results indicate that division of teams into sub-groups or existence of groups at the time of team formation influence the formation of TMM differently. These preliminary results open up many questions. Should the teams always start as flat teams and later divided into task-groups or should it be the other way round? Should non-task-groups be promoted in teams or should they be discouraged in the early phases of team formation? What is the ideal time the team should at least spend together to form a well-developed TMM? The long term and short term effects of these strategies can be only be understood through longitudinal study involving multiple projects with the same group of agents. Experiments will be conducted to test these scenarios.

4.3. Limitations and Implications for Further Research

The simulation results discussed in this paper investigate only two learning cases, i.e. learning only from personal interactions, and learning from personal interactions as well as through observation of task performance and agent interactions. Each of the learning modes needs to be tested separately, and in superposed combinations for detailed analysis, such as learning from personal interaction and task observation but no interaction observation or learning from personal interaction and interaction observation but no task observation.

These experiments are useful to allow comparisons of the role of different learning modes in a controlled environment, which is rarely the case in a real world environment. To obtain more interesting results and develop a better understanding of the roles of the different learning modes on development of TMM and team performance the following changes to the experiments are planned. (1) Team environments will be made more complex through super-position of situational factors such as attrition

and busyness. (2) The design task used so far can be considered routine design tasks that do not require re-work. A design task involving a basic preference-based model will be used that will require more complex decision making than at present. (3) With the increased complexity of the team environment, design task, and a preference-based design model the AMM/TMM can be more detailed. An agent's modes of learning may still be the same but the scope of learning and knowledge about the team and other agents will increase.

As discussed in the literature the computation models are very useful for testing hypothetical scenarios and advancing social theories. Wherever applicable the results should be tested in real-world scenarios for external validation.

5. CONCLUSION

This paper presents a computational framework developed to study the roles of different modes of social learning on the formation of team mental model. A team mental model is taken here as the knowledge of the expertise distribution within the team, which allows the team to efficiently allocate the tasks to the competent agents, and perform as an expert team. In this model agents learn (a) from their personal interaction with task and the other agent, and (b) by observing the agent-agent and agent-task interactions within the team. Different learning assumptions in a team environment have been identified and represented. The research framework, experiment design and simulation results used for validation of the model discussed in the paper demonstrate the usability of the proposed model for advancing the understanding of the roles of the different modes of social learning on the formation of team mental model and team expertise. Discussions indicate that the computational model can be used as a test-bed to run a range of experiments to study different aspects of social learning, formation of TMM and team expertise.

ACKNOWLEDGMENTS

This research is supported by an EIPRS scholarship. The research is carried out at Design Lab, University of Sydney, Australia.

REFERENCES

[1] Cross, N. (2004). Expertise in design, an overview, Design Studies, **25**(5), pp. 427–441.
[2] LaFrance, M. (1989). The quality of expertise, implications for expert-novice differences for knowledge acquisition, SIGART Newsletter **108**, pp. 6–14.
[3] Candy, L. and Edmonds, E. (2004). Collaborative expertise for creative technology design, URL: http://research.it.uts.edu.au/creative/design/papers/42CandyDTRS6.pdf., accessed on 15/03/06.
[4] Badke-Schaub, P., Neumann, A., Lauche, K. and Mohammed, S. (2007). Mental models in design teams: a valid approach to performance in design collaboration? CoDesign, **3**(1), pp. 5–20.
[5] Klimoski, R. and Mohammed, S. (1994). Team mental model: Construct or metaphor? Journal of Management, **20**, pp. 403–437.
[6] Beng-Chong, L. and Katherine, J. K. (2006). Team mental models and team performance: A field study of the effects of team mental model similarity and accuracy, Journal of Organizational Behavior, **27**, pp. 403–418.
[7] Wegner, D. M. (1986). Transactive memory: A contemporary analysis of the group mind. In B. Mullen & G. R. Goethals (Eds.), Theories of group behavior (pp. 185–208). New York: Springer-Verlag.
[8] Grecu, D. L. and Brown, D. C. (1998). Dimensions of learning in design, Artificial Intelligence for Engineering Design, Analysis and Manufacturing, (in) Duffy, AHB, Brown, DC and Goel, AK (eds.) Special issue on Machine Learning in Design **12**, pp. 117–122.
[9] Wu Zhichao and Duffy, A. H. B. (2004). Modeling collective learning in design, Artificial Intelligence for Engineering Design, Analysis and Manufacturing **18**, pp. 289–313.
[10] Macy, M. W. and Willer, R. (2001). From factors to actors, computational sociology and agent based modelling, Annual Review of Sociology **28**, pp. 143–166.
[11] Gilbert, N. (2004). Agent-based models, dealing with complexity, URL: http://www.econ.iastate.edu/tesfatsi/ABM.DealingWithComplexity.Gilbert.pdf, accessed on 01/12/08.
[12] Gero, J. S. (1998). Conceptual designing as a sequence of situated acts in I Smith (ed) Artificial intelligence in structural engineering, Springer pp. 165–177.

[13] C. Castelfranchi and Y. Lespérance (Eds.) (2001). Developing Multi-Agent Systems With JADE, Intelligent Agents VII, LNAI 1986, pp. 89–103, Springer-Verlag Berlin Heidelberg.
[14] FIPA (2002). Foundation for Intelligent Physical Agents, FIPA ACL Message Structure Specification, URL: http://www.fipa.org/specs/fipa00061/SC00061G.pdf, accessed on 03/06/06.
[15] Carley, K. M. (1997). Validating computational models, Working Paper: Social and Decision Sciences, Carnegie Mellon University, Pittsburgh, PA.
[16] Levitt, R. E., Orr, R. J. and Nissen M. E. (2005). Validation of the Virtual Design Team (VDT) computational modelling environment, CRGP working paper series#25.
[17] Baufer, W. The Monte Carlo method, Journal of the Society for Industrial and Applied Mathematics **4**(6), pp. 438–451.