



AFRL-RH-WP-TR-2009-0110

**Simulation Modeling and Statistical
Network Tools for Improving
Collaboration in Military Logistics**

Ronald Giachetti

**Florida International University
11200 S.W. 8th Street
Miami FL 33199**

Jose A. Rojas-Villafane

**University of Turabo
PO Box 3030
Gurabo PR 00778**

October 2008

Final Report May 2007 to October 2008

**Approved for public release;
distribution is unlimited.**

**Air Force Research Laboratory
711th Human Performance Wing
Human Effectiveness Directorate
Anticipate & Influence Behavior Division
Sensemaking & Organizational Effectiveness Branch
Wright-Patterson AFB OH 45433-7604**

NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report was cleared for public release by the 88th Air Base Wing Public Affairs Office and is available to the general public, including foreign nationals. Copies may be obtained from the Defense Technical Information Center (DTIC) (<http://www.dtic.mil>).

AFRL-RH-WP-TR-2009-0110 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

//SIGNED//
FRANCIS L. TARTAGLIA
Work Unit Manager
Behavior Modeling Branch

//SIGNED//
GLENN W. HARSHBERGER
Anticipate and Influence Behavior Division
Human Effectiveness Directorate
711th Human Performance Wing
Air Force Research Laboratory

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing this collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing this burden to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**

1. REPORT DATE (DD-MM-YYYY) October 2008		2. REPORT TYPE Final		3. DATES COVERED (From - To) May 2007 - October 2008		
4. TITLE AND SUBTITLE Simulation Modeling and Statistical Network Tools for Improving Collaboration in Military Logistics				5a. CONTRACT NUMBER FA8650-07-1-6848		
				5b. GRANT NUMBER		
				5c. PROGRAM ELEMENT NUMBER 62202F		
6. AUTHOR(S) ¹ Ronald Giachetti, ² Jose A. Rojas-Villafane				5d. PROJECT NUMBER 7184		
				5e. TASK NUMBER D2		
				5f. WORK UNIT NUMBER 7184D210		
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) ¹ Florida International University 11200 S.W. 8 th Street Miami FL 33199				8. PERFORMING ORGANIZATION REPORT NUMBER		
 ² University of Turabo PO Box 3030 Gurabo PR 00778				10. SPONSOR/MONITOR'S ACRONYM(S) 711 HPW/RHXS		
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Air Force Materiel Command Air Force Research Laboratory 711th Human Performance Wing Human Effectiveness Directorate Anticipate & Influence Behavior Division Sensemaking & Organizational Effectiveness Branch Wright-Patterson AFB OH 45433-7604						
11. SPONSOR/MONITOR'S REPORT NUMBER(S) AFRL-RH-WP-TR-2009-0110						
12. DISTRIBUTION / AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.						
13. SUPPLEMENTARY NOTES 88ABW cleared on 22 October 2009, 88ABW-2009-4494.						
14. ABSTRACT This report describes the research completed by Florida International University on military collaboration. The research first reviewed and generated a literature review on collaboration. The research created an agent-based simulation model named the Team Coordination Model to study collaboration between a group of people on a defined job or mission. The agent-based model was used to study military operations in an urban terrain scenario. Additionally, we developed statistical tools to analyze social networks to uncover collaboration patterns.						
15. SUBJECT TERMS Collaboration, Social Networks, Teams, Agent-Based Simulation, Social Network Theory						
16. SECURITY CLASSIFICATION OF: Unclassified			17. LIMITATION OF ABSTRACT SAR	18. NUMBER OF PAGES 44	19a. NAME OF RESPONSIBLE PERSON Francis L. Tartaglia	
a. REPORT U	b. ABSTRACT U	c. THIS PAGE U			19b. TELEPHONE NUMBER (include area code) NA	

THIS PAGE LEFT INTENTIONALLY BLANK

TABLE OF CONTENTS

1.0 SUMMARY	1
2.0 BACKGROUND AND PROBLEM	2
2.1 ORGANIZATION OF REPORT	4
3.0 SIMULATION MODEL	5
3.1 TEAM MEMBERS	5
3.2 TEAM ORGANIZATION.....	6
3.3 MISSION.....	6
3.3.1 Task Definition	7
3.3.2 Dependency Definition.....	8
3.4 TASK EXECUTION	9
3.5 SIMULATION FLOW	9
3.6 SIMULATION CONTROL AND STATISTICS REPORTER	11
3.7 COMMUNICATION	11
4.0 SIMULATION ARCHITECTURE	16
5.0 CONTRIBUTIONS	18
6.0 DEFINING THE SCENARIO	19
6.1 DEFINING THE TEAM SCENARIO.....	19
6.1.1 Organizational Parameters	19
6.1.2 Team Members Data.....	19
6.2 DEFINING THE JOB ENVIRONMENT SCENARIO.....	20
6.2.1 Job General Information	20
6.2.2 Defining Task Data.....	20
6.2.3 Defining Dependencies Data	21
7.0 ENTERING THE SCENARIO	22
7.1 MAIN APPLICATION CONTROLS	22
7.2 ENTER THE TEAM DATA	23
7.2.1 Position Panel.....	24
7.2.2 Jobs Skills Panel	25
7.2.3 Team Data Panel.....	25
7.2.4 Team Member Data Form.....	25
7.3 ENTER THE JOB DATA	27
8.0 SIMULATION PARAMETERS	31
8.1 RANDOM BEHAVIOR PARAMETERS	31
8.2 COMMUNICATION PARAMETERS	32
9.0 RUNNING THE SIMULATION	34
9.1 SIMULATION OPTIONS.....	34
9.2 RUNNING THE SIMULATION	34

REFERENCES.....	36
LIST OF ACRONYMS	38

LIST OF FIGURES

Figure 1: Representation of the System Model.....	5
Figure 2: State Diagram for Task Execution	9
Figure 3: Flow of Task Processing Events	10
Figure 4: Taxonomy of Communications Events	11
Figure 5: Implementation of Reciprocal Dependencies.....	12
Figure 6: Negotiation Process between Team Members	13
Figure 7: Team Coordination Model Application Architecture	16
Figure 8: Composition of the Team Member Agent.....	17
Figure 9: Initial Screen.....	22
Figure 10: Main Menu Form.....	23
Figure 11: Enter Team Data Form	24
Figure 12: Team Member Data Form	26
Figure 13: Define Job Environment Form	27
Figure 14: Task Data Form.....	29
Figure 15: Define Job Structure Form	30
Figure 16: Random Behavior Parameters Form	32
Figure 17: Communication Parameters Definition Form	33
Figure 18: Form to Define the Communication Media	33
Figure 19: Simulation Options Form	34
Figure 20: Replication Status Display	35

LIST OF TABLES

Table 1: Attributes that Define a Task	7
Table 2: Type of Interdependencies included in the Model	8

1.0 SUMMARY

This final technical report describes the research findings of the project *Simulation Modeling and Statistical Network Tools for Improving Collaboration in Military Logistics* contracted with the US Air Force in response to *BAA 07-04-HE*. The research was carried out by Ronald Giachetti and Marcus Perry of Florida International University in Miami FL and Jose Rojas-Villafana of Turabo University in Puerto Rico. This report contains an overview of the research problem we addressed and a description of the team coordination model, an agent-based simulation model.

The research project made the following contributions: (1) it created an agent-based simulation model that can be used to model and study human interaction and performance in various job scenarios; (2) the research advanced the analysis of social networks using statistical techniques. Additionally, the research supported two graduate students, one of which is earning his PhD in this area.

2.0 BACKGROUND AND PROBLEM

Military work environments, as exemplified by the logistics process, are highly distributed, complex, and dynamic, which places tremendous information requirements on the individuals who need to work in these environments. For example, providing logistical support to an evolving battlefield depends on countless, highly interdependent decisions by hundreds or thousands of individuals, from senior officers to civilian workers, spread geographically among field command centers, foreign and United States bases, and civilian operated warehouses (Lyons et al., 2006). To succeed, the individuals in the logistics process must work together or be integrated so that the differentiated organizational units are brought together for effective operation of the overall logistics process. In other words, collaboration, teamwork, and cooperation are needed whether achieved through organizational means (e.g., team training, social norms, etc.) or through technology (e.g., groupware systems, email, collaborative work systems, etc.).

Collaboration is when two or more individuals share information and knowledge toward achieving a common goal (McQuay, 2004) or complementary goals (Goldman et al., 1977). There are different types of collaboration. In formal teams, collaboration is called teamwork and is an essential characteristic of teams (McNeese, 1998; Paris et al., 2000). Here we define a *team* as a group of people “*with different tasks who work together adaptively to achieve specified and shared goals.*” These authors point out that the interdependence between team members’ tasks and the requirements for coordination differentiate a team from simply being a group of people working together. Teams are an alternative to perform a task that cannot be completed effectively by a single individual or by the aggregated independent efforts of a group of individuals (Marks, 2000).

Teams use a blend of implicit and explicit coordination to work together (Wang et al., 2001). Implicit coordination occurs when a team member takes actions or decisions that affect the team or other team members activities based on situational information. Implicit coordination depends on team members having precise mental models about the team’s goals, current situation and other members’ needs. Explicit coordination occurs through communication between team members. Communication provides team members with information about other members’ needs and the team’s current situation.

The growing relevance and importance of teams to organizations has encouraged researchers in diverse fields to start paying an increasing amount of attention to team performance since the 1980’s (Baker & Salas, 1997), and to the characteristics and processes that contribute to it (Marks, 2000). Stewart (2006) concludes that teams can be designed for high performance and design factors exist for the design of team composition, task structure, and organization context. Salas et al., (Salas et al., 2005) provide an extensive list of factors required to modeling and design teams.

The design factors have been defined and understood through traditional research methods using human subjects, such as case studies and controlled laboratory experiments. Nonetheless, there is limitation to the traditional research methods using human subjects to being used to design and to further advance the research on team performance. To further the study of team coordination

and team performance, experimental methods or tools should be developed to overcome the limitations of case studies, survey assessments, or laboratory studies to analyze a large number of factors, at large range of levels, for different types of teams.

One alternative to develop and test theories on team coordination and performance is the development of computational models. As discussed and exemplified in Burton & Obel (1995), computational models have been used for decades to test hypotheses, explore organizational processes, and for the study of theoretical and practical issues. These authors argue that validity of computational models in organization science is a function of the purpose of the study, the experimental design, and the computational model. However, up to a decade ago, there was limited use of computers as tools in the design of social aspects of organizations (Jin et al., 1995). Computational models have the potential “*to move theories of organization beyond empirical description to generative formalizations*” (Carley, 1995). Wang et al., (2001) asserted that the study of teams’ coordination issues will be benefited by combining various discipline perspectives through modeling and empirically studies.

Simulation models provide many advantages as a tool to conduct studies about teams and organizations. A simulation model allows a precise implementation of social sciences theory, and allows a precise and testable representation of conceptual entities with their functions, structure, and behaviors (Kunz et al., 1998). Researchers using simulation models can measure and test with precision some variables while controlling other factors and can test a wide range of organizational variables (Kim & Burton, 2003), overcoming the limitations of controlled laboratory studies with human subjects. Furthermore, for highly complex systems, such as teams, computer simulation is the only viable method for system analysis and evaluation (Clymer, 1992).

Beyond the need to have a computational tool to further the study of teams, is the need of having a computational tool to design teams. Carley (1995) pointed that organizational design is one the most relevant issues within an organization because changing its design allows organizations to adapt to its task environment and alter its performance. This author also pointed out that computational models are particularly useful to evaluate organizational design alternatives.

The use of organizational simulation to model team performance is becoming “increasingly important,” but the field is still in its infancy (Salas et al., 2005). Salas et al., (2005) points out that there is a need for better simulation tools that capture the complexities of team performance. These tools should be “fluid, flexible, and adaptable” and should capture realistic team performance.

The reason for selecting an agent-based architecture for the simulation model is the close correspondence to how we view team composition and team processes. Teams are composed of members, each with different skills, knowledge, and motivations. Team members act autonomously and must coordinate their work with the work of other team members. Coordination is primarily via communication between members. The agent-based simulation allows the implementation of this view by the modeling of each team member as an autonomous agent and the agents coordinate by sending/receiving messages.

2.1 Organization of Report

The report is organized as follows. Section 3 describes the simulation model. We describe the team member agents, mission agents, how communication is handled, and how tasks are processed. Section 4 briefly describes the simulation architecture. Section 5 summarizes the main contributions of the simulation model. Section 6 explains how to create a simulation model using the software. Specifically, it describes how to create a team and create a job or mission. Section 7 explains how to enter data into the simulation model. Section 8 describes the simulation parameters and explains how to set them. Section 9 describes how to run the simulation model.

3.0 SIMULATION MODEL

To study collaboration and teamwork, we developed an agent-based simulation model called the Team Coordination Model (TCM). The simulation is built using the agent infrastructure CybelePro written in Java. In CybelePro, agents are defined as, “a group of event-driven activities that share data, thread, and execution concurrency structure.” Activities are internal to the agent, and act on internal data in response to incoming events. Other agents are incapable of accessing or manipulating the internal data, which reinforces the notion of agent autonomy. In CybelePro, agents generate and deliver events that may or may not, trigger the desired response in other agents of the multi-agent simulation.

The TCM can emulate teams who are assigned a job or mission (Figure 1). This is reflective of many military scenarios where teams are goal-oriented and have a defined mission. The mission is modeled as a network of tasks. Each individual task is performed by a single team member. In the execution of the mission, the team members will communicate with each other to coordinate their work. Each mission has well-defined termination criteria, which when met ends the simulation.

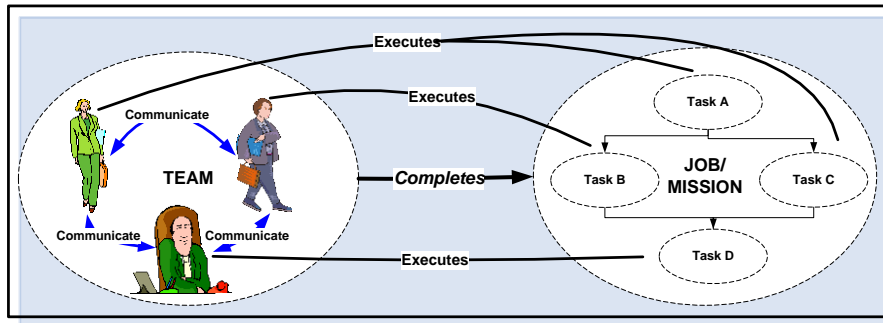


Figure 1: Representation of the System Model

3.1 Team Members

Each team member is modeled as an autonomous agent consisting of three activities:

1. Task Processing Activity – Handles the processing of the assigned mission tasks. This includes performing a task, momentarily stopping a task due to an interruption, preempting a task, and reworking a task if an error or failure occurs.
2. Decision-Making Activity – Orders the assigned tasks for execution. Decides whether to start/end a task. Determines task assignment based on the shared mental model.
3. Coordination Activity – Handles the communication with other team members.

A central component of the simulation is the concept of a shared mental model. A shared mental model (SMM) is a theoretical concept that provides an explanation of how effective teams are able to anticipate other team member’s needs and utilize a push type of communication whereby

the team member sends unsolicited information assuming the receiver will need this information. The premise of a SMM is that individuals form a mental construct about their roles or responsibilities in the team and the roles of their team members. In other words, individuals formulate plans on how to execute the team mission.

There are two possible mission scenarios under which a mission may start. The first scenario is when a pre-defined task assignment plan exists, called a Task Organization Mechanism. In this scenario, the task assignment is given by the plan. The second scenario is when there is no plan and part of the team mission is to formulate a plan. It is in this scenario that the SMM plays a major role. When no task organization mechanism is available, individuals will form their mental plans based according to two rules:

1. Assign a task to the team member that is best qualified according to the match between the task's required skills and the team member's skills.
2. Distribute the task assignment as evenly as possible.

The first rule assumes that a good task assignment is formed by best meeting individual task requirements. The second assignment recognizes that overall performance is improved if the division of work is more even. It is clear that this heuristic approach will not lead to an optimal task assignment, but the purpose is to emulate how people make mental assessments of task assignment. The simulation model developed an algorithm to implement the rules and arrive at a task assignment.

In the model, the SMM is implemented by manipulating two probabilities. First, is the probability of agreement between team members of task assignments. Each team member creates a mental model of which team member should be assigned to each task. The presence of a strong SMM leads to greater agreement among team members on task assignment. Second, is the probability of unsolicited information being sent by one team member to another – one of the observed behaviors of teams with a strong SMM.

3.2 Team Organization

A team can be organized into a hierarchy defined by a '*reports to*' relationship. In this relationship, each team member has a supervisor. The team leader is the member at the top of the hierarchy. Supervisors resolve task assignment conflicts, can assign task priority, and can over-ride team member decisions.

The level of organizational centralization will have an impact on the process of dividing the job's tasks among team members. In an organization with a high level of centralization, it will be most likely for team members to consult or confirm with the team leader the tasks they should perform, and it will be less likely they object their assignments when they differ. On the other hand, in low centralization organizations, team members will rely more on their mental models for decisions on their task assignments.

3.3 Mission

A mission is defined as a collection of related work tasks with a defined completion state such as the generation of a certain output. Thus, we exclude modeling such missions that have unclear

outcome objectives. A mission is modeled as a network of tasks similar to a program evaluation and review technique (PERT) network. The network nodes represent the tasks and the arcs connecting the nodes represent the task dependencies. The network representation is intended to model a mission as a collection of partially ordered tasks. To model the mission the modeler needs to define the tasks and the dependencies.

3.3.1 Task Definition

Tasks are represented with the attributes shown in Table 1. The task duration is modeled as a triangular distribution. The triangular distribution is used because it is easy to implement and frequently there is insufficient data to fit to other probability distributions. The triangular distribution only requires the modeler to estimate the minimum, the most likely, and the maximum duration for the task. Future extensions to the model can allow for greater flexibility in modeling task duration without affecting other aspects of the model.

Table 1: Attributes that Define a Task

<i>Attributes</i>	<i>Description / Values</i>
Task ID	Identifier of the task
Complexity	Cognitive demand on team members
Duration	Defined as a triangular distribution (min, most likely, max)
Priority	Indicate the recommended order in which a task should be performed related to the other tasks in the job.
Assigned Agent	Team member recommended or assigned to perform the Task
Skills	Set of skills required to perform the tasks adequately
Visibility	Indicates if team members can perceive the status of this task from the environment, visually or through another sense, without the need to receive communication from the agent working it.

A task may require one or more skills for performance. The modeler may define any skill needed to perform the task. Team members are modeled as possessing skills at one of three levels of high, medium, and low. The match between the team member's possession of a skill and the task requirement for a skill affects two aspects of task performance: (1) the duration of the task, and (2) the probability of task failure. For example, if a task is specified to require the skill Navigation and the performing team member possess the Navigation skill at a low level, then it will take longer to execute the task, and the probability of task failure is greater. Exactly how the mis-match affects task performance is set by the modeler as a Simulation Control parameter discussed later in the report.

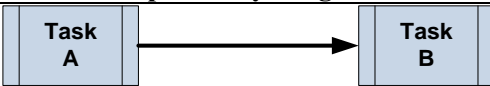
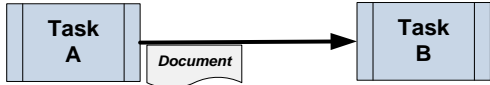
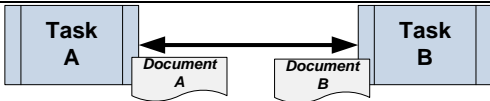
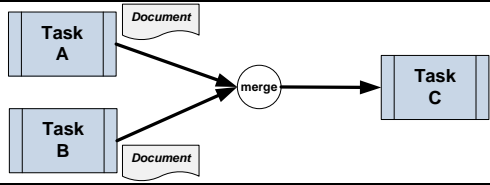

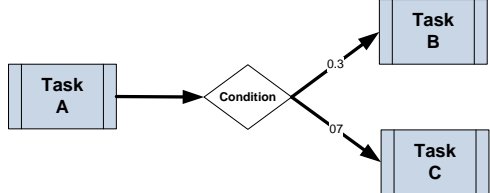
The output of a task may be visible for other team members besides the team member performing the task. If a task or its output is visible, the need for communication is reduced. However, if a task is not visible then there is a communication need for the performing team member to alert others as to the status of the task.

3.3.2 Dependency Definition

The modeling of dependencies between tasks is critical to the modeling of collaboration in a team. Without dependencies there would be little need for collaboration. Table 2 shows all the possible dependencies between tasks. The first four dependencies are deterministic, meaning there is no uncertainty or probabilistic characteristics to the dependency. The Controlled Sequential dependency is a pre-requisite dependency that says Task A is a predecessor of Task B. The Information Sequential is similar, except the dependency is represented by information (a ‘document’) that must be received by Task B to begin processing. The Reciprocal dependency says that Task A and Task B may be done in parallel, but they share information during their execution. The Information Merge dependency is a sequential dependency but Task C depends on two or more predecessor tasks.

The latter two dependencies shown in Table 2 are conditional dependencies. In these dependencies uncertainty is introduced by the probability associated with the condition. In the Single-Conditional Control dependency, Task B is only done if the condition is satisfied. This establishes an “IF *condition* THEN do *Task B*” logic. The condition would likely be an external environmental event that the modelers can assign a probability to. For example, if modeling a fire fighting team, the condition might be whether there is high wind. In this case, Task B is only performed if high wind is present. There may be multiple possible tasks dependent on the condition; this is shown as the last dependency in Table 2.

Table 2: Type of Interdependencies included in the Model

Dependency	Dependency Diagram
Control Sequential	
Information Sequential	
Reciprocal	
Information Merge	
Single-Conditional Control sequential	
Conditional Control Sequential	

It is possible that the mission structure is constructed entirely of deterministic dependencies, in which case every simulation of the mission will follow the same path in the network. The only uncertainty is the task durations and the task failures. If the mission structure includes the conditional dependencies, then it is possible that each simulation is different depending on the random behavior of the conditions. In summary, there are three types of mission uncertainty that affect mission performance. These are: (1) task duration, (2) task failure, and (3) conditional task execution. The representation of this uncertainty is needed to model the environmental uncertainty found in actual mission scenarios.

3.4 Task Execution

A task is modeled as a state machine that transitions between the states shown in Figure 2. The initial state is *Hold*, which describes a task for which predecessor tasks as defined by the network of dependencies are not yet completed. Once all the predecessor tasks and information inputs are available the task state transitions to *Pending*, meaning it ready to be executed by a team member. When a team member starts work on the task it transitions to *On Process*. Note that a task may cycle back and forth between *Pending* and *On Process* if the team member allocates their attention elsewhere; for example if the team member switches to another task. Once the task duration is reached the task transitions to *Done*. If the task fails during execution, then it requires rework and transitions to the state *Pending Rework*. When the rework is completed, the task transitions back to *On Process*. The *Not Required* state is reached in conditional probabilities where the condition is not met and the task is consequently not required as part of the mission. A task is complete when it reaches either the *Not Required* or *Done* absorbing states.

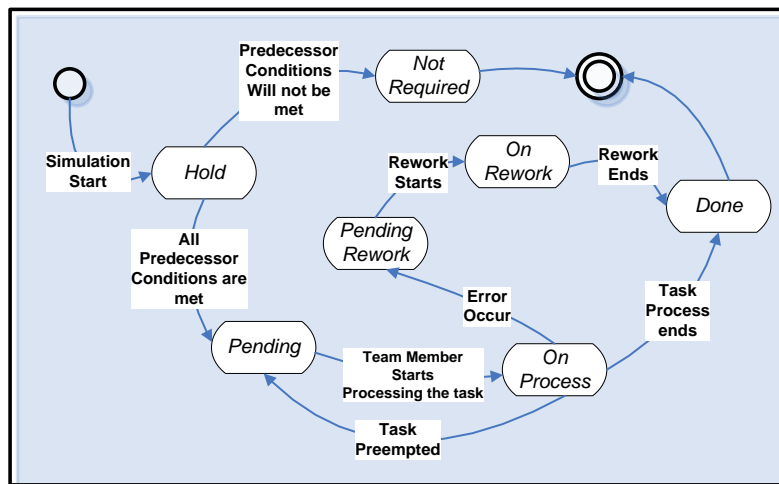


Figure 2: State Diagram for Task Execution

3.5 Simulation Flow

Figure 3 shows the simulation flow of events. The environment agent releases the mission (or job) to all the team member agents. Each team member forms a mental model of the mission as described in Section 3.1. Once the task assignment is resolved, the tasks are placed in the task

list of each team member. The team member uses his decision logic to select a task to begin execution on. If no tasks are ready for processing, then the team member must wait.

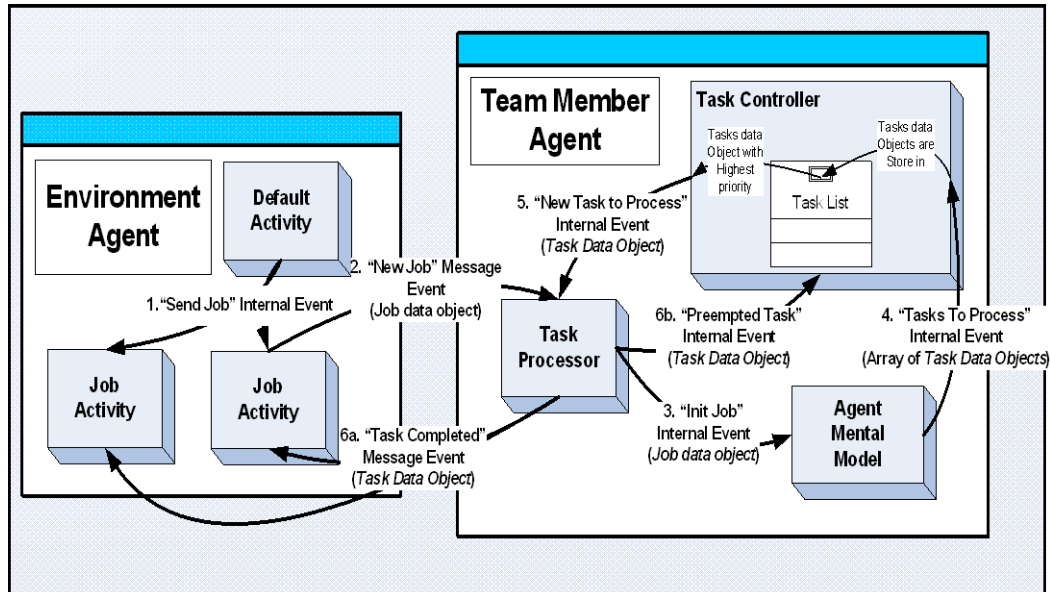


Figure 3: Flow of Task Processing Events

During the task processing, the team member agent may receive communications from other team members. These messages interrupt the task execution. The team member agent pauses task execution, attends to the message, and then resumes task execution based on the priority of the message. Also, during the execution of the task, an agent could make a mistake that requires the task to be reworked. A task to be reworked is return to the Task List in the Task Controller object.

Once a task is complete, the agent sends a message notifying the completion event to other team members. The decision to send the notification to all team members or to only those with successors is made according to the simulation logic. If the agent decides to send the notification to every team member, then it will increase the communication overhead. On the other hand, if the agent sends the notification only to team members with successor tasks, this might cause problems because the task assignment might change and the actual agent assigned the successor task would not then receive the message.

The Task Data object of a completed task is transferred back to the job activity object in the Environment agent. Once the team member agent successfully completes the task, he will start the next task in his list and the above process is repeated until no more tasks are in the *Pending* state. If an agent does not have a task to process, it will remain idle but continue to receive messages and update task information. If a task reaches a *Not Required* status, the team member agents also sends the Task Data object back to the Environment agent. The simulation ends

when all the Job objects in the Environment Agent receive all of their task data objects back from the team.

3.6 Simulation Control and Statistics Reporter

The Simulation Controller Agent controls the creation of the rest of the agents and controls the start and end of each simulation replication. The Statistic Reporter agent collects the statistics from the other agents and prepares the report of each replication.

3.7 Communication

Throughout the mission, the team members communicate with each other. Figure 4 shows a taxonomy of communication events. A communication event is either asynchronous or synchronous and is tied to the media type used. For example, an e-mail is asynchronous while a telephone message is synchronous. Each communication event is classified by its intent, content, and message. The communication intent indicates the purpose of the message: to transfer, request, or acknowledge information. The communication content refers to the type of message to be transferred or requested. The content can be information, a decision, or an action. The message specifies the information, decision, or action of the communication event. The combination of intent, content and message defines the actions taken by the receiver to process the event.

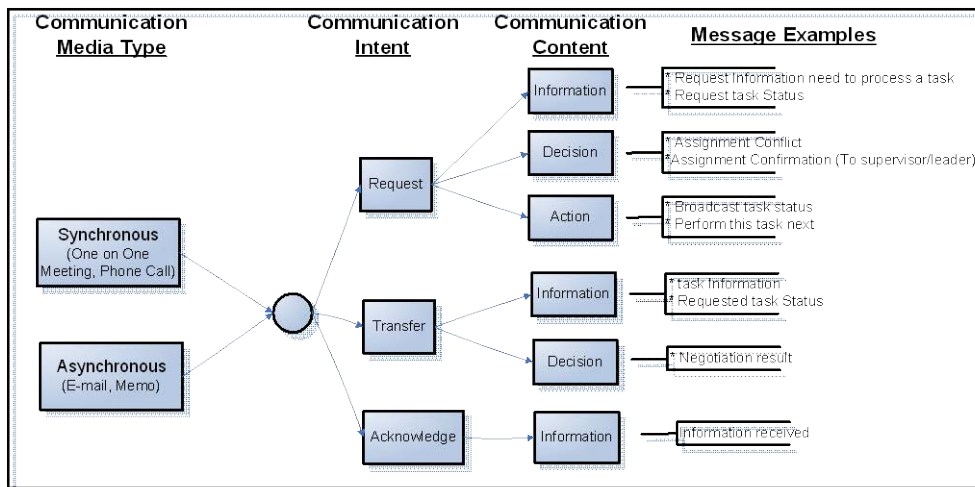


Figure 4: Taxonomy of Communications Events

The defined communication events between team members are:

- *Task Status* – These messages are triggered when a team member completes the execution of a task or when a team member requests the status of a task. The owner of the task will send a message to those team members that are in charge of executing the successor tasks to notify about the status of the predecessor task. If

the dependency type between tasks is information sequential and the status of the predecessor task is *Done*, then the message contains the information document.

- *Request Information/Status about Task* – These messages are triggered when a team member cannot perform any of his tasks because they are all in the *Hold* state. In this case, the team member will select the highest priority task and sends a request message to every team member that is assigned to the predecessor tasks. The team member receiving this message will respond with one of the following messages:
 - *Task Status* – If the receiver of the request has the task assigned.
 - *Not My Task* – If the receiver of the request does not have the task assigned.
- *Communication Due to Reciprocal Dependencies* – The model assumes that in reciprocal dependencies, the information from the reciprocal task is needed at the mid-point of a task execution. This assumption simplifies what is more likely, several smaller communications that would normally occur throughout the task execution. In implementation, a team member will send a message with the reciprocal information to the team member assigned to the reciprocal task. If the reciprocal information has been received, the team member continues the execution of the task; otherwise he/she is blocked until the reciprocal information is received (Figure 5). If a team member is blocked for a reciprocal task, he/she sends a *Request Task Status* message to the team member assigned to the reciprocal task.

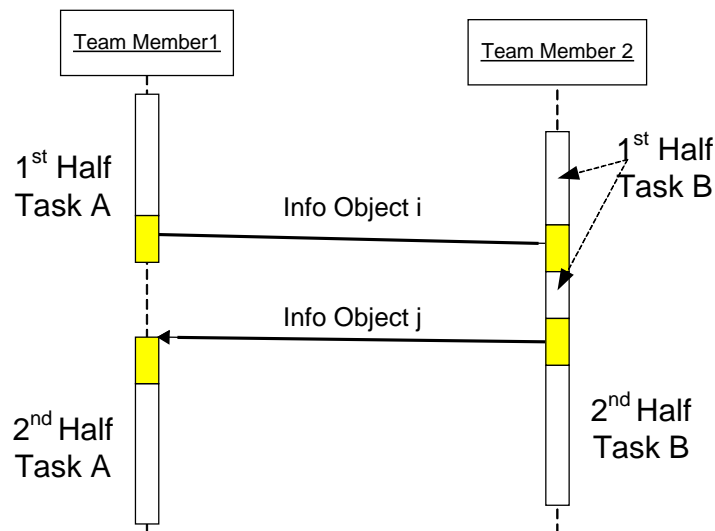


Figure 5: Implementation of Reciprocal Dependencies

- *Negotiation* - A conflict emerges between two team members when both are (or think they are) assigned to the same task. Team members employ negotiation to resolve this conflict (Figure 6). The negotiation process starts when a team member receives a notification from another team member that is starting a task.

If the receiver of the task recognize the task as one of the task he/she is suppose to process, he/she will send a *Negotiation Required* message to the sender of the notification. When the member processing the task receives this message, he/she will evaluate who should keep the task, and returns a negotiation response. This response could be one of the following messages:

- *Yield Task* – the member who starts the negotiation will keep the task assignment.
- *Claim Task* – the member processing the task will keep the task assignment.
- *Negotiation Tie* – neither of the members wins the task over the other. The team members will ask the team leader to solve the conflict.

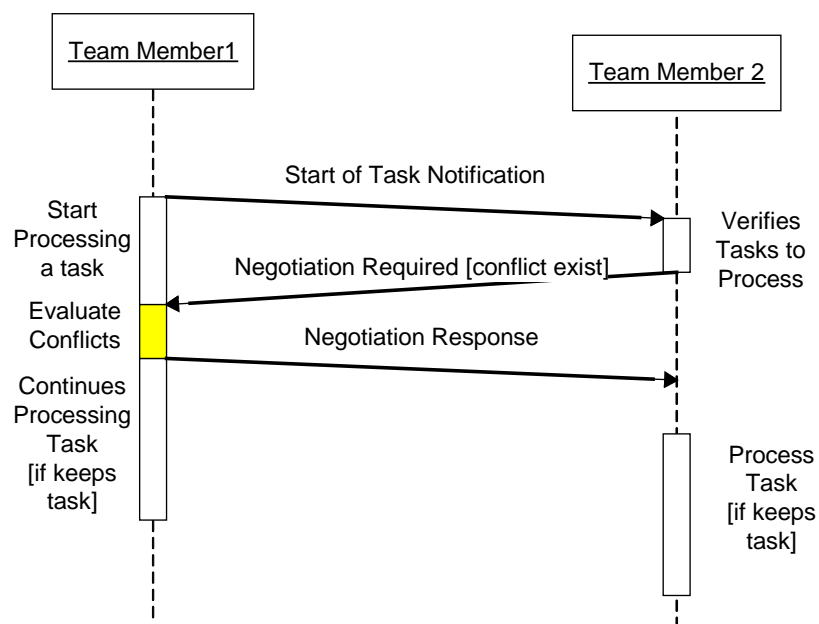


Figure 6: Negotiation Process between Team Members

This research assumes that the negotiation is solved using a simple set of rules. The team member that keeps the task assignment is the one that can finish the processing of the task first. Tie-breakers are solved using the seniority of the team members, the more experienced get the task. If the tie continues after the first two rules are applied, then the conflict is solved by the leader.

Leadership is a key component for the success of a team. The model requires that one of the members be the team leader which performs the following tasks:

- Make changes in assignments
- Confirm a task assignment
- Order a team member to report the status of a task
- Resolve the conflict in assignments between team members

The frequency the team leader performs these tasks depends on the centralization level of the team. Since the leader depends on the communication with the rest of the team to perform his/hers responsibilities, the centralization level increases the coordination load of the team. The communications between team members and the team leader included in the model are:

- *No Task To Process* – A team member (sender) notify the leader that he/she are idle because all of his/hers tasks are on “*Hold*” state or he/she have no tasks assigned. The leader responds with one of the following messages:
 - *Perform Task* – If the leader finds a task assigned to the sender with a “*Pending*” status (the sender might not have received a predecessor information); OR otherwise, the leader finds a task with “*Pending*” status which required skills are a good match with the skills of the sender.
 - *Task Status*- If the leader do not found a tasks with “*Pending*” status to assigned to the sender, then he/she will reply with the status of every predecessor task with a, “*Done*” or “*Not Required*” status, of the highest priority task assigned to the sender. Also, the leader will send a *Request about Task* message to every team member with a predecessor (of the sender’s task) task assigned that has a “*Hold*” or “*Pending*” status.
 - *No Task to Perform* – The leader sends this message if he/she do not find a task assigned or to assign to the sender.
- *Confirm Assignment* - When a team member is ready to start the execution of a task, it might ask the leader to confirm if the task is the one he/she is suppose to do. The probability of a member sending a confirmation message is influence by the team centralization level. When the leader receives a confirmation message, he/she will respond with one of the following messages:
 - *Go Ahead* – If the task is the one the team member is suppose to do next.
 - *Perform this Task Instead* - If the member is suppose to execute another task instead
 - *No Task to Perform* – if the leader found no task for the member to execute.
- *Identify Owner* – This message is send to the leader when a team member receives a “*Not My Task*” message from a team member. The leader, after receiving the “*identify*” message, founds out who is the owner of the task in question and returns a “*Task Owner*” message.
- *Assignment Conflict* - When the leader receives this message from a team member, he/she determines who of the two members involved in the conflict is supposed to execute the task. Then, he/she sends a “*Perform Task*” message to the member responsible for the task, and a “*Drop Task*” message to the other member involved.
- *Decide to Accept a Synchronous Communication Request* – When a team member receives a request from a team member to engage in a synchronous communication, he/she might do one of three behaviors:
 - *Accept the Request*- If the team member is idle, performing a non-urgent task, or preparing a low-priority asynchronous message, then the

member will establish synchronous communication with the team member requesting it.

- *Reject the Request*- If the team member is performing an urgent task or creating a high priority asynchronous message, then he will communicate to the team member requesting the synchronous communication that he rejects the request. Then the requesting team member will save the message and might try to communicate later on.
- *Ignore the Request*- If the team member is busy on another synchronous communication, is performing a task, or working on an asynchronous communication, the member can choose to ignore the request and do nothing about it. The requesting team member will try again to establish the communication.

4.0 SIMULATION ARCHITECTURE

The simulation model is written in CybelePro and Java. CybelePro provides a ready framework for developing agent-based simulations. Figure 7 shows the agents and how they are related. Figure 8 shows that a team member agent is composed of three classes of coordinator, decision making, and task processing. The coordinator activity manages all communication and coordination work. The decision-making activity sets priorities such as which job task to do first. The task-processing activity completes each task.

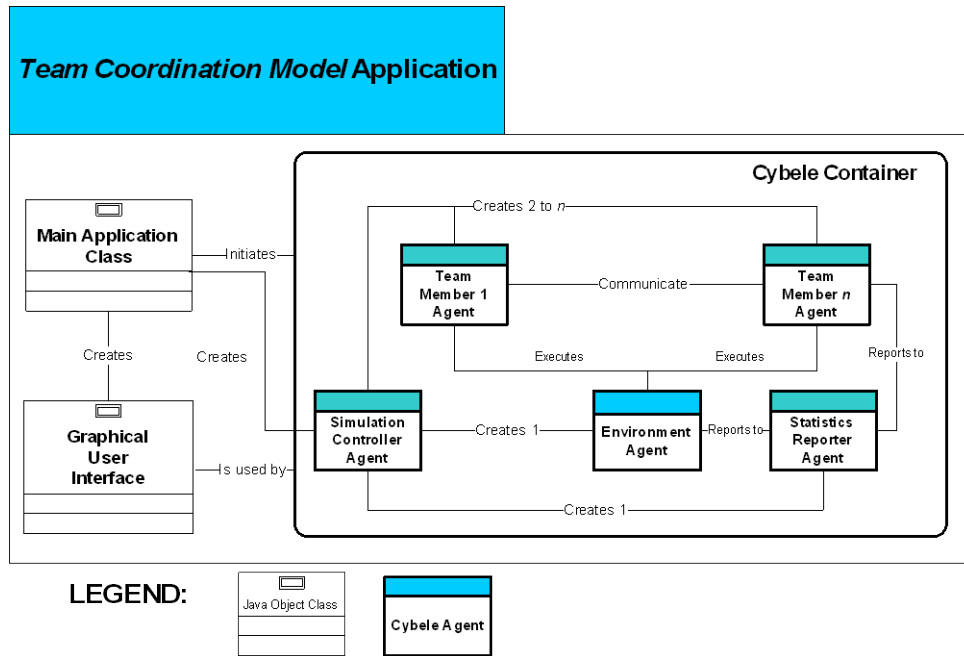


Figure 7: Team Coordination Model Application Architecture

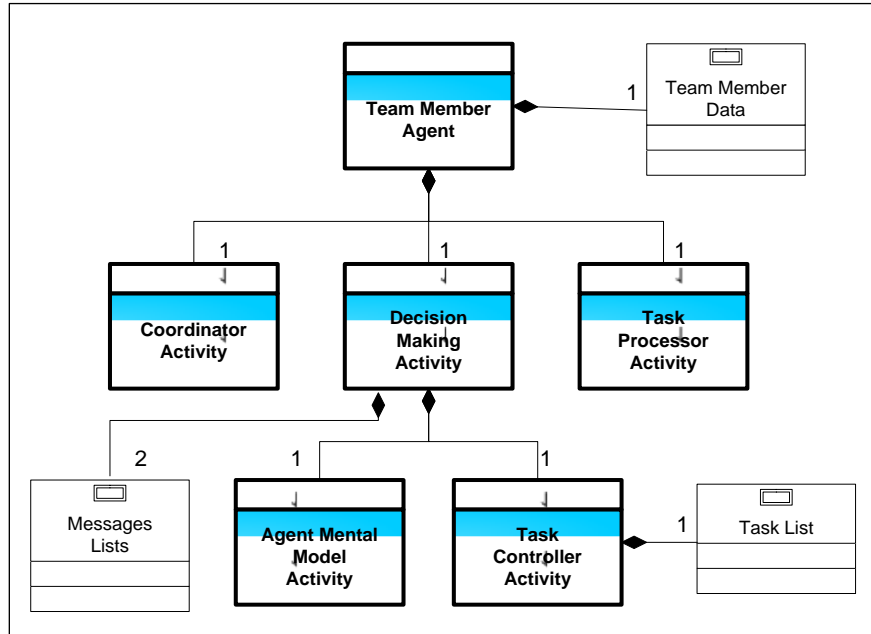


Figure 8: Composition of the Team Member Agent

5.0 CONTRIBUTIONS

The main contribution of this research is the agent-based simulation model to analyze collaboration in a team and the team's performance. Several advanced in agent-based modeling of teams and collaboration has been achieved. Existing simulations tend to have static mission structures such that all the task dependencies are given. The agent-based simulation model can also represent stochastic mission structures such that task execution is dependent on conditions representing uncertainty in the environment. The inclusion of the conditional dependencies gives rise to what we refer to as a dynamic job structure.

6.0 DEFINING THE SCENARIO

This section describes the information required to define a scenario for the Team Coordination Model simulation. The scenario is composed of the team definition and the job environment definition.

6.1 Defining the Team Scenario

To define the team scenario you need to define the positions, skills, and team data. The positions are the team members' roles within the organization or within the team. The skills are the ones required by the team members to perform the tasks of the jobs. Each team member requires a proficiency rating on each skill. The rating is done in a high, medium, or low scale.

The team data is composed of the team members' data and the organizational parameters.

6.1.1 Organizational Parameters

1. Team Centralization- refers to the hierarchical level that has authority to make decisions and then measure in high, medium or low.
 - a. High Centralization – almost every decision goes through the team leader, team members ask the leader for task confirmation.
 - b. Medium Centralization – fewer decisions go through the team leader.
 - c. Low Centralization – team members seldom consult with the leader.
2. Team Formalization – refers to the degree of formal/ written communication and documentation in the organization.
3. Task Assignment Plan – the team has some plan defined on how to perform the job.

6.1.2 Team Members Data

1. Team Members ID – an identifier for the team member.
2. Members' Supervisor – the team leader.
3. Experience with the Team – relative experience of the team member working with the team or similar teams measured in high, medium, or low.
 - a. High Team Experience – the team member is very knowledgeable about team processes and/or has worked on many previous occasions in similar teams.

b. Medium Team Experience – the team member is somewhat knowledgeable about team processes and/or has worked once or twice before with similar teams.

c. Low Team Experience – it is the first time the team member has worked with a similar team.

4. Teamwork Skills – level of training and/or skill of the team member working with teams in general, measured in a high, medium, or low scale.

6.2 Defining the Job Environment Scenario

The jobs are defined by the job general information, the tasks, and the dependencies.

6.2.1 Job General Information

1. Job Identifier – a name to identify the job.

2. Due Date (in minutes) – time to complete the job after the start.

3. Release Time – time to start the job after the start of the simulation. Usually zero when just one job is included.

6.2.2 Defining Task Data

1. Task Identifier – a name to identify the task.

2. Task Complexity – refers to the degree of cognitive demand of task on team members, measured as high, medium, or low.

a. High Complexity – the task requires a high degree of concentration and the risk of committing a mistake is high.

b. Medium Complexity – the task requires a medium degree of concentration and the risk of committing a mistake is medium.

c. Low Complexity – the task is fairly simple, and the risk of committing a mistake is low.

3. Task Priority – relative order in which the tasks should be executed. For example, if task A should be executed first it will have a priority of 1. If two tasks can be executed simultaneously, they should have the same priority number.

4. Assigned Agent – team member that most likely will execute the task.

5. Task Outcome Visible – everybody from the team will be able to know when the task is completed.

6. Task Duration – duration distribution of the task defined as minimum, most likely, and maximum duration.

7. Skills Requirements – skills required by team members to successfully perform the task.

6.2.3 Defining Dependencies Data

There are six types of dependencies defined by the model. The six dependencies are:

1. Control Sequential – Requires only the predecessor and the successor tasks to be defined.
2. Information Sequential – Requires the predecessor and the successor tasks to be defined, as well as the information to be passed between them.
3. Reciprocal – The reciprocal dependencies occur when two tasks depend on information from the other before they can be completed. Besides the two tasks involved, this dependency requires the information documents to be defined as well.
4. Single Conditional Sequential – This dependency is used when there is a task that may or may not be performed depending on the result of a predecessor task. This dependency requires the probability that the successor task will be performed.
5. Conditional Sequential – This dependency is used when one of two or more successor tasks will be performed depending on the result of a predecessor task. This dependency requires the probabilities that each successor task will be the one performed.
6. Merge - A merge dependency is based on a logical OR; in this dependency a task can start when any of one or more predecessor tasks are completed.

7.0 ENTERING THE SCENARIO

This section provides information on how to enter the information in the application to create a scenario. To create the scenario, first you need to enter the team data and then enter the jobs information.

7.1 Main Application Controls

The *Initial Screen* (Figure 9) has a graphic to depict the structure of the simulation model. To continue to the main menu press the large button on the bottom.

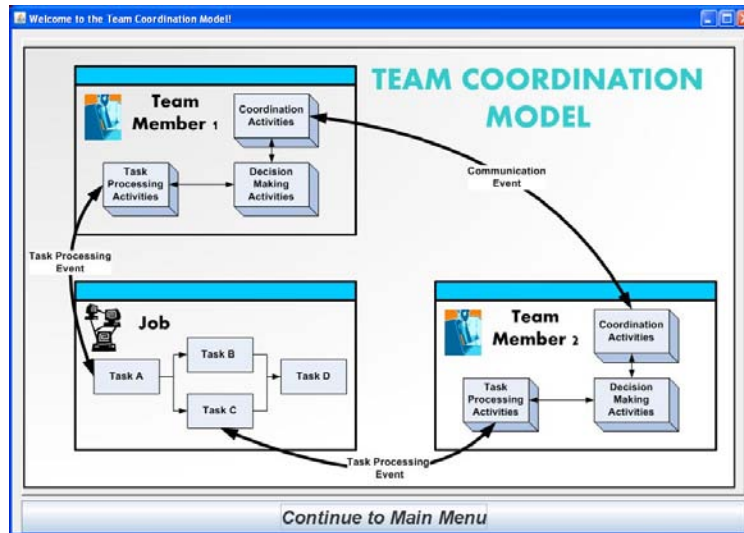


Figure 9: Initial Screen

The *Main Menu* (Figure 10) lets you navigate to the submenus to create the simulation scenario. The recommended sequence is:

1. Define Team
2. Define Jobs
3. Random Behavior Parameters
4. Communication Parameters
5. Simulation Options
6. Run Simulation

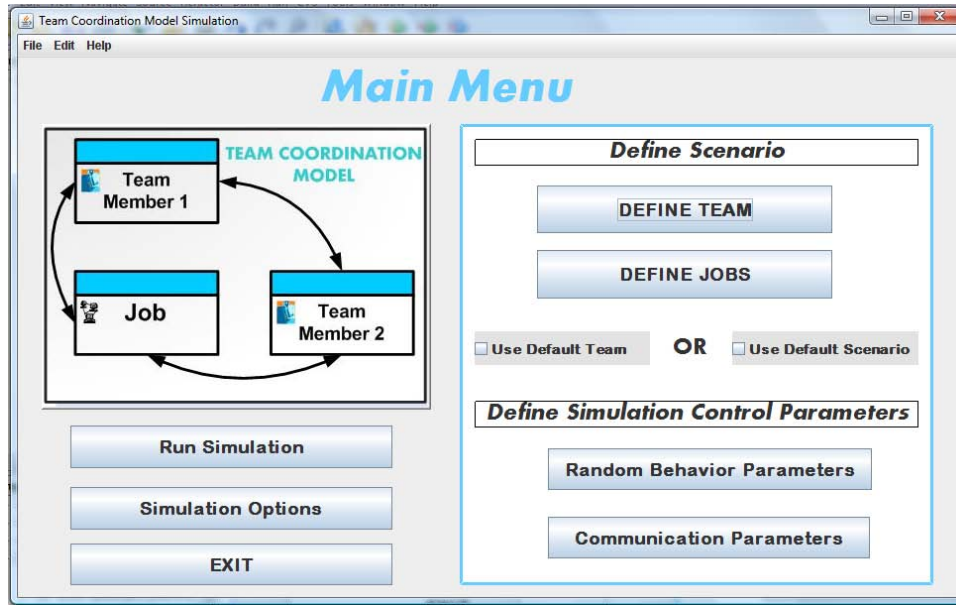


Figure 10: Main Menu Form

For demonstration purposes, you can choose to use the default team or run the whole default scenario by selecting the appropriate checkmarks.

7.2 Enter the Team Data

Enter the Team Data is shown in Figure 11. The team is defined in three panels of controls:

1. Positions Panel
2. Job Skills Panel
3. Team Data Panel

Figure 11: Enter Team Data Form

7.2.1 Position Panel

The *Positions* panel contains the controls to define the positions or ranks of the team members. It contains the following controls:

- *Current Number of Positions counter* – displays how many positions have been already defined.
- *Add Position button*– opens the *Enter New Position* form to add positions to the list.
- *Use Default button* – substitute the current list with a predefined set of positions. Users can add new positions to the list after inserting the defaults.
- *Clear List button* – this button erases all the positions currently in the list.
- *Display List button*- this button opens the *Display Positions List* form. This form displays all the positions currently in the list with an associated checkbox. The users can use the checkboxes to mark the positions that they want to delete (if any) when pressing the *Save Changes button* of this screen.

7.2.2 Jobs Skills Panel

The *Job Skills* panel contains the controls to define the set of skills required to perform different tasks of the jobs or missions. The controls and processes of this panel are analogous to those of the *Positions* panel.

7.2.3 Team Data Panel

The team is defined in the *Team Data Panel*. This panel contains the following controls:

- *Current Team Size indicator* – displays the current number of team members in the team.
- *Team Centralization Level selector*- users select from a predefined list, the level of *Centralization* of the team organization.
- *Team Formalization Level selector*- users select from a predefined list, the level of *Formalization* of the team organization.
- *Task Organization checkbox* - if selected, indicates that the team uses a form of explicit task organization mechanism to coordinate the job.
- *Add New Member button* – opens the *Team Member Data* form to define and add a new team member. The team members can be added just one at a time.
- *Modify Team Members button* – users can press this button to view and modify the information of existing members. This button opens the *Team Member Data* form loaded with the information of the first team member in the team. Users can leave the data as it is, modify any of the parameters, or delete the whole team member data from the team. This option does not allow the creation of new team members.

7.2.4 Team Member Data Form

The *Team Member Data* form (Figure 12) contains all the controls required to define a team member. The controls are divided in three panels: Team Member Identity, Team Work, and Job Skills. In addition to these panels, this window contains four buttons:

- *Next Member button* – this button saves the data from the current team member and loads the data from the next one. This button is only enabled when the form is open to modify team member's information.
- *Save button* – if the form is open in the “add new member” mode, this button creates new team member data, based on the current form data and adds it to the team data, then closes the *Team Member Data* form and returns to the *Team Definition* form. If the form is open in the “modify” mode, the button just saves the information from the current team member displayed and closes the form.
- *Cancel button*- closes the form and the current data is lost.
- *Delete button*- This button is only enabled when the form is open in the “modify” mode. When activated, this button deletes the data of the team

member displayed from the team information, and then loads the next member information (or the previous one if the deleted member was the last).

Figure 12: Team Member Data Form

The *Team Member Identity* panel contains the controls to define the identity and role of the team member. The controls contained in this panel are:

- *Member ID textbox* – to enter the name or ID of the team member.
- *Member Position listbox* – to select the position of the team member from the list previously defined.
- *Member Supervisor listbox* – to select the name or ID of the team member supervisor from the team members already defined. If left blank, the application will assign the team leader.
- *Team Leader checkbox* – if selected, indicates that the team member is the leader of the team.

In the *Team Work* panel, users define their experience with the team and the teamwork skills of the team member. The *Job Skills* panel creates a group of radial buttons, containing the possible skill levels for each job skill previously defined.

7.3 Enter the Job Data

A job is defined in the *Define Job Environment* form shown in Figure 13. Users need to enter:

1. Job Information
2. Job Tasks
3. Job Structure

The screenshot shows a software window titled "DEFINE ENVIRONMENT" with a sub-header "DEFINE JOB ENVIRONMENT". The main content area is titled "Job Definition" and is divided into several sections:

- Job Information:** Contains four input fields: "Job Identifier (*)" with value "Job1", "Date Line" with value "3000", "Release Time" with value "0", and "Permissible Delay (#)" with value "0.18". A note below states "(#) A decimal value between 0 and 1".
- Define Job Tasks (*):** Contains a "Current Job Size" field with value "10" and two buttons: "Add New Task" and "Modify Tasks".
- Define Job Structure (*):** Contains a "Current Number of Dependencies" field with value "13" and two buttons: "Add New Dependencies" and "Modify Dependencies".
- Navigation:** A row of three buttons: "Create New Job", "View Previous Job", and "View Next Job".
- Actions:** A row of two buttons: "Save Modifications to Current Job" and "Delete Current Job".
- Current Job:** A section titled "CURRENT JOB STATUS:" with two rows of labels and values: "Information Modify?" with a "NO" button, "Modifications Save?" with a "NO" button, "Current Job #" with a field containing "1", and "Total Jobs Saved" with a field containing "1".
- Footer:** A row of three buttons: "Save Environment", "Load Last Environment", and "Close Form".

A note at the bottom of the "Job Definition" section reads: "(*) Required fields. Enter Job ID first."

Figure 13: Define Job Environment Form

This form has the following controls:

- *Job ID textbox* – used to enter a name or ID for the job or mission.
- *Date Line textbox* – used to enter the date line of the job.
- *Release Time textbox* – used to enter the time the job will be available to the team for processing.
- *Permissible Delay textbox* – used to enter a percentage (enter as a decimal) of the date line or original estimated finish time that the job could be delayed.

- *Define Job Task Panel:*
 - *Job Size Indicator* – displays the current number of tasks defined for the job.
 - *Add Task button* – opens the Task Data form used to define a new task.
 - *Modify Task button* – opens the Task Data form to modify tasks already defined.
 - *Define Job Structure Panel* – has controls similar to the *Define Job Tasks Panel*, with the ability to enter the job dependencies. The *Add New Dependencies & Modify Dependencies* buttons opens the Task Dependencies form.
 - *Create New Job button* – this button creates a new job in the job array and clears the form.
 - *View Previous Job & View Next Job buttons* – used to navigate the job array and modify previously defined jobs.
 - *Save Modifications To Current Job* – this button saves the modifications made to the job currently displayed. No changes are saved if the user does not press this button.
 - *Delete Current Job* – this button deletes the job currently displayed.
 - *Current Job Panel* – this panel indicates to the user the number of total jobs in the job array and the position in the array of the current job. Also, has an indicator for the user to let them know if any modification has been made to the current job and if the changes have been made.
 - *Save Environment button* – this button saves the job array and creates a file with the job array information for later used during the simulation.
 - *Load Last Environment button* – this button loads the last job array saved in the file.
 - *Closed Form button* – closes the form without saving the information added or changes to the job array file to be used by the simulation.
- The *Task Data* form (Figure 14) has three main sections: Define Task Characteristics, Task Duration, and Skill Requirements. In addition has four control buttons: *Save & Return*, *Next Tasks*, *Closed Form* (without saving), *Delete Task*.

Figure 14: Task Data Form

The Define Task Characteristics contains the following controls:

- *Task ID textbox* – to enter the name or ID of the task.
- *Task Complexity listbox* – to pick the task complexity level.
- *Task Priority textbox* – to enter the priority of the task.
- *Assigned Agent listbox* – to pick the team member responsible to execute the task from a list.

The *Task Duration* panel contains three textboxes to enter the minimum, mean, and maximum duration parameters of the task. The *Skills Requirements Panel* contains an array of radio buttons to check all the skills required by this task. The skills were previously entered at the team definition.

The *Define Job Structure* form (Figure 15) is used to enter the dependency relations of a job. The *Dependency Tasks* panel contains two listbox controls to pick the predecessor and successors tasks in the dependency. The *Dependency Type* panel has the listbox controls to pick the dependency type and execution type. If the dependency type is “information,” the *Information Object* combo box is used to enter or pick the information object. The *Condition ID* combo box is used to enter or pick an identifier for the condition in case the execution type is “conditional” or “single conditional.” In the case of a “conditional” execution type, the *Condition ID* value should be the same for all

dependencies of the condition. Also, the values entered in the *Probability* textbox should add to one.

The screenshot shows a software window titled "DEFINE JOB STRUCTURE". It is divided into three main sections. The top section, "Dependency Tasks", contains two dropdown menus: "Predecessor Task" and "Successor Task", both with a list of tasks including "Consult", "Design", "Preparation", and "Purchase". The middle section, "Dependency Type", contains two dropdown menus: "Dependency Type" (with options: CONTROL SEQUENTIAL, INFORMATION SEQUENTIAL, RECIPROCAL) and "Execution Type" (with options: DETERMINISTIC, CONDITIONAL, SINGLE CONDITIONAL). To the right of these are three textboxes: "Information Object", "Condition ID", and "Probability" (containing the value "1.0"). The bottom section, "Merge Parameters", contains a checkbox labeled "Merge Dependency?" which is unchecked, and a dropdown menu for "Merge ID". Below the form are four buttons: "Save & Return", "Next Dependency", "Close Form", and "Delete".

Figure 15: Define Job Structure Form

The *Merge Parameters* panel is used in case the dependency ends in a merge with others. In those cases, the *Merge Dependency* checkbox should be checked, and the merge identifier picked or entered using the *Merge ID* combo box. The control buttons works in the same way as in the Task Data form.

8.0 SIMULATION PARAMETERS

The simulation parameters are not part of the scenario definition but could have an impact on the simulation realism and performance. Two types of parameters are used by the model: Random Behavior Parameters and Communication Parameters.

8.1 Random Behavior Parameters

The Random Behavior Parameters shape the behavior and decision-making of the team members. The behavior is defined by the use of a base value and a modifying value for each parameter that interacts with other characteristics of the team member and the team. The form to modify the *Random Behavior Parameters* is shown in Figure 16. These parameters are the following:

- Share Mental Model Factor – defines the probability that the information in each team member’s mental model agrees with the reality of the team and job scenario.
- Probability of Decision Allocated to Leader – defines the probability of a team member referring to the team leader for a decision. This parameter is influence by the Centralization level of the team.
- Probability of Unsolicited Communication – defines the likeliness of a team member sending information to another team member without that team member requesting.
- Probability of Task Error – defines the probability that a team member commits a mistake during the execution of a task that requires the use of rework. This probability is influenced by the skill fit of the team member with the task.
- Probability that the Transferred Information is Useful – defines the probability that the information sent by a team member will be useful to another agent.
- Time to Wait for Requirements – defines the time a team member will wait for the requirements of a task before requesting the information.

Figure 16: Random Behavior Parameters Form

8.2 Communication Parameters

The Communication Parameters control how the team members communicate with each other including the media definition. The form to define the *Communication Parameters* is shown in Figure 17. These parameters are:

- Probability of Using Synchronous Communication – describe the likeliness of a team member preferring the use of a synchronous media to send a message versus an asynchronous one.
- Media Preferences – define the media to use for each message classification. A “variable” value is recommended.
- Define Media – the “Add Or Modify Media” button opens the form shown in Figure 18, this form is used to define the properties of *Communication Media*.

Figure 17: Communication Parameters Definition Form

Figure 18: Form to Define the Communication Media

9.0 RUNNING THE SIMULATION

9.1 Simulation Options

The *Simulation Options* form (Figure 19) provides the following controls:

1. Time Units – currently inactive.
2. Number of Replications – controls the number of consecutive replications to be executed.
3. Random Generators Seed Controls – allows changes to the seeds used by the random numbers generators. The “Test Seed” buttons test the cycle length and the average of the seed. The seed is recommended if the cycle is greater than 10,000 numbers and the average is within 5 percent of 0.5.
4. Update Button – saves the changes made to the controls and closes the form.
5. Cancel Button – closes the form without saving the changes made to the controls.

Time Units		
<input type="radio"/>	Seconds	
<input checked="" type="radio"/>	Minutes	
<input type="radio"/>	Hours	
<input type="radio"/>	Days	

Replications	
Number of Replications	1
Replications Time Limit	3500

Random Generators Seeds		
Random Generator #1	17379195	Test Seed
Random Generator #2	735771	Test Seed
Random Generator #3	571	Test Seed
Random Generator #4	71717	Test Seed
Random Generator #5	773	Test Seed

Next Set of Seeds

UPDATE CANCEL

Figure 19: Simulation Options Form

9.2 Running the Simulation

After defining the team and the jobs, and setting up all the parameters, press the “Run Simulation” button in the *Main Menu* form (Figure 10) to run the scenario. While the simulation is running the *Replication Status Display* (Figure 20) will appear in the screen. This display shows the replication number and events indicating the start and end of each replication.

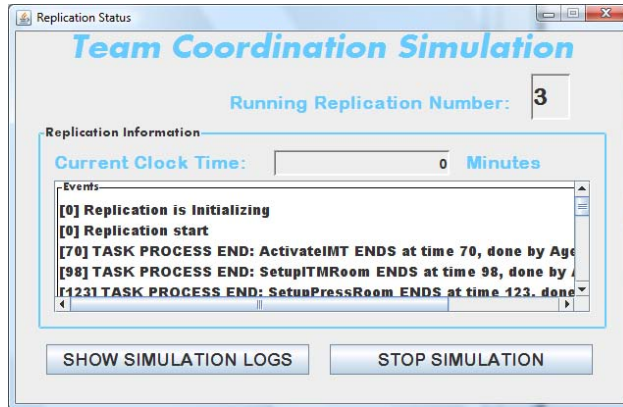


Figure 20: Replication Status Display

REFERENCES

- Baker, D. P., & Salas, E. (1997). Principles for measuring teamwork: A summary and look toward the future. In M. T. Brannick, E. Salas & C. Prince (Eds.), *Team performance assessment and measurement: Theory, methods, and applications* (pp. 331-355). Mahwah, NJ: Lawrence Erlbaum Associates.
- Burton, R. M., & Obel, B. (1995). The validity of computational models in organizational science: From model realism to purpose of the model. *Computational & Mathematical Organization Theory*, 1(1): 57-71.
- Carley, K. M. (1995). Computational and mathematical organization theory: Perspective and directions. *Computational & Mathematical Organization Theory*, 1(1): 39-56.
- Clymer, J. R., Cheng, D. J., & Hernandez, D. (1992). Induction of decision making rules for context sensitive systems. *Simulation*, 59(3): 198-206.
- Goldman, M., Stockbauer, J. W., & McAuliffe, T. G. (1977). Intergroup and intragroup competition and cooperation. *Journal of Experimental Social Psychology*, 13(1): 81-88.
- Jin, Y., & Levitt, R. E., Kunz, J. C., & Christiansen, T. R. (1995). The virtual design team: A computer simulation framework for studying organizational aspects of concurrent design. *Simulation*, 64(3): 160-173.
- Kim, J., & Burton, R.M. (2003). The effect of task uncertainty and decentralization on project team performance. *Computational & Mathematical Organization Theory*, 8(4): 365-384.
- Kunz, J. C., Christiansen, T. R., Cohen, G. P., Jin, Y., & Levitt, R.E. (1998). The virtual design team: A computational simulation model of project organizations. *Communications of the ACM*, 41(11): 84-92.
- Lyons, J., Ritter, J., Thomas, K., Militello, L., & Vincent, P. (2006). Collaborative logistics: Developing a framework to evaluate socio-technical issues in logistic-based networks. *International Symposium on Collaborative Technologies and Systems (CTS 2006)*.
- Marks, M. A. (2000). A critical analysis of computer simulations for conducting team research. *Small Group Research* 31(6): 653-675.
- McNeese, M. D. (1998). Teamwork, team performance & team interfaces: Historical precedence and application significance of the research at the USAF Fitts Human Engineering Division. *Proceedings of the IEEE International Symposium on Technology and Society* (pp 161-166). IEEE Society on Social Implications of Technology, South Bend, IN.

McQuay, W. K., The collaborative grid: Trends for next generation distributed collaborative environments. *Proceedings of the SPIE Enabling Technologies for Simulation Science Conference, Defense & Security Symposium*. Orlando, FL, 2004.

Paris, C. R., Salas, E., & Cannon-Bowers, J. A. (2000). Teamwork in multi-person systems: A review and analysis. *Ergonomics* 43(8): 1052-1075.

Salas, E., Guthrie, J. W., Wilson-Donnelly, K. A., Priest, H. A., & Burke, C. S. (2005). Modeling team performance: The basic ingredients and research needs. In W. B. Rouse & K. R. Boff (Eds.), *Organizational simulation* (pp. 185-228). New Jersey: John Wiley & Sons.

Stewart, G. L. (2006). A meta-analytic review of relationships between team design features and team performance. *Journal of Management* 32(1): 29-55.

Wang, W. P., Kleinman, D. L., & Luh, P.B. (2001). Modeling team coordination and decisions in a distributed dynamic environment. In G. M. Olson, T. W. Malone & J. B. Smith (Eds.), *Coordination theory and collaboration technology* (pp. 7-50). Mahwah, NJ: Lawrence Erlbaum Associates.

LIST OF ACRONYMS

MOUT	Military Operations in an Urban Environment
PERT	Program Evaluation and Review Technique
SMM	Shared Mental Model
TCM	Team Coordination Model