Modeling for Future Command and Control Architectures[1]

Holly A. H. Handley and Alexander H. Levis
C3I Center, MSN 4B5
George Mason University
Fairfax, VA 22030

ABSTRACT

Future command and control centers are being designed based to exploit the capabilities offered by information technologies; models of these proposed architectures are necessary to predict the performance of alternative designs. However, many of the performance metrics that these future centers will be evaluated on, such as speed of command and shared situational awareness, have not been included in previous command and control models. An enhanced command center model has been created that combines both a task process model and a decision maker model in order to provide the necessary degrees of freedom required to evaluate such performance metrics. The task process model was developed as part of a recent pre-experimental modeling iteration for a subject experiment and captures the stages of a task over its lifetime; the decision maker model is required in order to explicitly represent the interaction between decision makers as they process the task. This enhanced model will allow sophisticated modeling of interactions between decision makers, such as decision maker synchronization and information sharing. By combining the task process model with the decision maker model, surrogate measures of speed of command and situation awareness can be developed and used to evaluate the behavior and performance of command and control information and decision processes, essential to assess any future command and control architecture.

1.0 Introduction

As the military moves to redesign command and control architectures to incorporate information technologies, models are necessary to predict the behaviors and performance of the proposed command structures. However, many of the performance metrics, such as speed of command and shared situational awareness, have not been included in previous command and control models. Models of command and control architectures have been developed over the last eight years in order to examine the behavior and performance of experimental command centers performing missions in a laboratory environment [Handley et al., 1999]. Each model met the requirements of the experiment and was validated with post experimental data; however, each model was limited to the conditions of the hypotheses and the performance metrics of the organizations and missions being developed.

A task process model has been developed as part of a recent pre-experimental modeling iteration for a subject experiment [Handley and Levis, 2003]. While previous models had

[1] This work was supported by the Office of Naval Research under grant no. N00014-03-1-0033

focused on the decision maker process and metrics of the decision maker workload, the task process model captures the stages of a task over its lifetime, including information needs and decision maker activity required at each stage. By using the task process model, metrics for decision maker participation and information requirements regarding specific tasks can be elicited. The task process model was found to correlate well with tasks that required a single decision maker completing a task with a single resource. The model did not explain, however, why some tasks stop mid process and resume at a later time and it does not represent decision maker synchronization well, i.e., two or more decision makers supplying resources within a specified time window to complete a task. Both of these conditions require a decision maker model to work in conjunction with the task process model, in order to explicitly represent the interaction between the task and the decision maker.

In order to reconcile the task process model with an existing decision maker model, the five-stage interacting decision maker model [Levis, 1995], the empirical data collected from the subject experiment used to validate the task process model were examined. From the data, the task stages that required different decision makers were identified, along with delays in the task stages due to decision maker synchronization and interruptions in task processing due to engaged decision makers. The empirical data offered insights on how decision makers coordinated on complex tasks. An enhanced model was then created that combined the task process model and the decision maker model. The enhanced model will allow more sophisticated modeling of interactions between decision makers, such as decision maker synchronization and information sharing. By combining the task process model with the decision maker model, surrogate measures for speed of command and situation awareness can be developed and used to evaluate the behavior and performance of command and control information and decision processes, essential to assess any future command and control architecture.

The remainder of the paper is organized as follows: the next section describes the task process model while section 3.0 identifies its limitations. Section 4.0 describes the five stage decision maker model and section 5.0 describes the enhanced model that results from joining these two component models. Section 6.0 describes the performance measures used with this model, specifically speed of command and shared situational awareness; section 7.0 concludes the paper.

2.0 Description of the Task Process Model

The task process model was designed in conjunction with a subject experiment examining the relationship between different command and control architectures and alternative scenarios [Diedrich et al., 2002]. The task process model emulates the series of stages a task follows over its lifetime; each task that appears in the scenario is represented and evaluated separately. A task is an activity that entails the use of relevant resources and is carried out by an individual decision maker or a group of decision makers to accomplish the mission objectives or in defense of own assets. The task stages are based on the simulator used in the subject experiment, the Dynamic Distributed Decision-Making (DDD) simulator. The model was developed and validated using trial

experimental data; see Handley & Levis [2003] for a complete description of the model development.

The task process model is shown in Figure 1. The first stage, *Appear*, occurs when a task (in this case a threat) is first present in the environment. This is controlled by the input scenario which specifies the time that each task appears. As soon as the task is noticed, either by a decision maker or a sensor, it is *Detected* and a decision maker initiates its processing. The task is then *Identified*; this indicates the decision maker knows what type of task it is and what type of resource (in this case a weapon) can be used to process the task. When the weapon is launched and travels to collide with the task, the task is defined as *Attacked*. When the resource has succeeded in completing the task (the weapon has destroyed the threat), the task is considered *Destroyed*. Lastly, the task *Disappears* from the simulator screen.

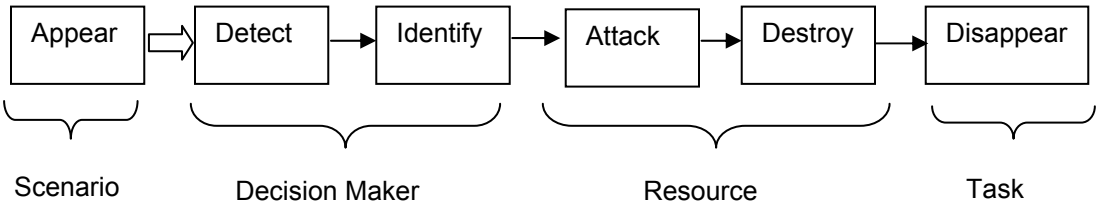

Figure 1: Task Process Model

The output of the task process model is a task completion time for every task in the scenario. Each stage of the sequential model has a delay determined by the attributes of the task, decision maker, or resource. Once the task enters the detect stage, it proceeds through the process uninterrupted, accumulating the delays at each stage until completion. The finish time of the task is the sum of the delays of each stage in the process; the task delay is the completion time minus the time the task first appeared, representing the actual processing time of the task:

$$t_{finish} = t_{appear} + t_{detect} + t_{identify} + t_{attack} + t_{destroy} + t_{disappear.} \quad [1]$$

$$t_{delay} = t_{finish} - t_{appear} \quad [2]$$

The time the task appears, t_{appear}, is predetermined by the scenario; the scenario is a list of all input tasks and the time they enter or appear in the model. The detect delay, t_{detect}, of each task is variable depending on the activity of the decision maker. If the decision maker responsible for the task is not currently acting on another task, the current task will be detected immediately. If, however, he is engaged with another task, the current task will wait until he is unoccupied; this variability is represented by the larger arrow in Figure 1. The delay associated with identifying the task, $t_{identify}$, represents the processing time by the decision maker. At the end of this stage the decision maker has identified the type of threat the task represents and the appropriate type of resource to use against it. This stage has a fixed delay to represent the decision maker's processing time. This value

was determined empirically by comparing model simulation data at increasing levels of decision maker delay to trial experimental output data. There is also a workload limit of one task imposed on the model at this stage. In the *Attack* and *Destroy* stages the parameters of the resource chosen by the decision maker to counteract the task provide the delay times; t_{attack} is the launch delay of the resource and $t_{destroy}$ is the travel time of the resource to the location of the task. The delay of the *Disappear* stage, $t_{disappear}$, represents the delay between the time the threat is attacked and the time it disappears from the display; this value is specific to each task class.

The model was implemented using Colored Petri nets, a graphical modeling language and a powerful modeling tool used to expose critical time dependencies, task concurrencies and behavior that is event driven. The model was implemented in Design/CPN and simulated under different conditions as determined by the experimental design. While the DDD simulator creates the environment for the subject experiments, it also captures the subject's actions and task data throughout the course of the experimental scenario. This information is made available after the experiment in log files, which can be sorted by decision maker, resource, or task identifiers to find timing information. The model was validated after the subject experiment by comparing the timing of the task stages recorded in the log files with the timing of the task stages used in the simulation model.

Certain delays in the process are fixed based on the task type or the resource chosen while other delays are variable depending on the activity level of the responsible decision maker. The task pattern in the scenario elicits different decision maker activity levels depending on the architecture; the architecture determines what resources a decision maker controls and what types or locations of tasks each is responsible for. This model was used to predict congruence between architectures and scenarios. Scenarios varied the arrival time, type, and location of tasks, which in turn changed the loading on decision makers and affected his choice of resource; congruence was evaluated as the ability of the different architectures to process the scenario tasks in a timely manner. The task process model allows the evaluation of individual task delays; looking at a single task process allows the correlation of the task delay, resource used and decision maker engaged with the experimental data Looking across multiple task processes can be used to identify concurrency between tasks, decision maker workload at a particular time, and platform activity across time.

3.0 Limitations of the Single Task Model

Empirical data from the subject experiment [Handley and Levis, 2003] was used to validate the task process model. The performance of the model was validated by comparing the task completion times of the model to the experimental results. The average correlation between the model data and the experimental data was 0.86, with an average of 58 tasks per scenario (see Appendix A). While the final output of the task stages correlated well, there was a discrepancy between the modeled tasks and some of the experimental results. Examination of the time data of the experimental task stages indicated that often tasks were interrupted in the middle of the process and then were resumed later on. In order to verify the sequence of stages that composed the task

process, experimental data from two task classes (threats) were examined in detail: enemy patrol boats and enemy air attacks. Both of these tasks required one decision maker and one resource to complete. Extraction of the empirical stage delays showed that the individual tasks fell into two categories: those that had an interruption, or a large time delay, in their task process, and those that did not. Note that the DDD simulator is a real time simulation, i.e., one second of real time represents one second of simulation time. Examples of the simulation time at each task stage are shown in Figure 2.

	Arrive	Detect	Identify	Select	Attack	Destroy
Patrol Boat #218						
Team MA	370	371	372	430	434	439
Team MC - Interruption	370	371	372	584	588	593
Air Attack #406						
Team SA	1251	1254	1322	1340	1342	1347
Team SE - Interruption	1251	1254	1322	1380	1389	1394

Figure 2: Example Tasks with Task Process Interruptions

The empirical data suggests that in some instances there is a break in the processing between the *Identify* and *Attack* stages. The stage *Select* was introduced in the log files to indicate the continuation of a task after an interruption. This break represents the decision maker disengaging from the current task to attend to another, higher priority task, before returning to the original task. This requires including another stage in the task process model, the *Select* stage, which represents another variable delay depending on the activity level of the decision maker. This also implies the need for a coupling of a decision maker model with the task process model to allow for variations in the task processing due to the activity level (workload) of the decision maker.

Many of the tasks in the scenario required the interaction of one or more decision makers to combine their resource in order to execute the task. These tasks were not included in the model simulation, but were present in the experiment and in the empirical data collected for examination. An example of the interaction of the decision makers synchronizing their resources is shown in Figure 3:

Stage	Time	DM	Resource
04-Select	1726	2	
04-Select	1756	3	
11-Attack	1759	2	SOF-500
12-Assist	1761	3	SOF-501
18-Destroy	1789	2	

Figure 3: Decision Maker Synchronization on Task 28, Team D Run S2

These synchronized tasks could not be included in the original task process model design; however, they can be addressed if the decision maker model is included explicitly.

4.0 A Five Stage Decision Maker Model

In order to study the behavior of an organization, it is necessary to have a model of its components, namely the individual decision makers. March and Simon [1958] hypothesized that decision makers follow a two step process: first determining the situation and then determining a response. This led to a two stage decision maker model by Wohl [1979] which was expanded to four stages by Boettcher and Levis [1982] in order to accommodate interactions between decision makers. Remy and Levis [1986] formalized these interactions. Levis [1992] presented a model of a five stage interacting decision maker that subsumed the previous models. This model presupposes that the decision makers are executing well-defined tasks for which they have been trained and that there is a limit to the amount of processing a decision maker can perform [Boettcher and Levis, 1982] in accordance with the bounded rationality constraint [March, 1978].

The five stage decision maker model is shown in Figure 4. The decision maker receives a signal, x, from the external environment or from another decision maker. The situation assessment stage (SA) represents the processing of the incoming signal to obtain the assessed situation, z, which may be shared with other decision makers. The decision maker can also receive a signal z' from another decision maker; z' and z are then fused together in the information fusion (IF) stage to produce z''. The fused information is then processed at the task processing (TP) stage to produce v. A command or control information from another decision maker is received as v'. The command interpretation (CI) stage then combines v and v' to produce the variable w, which is input to the response selection (RS) stage. The RS stage then produces the output y to the environment, and/or the output y' to other decision makers.

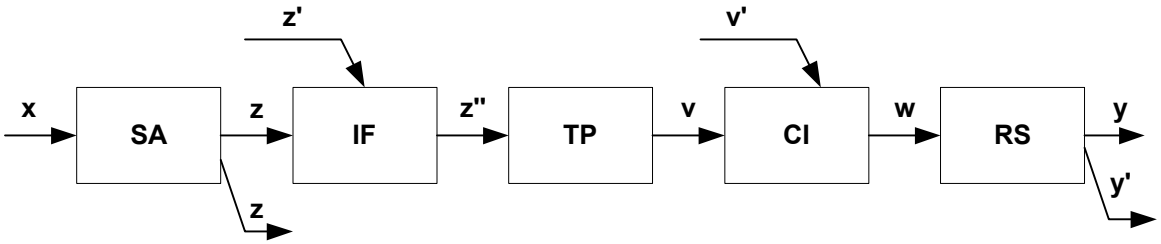

Figure 4: Five Stage Interacting Decision Maker

The model depicts explicitly the stages at which a decision maker can interact with other decision makers or the environment. A decision maker can receive inputs from the external environment only at the SA stage. However, this input x can also be from another decision maker (the y' output) from within the organization. A decision maker can share his assessed input through the z output at this stage. The z' input to the IF stage is used when the decision maker is receiving a second data input. This input must be

generated from within the organization and can be the output of another decision maker's SA or RS stage. The fused information from the IF stage, z", is the input to the TP stage. The decision maker's function is performed at this stage and results in the output v. In the CI stage, the decision maker can receive control information as the input v'. This is also internally generated and must originate from another decision maker's RS stage. In the RS stage, an output is produced; y is the output to the environment and y' is the output to another decision maker. Thus the interactions between two decision makers are limited by the constraints enumerated above: the output from the SA stage, z, can only be an input to another decision-maker's IF stage as z', and an internal output from the RS stage, y', can only be input to another decision maker's SA stage as x, IF stage as z', or CI stage as v'.

A decision maker need not exercise all five stages when performing a task. Depending on the inputs and outputs required, a decision maker can instantiate different subsets of the five stage model.

5.0 Enhanced Task Process Model

The two limitations identified in the current task process model are the inability to allow a decision maker to disengage from a task in order to initiate the processing of another task and the inability to represent complex tasks, i.e., tasks requiring multiple decision makers to synchronize resources to accomplish a task. Both of these limitations require a coupling of the task process model with the five stage interacting decision maker model.

5.1 Task Interruption

Currently in the single task process model, the delays of the *Detect* and *Identify* stages are due to the decision maker; the decision maker is implicitly associated with these task stages. This relationship can be made explicit by associating the *Detect-Identify* stages of the task process model with the Situation Assessment (SA) – Response Selection (RS) stages of the decision maker model. The task process model is now constrained by the decision maker model during these stages. An additional task stage was identified in the empirical data: the *Select* stage preceded the *Attack* stage and was used to indicate that a decision maker had continued processing the interrupted task. The decision maker SA-RS stages can again be associated with the task process *Select – Attack* stages. This coupling of the task process model with the decision maker model is shown in Figure 5.

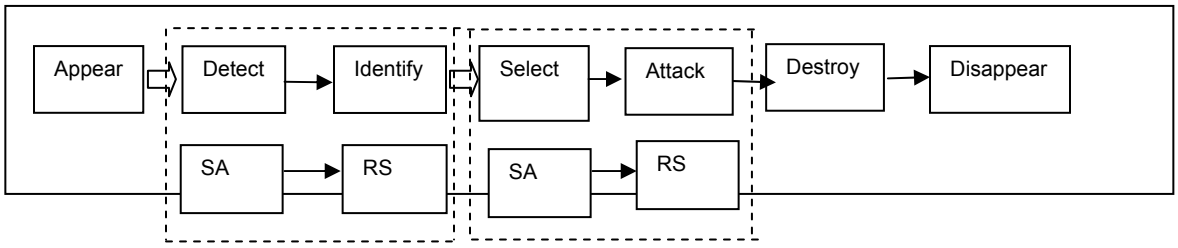

Figure 5: Single Task Model Coupled with Decision Maker Model

This coupling allows a second variability in the task processing: the original interruption in processing occurs between the *Appear* and *Detect* stages, which represents the variable delay due to an engaged decision maker, and an additional interruption between the *Identify* and *Select* stages, when a decision maker may disengage from the task in order to attend to another task. The *Select* stage delay is also a variable delay depending on the activity of the decision maker. The linear delay equations are modified by adding the additional variable delay term t_{select}:

$$t_{finish}` = t_{appear} + t_{detect} + t_{identify} + t_{select} + t_{attack} + t_{destroy} + t_{disappear.} \quad [3]$$

$$t_{delay}` = t_{finish}` - t_{appear} \quad [4]$$

5.2 Decision Maker Synchronization

The task process model is limited in that it can currently process only tasks requiring a single decision maker with a single resource. While these tasks did account for the majority of the tasks in the experimental scenarios, these were mostly tasks that defended own assets. The tasks that defined the objective of the scenario mission were the complex tasks that required two or three decision makers to synchronize their resources to accomplish the task within a defined time window; if any one decision maker applied his resource outside of that time window the task would fail.

In order to represent the synchronization of decision makers in the model, the different roles a decision maker assumes in processing different types of tasks must be defined in terms of the five stage model. Three decision maker roles have previously been identified [Levis et al., 1998]. The Independent role is defined as a decision maker acting on a task that he can then execute without interacting with other decision makers; this is the role of the single task in the current task process model. The Leader role is defined when a decision maker has to execute a task by interacting with other decision makers, however this decision maker is the initiator and sends synchronization messages to the other decision makers. The Follower role is defined for decision makers who must provide resources to execute a task with other decision makers, but it is another decision maker that sends the synchronization information.

The Independent role is used in the single task process model; a single decision maker with a single resource processes the task. Figure 4 and equations [3] and [4] describe this model. The Leader and Follower roles are bound together by the interactions required to synchronize their efforts. In all complex tasks, the commander's intent identifies a specific decision maker as the leader for that task. This decision maker will initiate the task and interact with the other decision makers as necessary to complete the task. These interactions are shown in Figure 6.

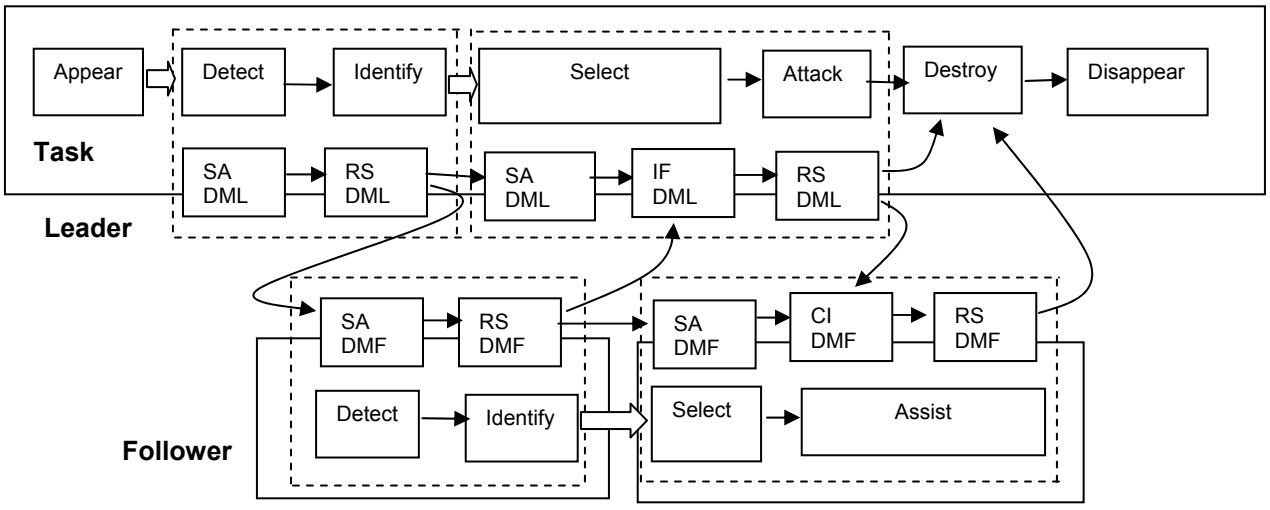

Figure 6: Leader and Follower Role Interactions

The Leader (DML) will first *Detect* and *Identify* the task; he identifies the additional resources and their responsible decision makers he needs for the complex task. The Leader is modeled with SA-RS stages, similar to the Independent role with the addition of the y` output. The y` output is used to alert the Follower decision maker(s) to *Detect* and *Identify* the task in preparation for a synchronized attack. The Follower (DMF) is modeled with the same SA-RS combination and the y` output is used at the Leader's *Select* stage to indicate the readiness of the Follower decision makers; the Leader's *Select* stage is still variable dependent on the other tasks he is engaged with. The Leader will then begin the attack by launching his resource and signaling the other decision makers to synchronize their resource launch. The Leader's *Select* stage is modeled by a SA stage to indicate the task has been selected and an IF stage to include the "ready" signal from the Followers. The Attack stage is modeled as a RS stage, which indicates the launch of the resource and again a y` signal is sent to interact with the Follower. The Follower's *Select* stage also has a variable delay depending on his task priorities; the SA stage represents his *Select* stage. The Follower's task process stage is indicated as "*Assist*" in the DDD simulator; the *Assist* stage is modeled as a CI stage that waits for the synchronization signal from the Leader, and then a RS stage where the resource is launched.

The completion time of the task must be represented as a combination of delays from both the Leader and the Follower decision makers. In terms of the delays incurred by the Leader, the task completion time is:

$$t_{finish}`` = t_{appear} + t_{detectL} + t_{identifyL} + t_{selectL} + t_{IFL} + t_{attackL} + t_{destroy} + t_{disappear.} \quad [5]$$

However, t_{IFL}, which represents the delay associated with waiting for the "ready" response from the Follower decision maker is not a valid entry; it cannot be traced as a task stage in the DDD simulator. This delay can be represented, however, as the *Detect* and *Identify* stages of the Follower decision maker, $t_{detectF} + t_{identifyF}$; but these delays may

be occurring concurrently with the *Select* delay of the Leader. The delay can then be correctly represented as a maximum of the pair:

$$t_{finish}`` = t_{appear} + t_{detectL} + t_{identifyL} + t_{selectL} + \max [0, (t_{detectF} + t_{identifyF}) - t_{selectL}] + t_{attackL} + t_{destroy} + t_{disappear.} \quad [6]$$

Likewise, the *Destroy* stage of the task is dependent on all synchronized resources arriving with the window of attack determined by the task class. In this case:

$$t_{assistF} - t_{attackL} < \Delta t_{attack} \quad [7]$$

where Δt_{attack} is the pre-determined time completion window of the task; if this condition is not met the threat is not destroyed. For completeness:

$$t_{delay}`` = t_{finish}`` - t_{appear.} \quad [8]$$

The enhanced task process model was implemented as a Colored Petri net and resimulated with the experimental scenario. The model output was again correlated with the experimental data; the average correlation over 12 teams was 0.80 with an average of 71 correlated tasks (see Appendix A). In order to add complex tasks to the model, the scenario was modified to accept the mission tasks. The mission tasks are complex tasks that must be completed in precedence order and require multiple resources; in some architectures one decision maker may own the complete set of resources, in other architectures he must coordinate with other decision makers to complete the set of resources required. While the existing scenario, the independent tasks, was an input list of tasks and their (fixed) arrival time, now the mission tasks are triggered by the completion of other tasks and as such have no fixed arrival time. This makes the correlation more complex as not only is the completion time variable, but also the *actual* appear time.

The delay times of this model are dependent on the interaction of the architecture and the scenario, and on the interaction of the Leader and Follower decision makers in complex tasks. The decision maker SA delay times (t_{detect}, t_{select}) represent the delay for the decision maker to commence work on the task, either initially or after an interruption. This delay depends on what other tasks the decision maker is engaged on and task priorities. The IF and CI stages are junctions where the decision makers exchange information; task processing suspends until all information is exchanged. It would be difficult to determine these compound delays without executing the model in simulation mode.

6.0 Performance Measures

6.1 Speed of Command

Speed of Command is defined as the time from when a threat is detected until it is engaged. A surrogate measure for the speed of command in the enhanced task process model is the task delay, i.e. the difference between the completion time of the task and

the time the task appeared; the time from the *Detect* stage to the *Disappear* stage in the model. This is equation [4] for a single task and equation [8] for a synchronized task. Tasks can be evaluated individually using this metric as in Figure 6, or the accumulated task delay over the course of the scenario can be compared across architectures, as in Figure 7.

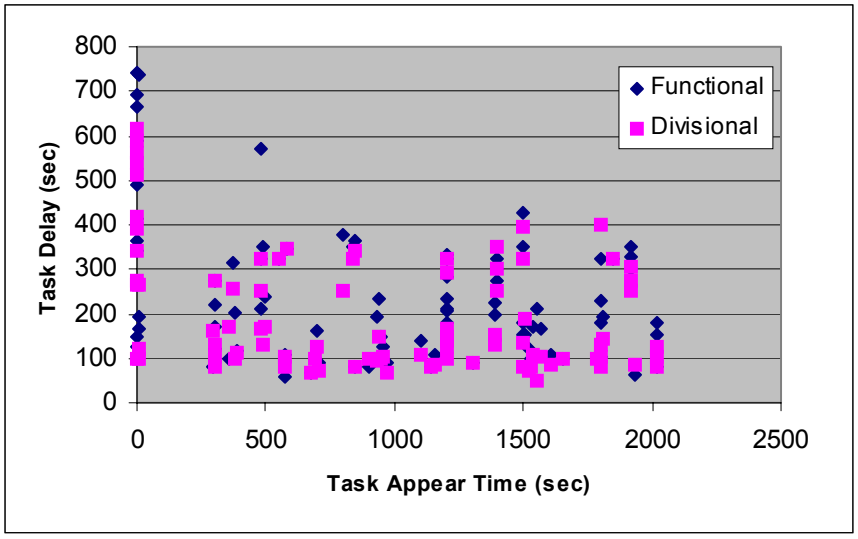

Figure 6: Speed of Command as Task Delay for Individual Tasks

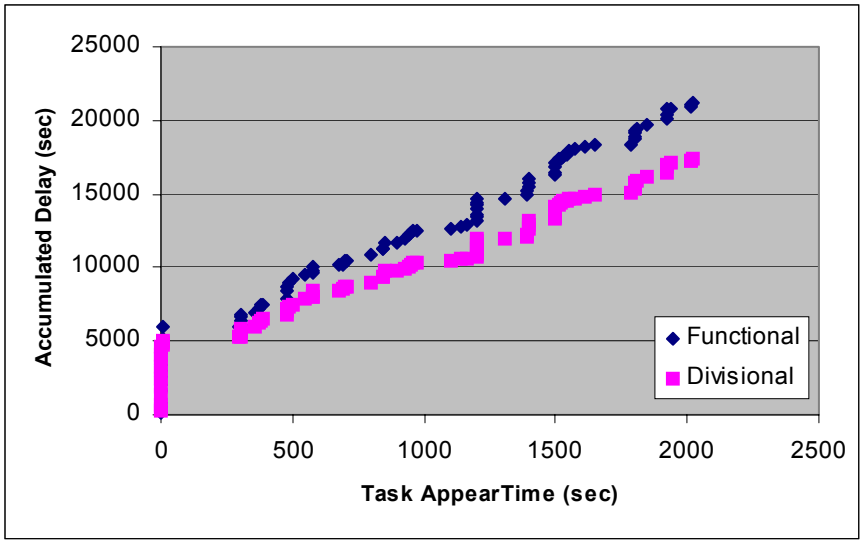

Figure 7: Speed of Command as Accumulated Task Delay for Scenario

The data in the graphs show the simulation results for two different architectures, termed Functional and Divisional; in both cases the same scenario was used. The delay of each task versus the time it appears for each architecture is shown in Figure 6. While this graph shows the differences in delay for each individual task, it does not give a good

indication of how each architecture is performing with regards to Speed of Command. A better indicator of the performance is the accumulated delay of individual tasks as the scenario progresses over time as shown in Figure 7; the Divisional architecture shows an improvement of 17.8% in accumulated task delay, or Speed of Command, over the Functional architecture. The enhanced model includes the variability of the decision maker's attention to the task, not only the initial delay in the *Detect* stage, but also delays that may occur due to interruptions in the task processing at the *Select* stage. These results concur with the experimental output; for this scenario, termed the "M" scenario, the Divisional organization outperformed the Functional organization.

6.2 Shared Situational Awareness

Shared Situational Awareness is the ability of a team of decision makers to perceive and understand a tactical picture that is complete and consistent across the team. In the single task process model there was no mechanism for decision makers to interact on a task; therefore there was no metric to gauge the situational awareness of multiple decision makers on the same task. The enhanced model specifically allows decision makers to synchronize their efforts to complete a task, which allows the opportunity to propose a metric to observe shared situation awareness.

On complex tasks that require multiple decision makers, a time window exists for each task in which all required resources must be fired. This allotted time can be described as a window of attack whose parameters are determined a priori by the requirements of the task; different task types may have different windows of attack. Two quantities are needed to specify the window of attack: the lower and the upper bounds of the time interval, t_s and t_f, respectively, or one of the bounds and the length of the interval, e.g. t_s and Δ t [Cothier and Levis, 1986]. The lower bound of the window is the time the first resource attacks the task and the length of the window is the predetermined time window of attack. In order for the attack to be successful, the time the final required resource attacks the task must be within the window's bounds:

$$t_f < t_s + \Delta t \qquad [9]$$

This window of attack (which is equivalent to [7]) can represents a surrogate measure for Shared Situation Awareness for the decision makers participating in the task. For the task to succeed the team of decision makers all need to apply the correct resources to the correct task within a finite period of time indicating a consistent and complete tactical picture. As the number of decision makers who participate in an attack increase, this metric becomes more meaningful.

The enhanced task process model can provide insights on Shared Situational Awareness based on the interactions between the Leader and Follower decision maker on synchronized tasks. There are three interaction points between the two roles: the output of the Leader's *Identify* (RS) stage to the Follower's *Detect* (SA) stage, the output of the Follower's *Identify* (RS) stage to the Leader's *Select* (IF) stage, and the output of the Leader's *Attack* (RS) stage to the Follower's *Assist* (CI) stage. While the first two will

affect the overall delay of the task, the last interaction affects the final synchronization of the launch of resources. The critical point is the variable delay of the Follower's *Select* stage; if he delays too long resuming processing of the task, he will miss the window of attack initiated by the Leader's resource launch. This variable delay is a function of the architecture interacting with the scenario and can be used to compare across architecture scenario pairs; an example is shown in Figures 8 and 9.

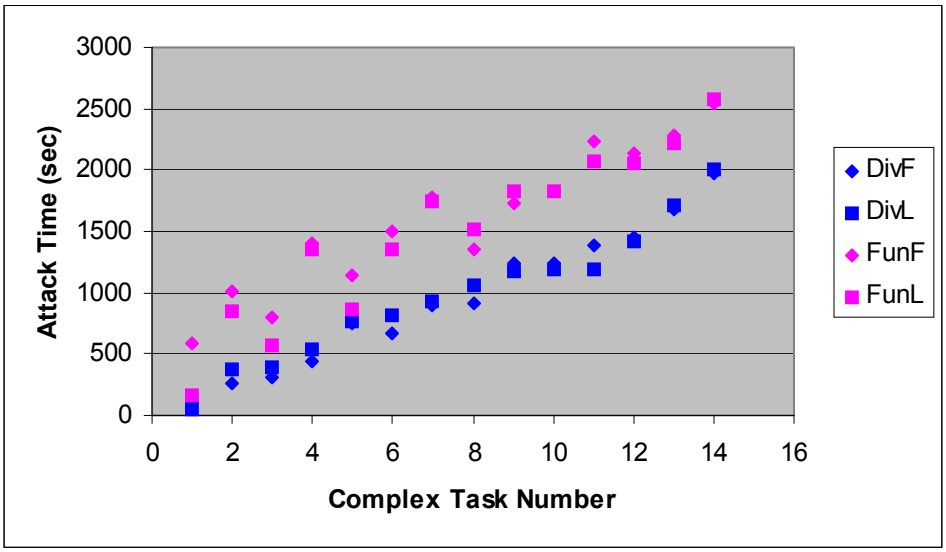

Figure 8: Shared Situational Awareness as Leader-Follower Attack Times

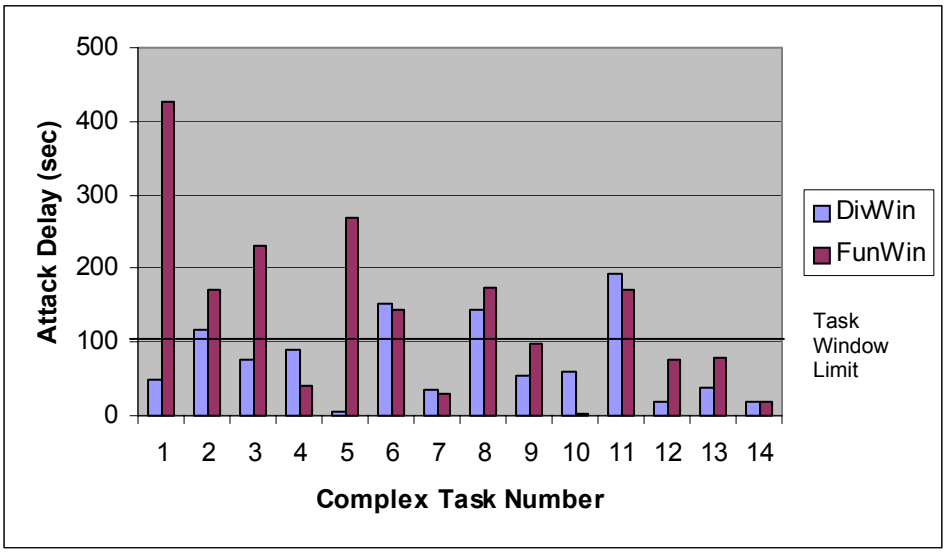

Figure 9: Shared Situational Awareness as Attack Delay

Figure 8 shows both the Leader and Follower attack times for a set of 14 complex tasks, similar to those used in the "M" scenario above but in a separate, investigational scenario

where the tempo of operations has been increased, completed by both Divisional (Div) and Functional (Fun) architectures. This can be used to evaluate how situational awareness varies over the course of this investigational scenario. In this case situational awareness seems to improve over the course of the scenario for both architectures. Figure 9 makes explicit the attack delay ($t_f - t_s$) versus the task window of attack (Δt_{attack}). The Functional architecture has seven tasks that miss the window and the Divisional has four. For a consistent Attack Window of 100 simulation seconds, the average window for the Divisional architecture is 73.8, while for the Functional architecture it is 136.4. In this case the Divisional architecture has a 46% improvement in Shared Situational Awareness. This metric was difficult to obtain from the subject experimental data, as many of the complex tasks were not attempted, and so no window data comparison was made.

7.0 Conclusion

The task process model was designed in conjunction with a subject experiment examining the relationship between different command and control architectures and alternative scenarios; the task process model emulates the series of stages a task follows over its lifetime. Limitations to the model were identified, specifically the inability to allow a decision maker to disengage from a task in order to initiate the processing of another task and the inability to represent complex tasks, i.e., tasks requiring multiple decision makers to synchronize resources to accomplish a task. Both of these limitations require a coupling of the task process model with a decision maker model. The five stage interacting decision maker model depicts the stages at which a decision maker can interact with other decision makers or the environment, i.e. the task process. The relationship between the models was made explicit by associating the *Detect* - *Identify* and *Select* - *Attack* stages of the task process model with the Situation Assessment (SA)-Response Selection (RS) stages of the decision maker model. The task process model is now constrained by the decision maker model during these stages.

The delay times of the enhanced task process model are dependent on the interaction of the architecture and the scenario, and on the interaction of the decision makers executing complex tasks. These can be used to define surrogate measures for Speed of Command and Shared Situational Awareness; the accumulated delay time was used to compare the Speed of Command across architectures and the task window limit was used to evaluate Shared Situational Awareness. By using empirical data collected after a subject experiment, enhancements made to an existing model have resulted in a model that is more realistic and versatile in evaluating command and control architectures operating under different scenarios. The performance measures reflect the completion times of time critical tasks; including the variability of the decision maker's attention to the task, not only the initial delay in the *Detect* stage, but also delays that may occur due to interruptions in the task processing at the *Select* stage and the interaction of decision makers due to the synchronization of the launch of resources. The model will be a valuable tool for evaluating proposed future command and control centers.

References

[Boettcher and Levis, 1982] Kevin L. Boettcher and Alexander H. Levis, "Modeling the Interacting Decisionmaker with Bounded Rationality," *IEEE Transactions on Systems, Man and Cybernetics*, Vol. SMC 12, No. 3, pp. 334-344.

[Boettcher and Levis, 1983] Kevin L. Boettcher and Alexander H. Levis, "Decisionmaking Organizations with Acyclical Information Structures," *IEEE Transactions on Systems, Man and Cybernetics*, Vol. SMC 13, No. 3, pp. 384-391.

[Cothier and Levis, 1986] Phillippe H. Cothier and Alexander H. Levis, 'Timeliness and Measures of Effectiveness in Command and Control," *IEEE Transactions on Systems, Man, and Cybernetics,* Vol. SMC 16, No. 6, pp. 844-853.

[Diedrich et al., 2002] F.J. Diedrich, S.P Hocevar, E.E. Entin, S.G. Hutchins, W.G. Kemple, and D.L. Kleinman, "Adaptive Architectures for Command and Control: Toward An Empirical Evaluation of Organizational Congruence and Adaptation," *Proceedings of the Command and Control Research and Technology Symposium*, Monterey, CA.

[Handley et al., 1999] H. A. H. Handley, Z. R. Zaidi, and A. H. Levis, "The Use of Simulation Models in Model Driven Experiments," *Systems Engineering*, John Wiley & Sons, Inc, 2:108-128.

[Handley and Levis, 2003] H. A. H. Handley and A. H. Levis, "Organizational Architectures and Mission Requirements: A Model to Determine Congruence," *Systems Engineering*, John Wiley & Sons, Inc, Vol. 6, No. 3.

[Handley, 2003] Holly A. H. Handley, "Dynamic Modeling of the Strategic Studies Group Architectures," Memorandum, January 14, 2003.

[Hiniker, 2002] Paul J. Hiniker, "A Model of Command and Control Processes for JWARS." 7th International Command and Control Research and Technology Symposium, September 16-20, 2002, Quebec City, Canada.

[Levis, 1992] Alexander H. Levis, "A Colored Petri net Model of Intelligent Nodes," *Robotics and Flexible Manufacturing Systems*, J.C. Gentina and S.G. Tzafestas, Eds., Elsevier Science Publishers, The Netherlands.

[Levis, 1995] A.H. Levis, "Human Interaction with Decision Aids: A Mathematical Approach," in Human/Technology Interaction in Complex Systems, Vol.7, W.B. Rouse, Ed., JAI Press, 1995

[Levis et al., 1998] Alexander H. Levis, Holly Handley, Zainab R. Zaidi, "Five Stage Decision Maker Model," Memorandum, August 28, 1998.

[March, 1978] J.G. March, "Bounded Rationality, Ambiguity, and the Engineering of Choice," Bell J. Economics, Vol. 9, pp. 587-608, 1978.

[March and Simon, 1958] J.G. March and H. A. Simon, *Organizations*, John Wiley and Sons, NY.

[Remy and Levis, 1986] P.A. Remy and A. H. Levis, "On the Generation of Organizational Architectures using Petri nets," in *Advances in Petri Nets*, G. Rozenberg, Editor, Springer-Verlag, Berlin.

[Wohl, 1981] J.G. Wohl, "Force Management Decision Requirements for Air Force Tactical Command and Control," *IEEE Transactions on Systems, Man, and Cybernetics*, Vol. SMC-11, No.9.

Appendix A: Correlation Data and Statistics

For each trial in the subject experiment, indicated by *Team*, a correlation was performed between the experimental completion time data and the simulated model completion time data. This value, indicated by *Correlation*, was obtained by correlating the tasks that were completed by both the team and the simulation; this number varies by team and is indicated by *Number of Tasks*. For each correlation a significance test was performed by using the F statistic; the results of the test are shown in the columns *F Value* and *F Significance*. In all cases, the null hypothesis of no predictive value can be rejected.

Table A.1: Original Task Process Model Correlation Statistics Between Experimental and Simulated Output

Team	Correlation	Number of Tasks	F Value	F Significance
FSf2W	.77	62	89.60	1.64E-13
FSf2D	.83	59	124.14	6.16E-16
FSf2B	.75	60	76.02	3.87E-12
FMd2W	.81	58	107.95	1.12E-14
FMd2D	.88	48	165.78	7.37E-17
FMd2B	.92	57	315.21	1.95E-24
DSf2E	.90	52	217.20	7.88E-20
DSf2C	.84	54	127.74	1.28E-15
DSf2A	.88	57	195.66	9.26E-20
DMd2E	.84	61	141.15	2.74E-17
DMd2C	.95	62	602.50	5.50E-33
DMd2A	.97	60	822.47	5.96E-36

Table A.2: Enhanced Task Process Model Correlation Statistics Between Experimental and Simulated Output

Team	Correlation	Number of Tasks	F Value	F Significance
FSf2W	.71	70	70.76	3.91E-12
FSf2D	.75	74	90.29	2.44E-14
FSf2B	.68	74	60.53	3.94E-11
FMd2W	.79	75	119.56	4.94E-17
FMd2D	.87	63	183.15	4.97E-20
FMd2B	.92	72	405.64	7.67E-31
DSf2E	.82	63	124.71	2.21E-16
DSf2C	.76	66	87.10	1.49E-13
DSf2A	.67	70	55.39	2.25E-10
DMd2E	.79	71	114.08	2.88E-16
DMd2C	.88	78	253.89	6.14E-26
DMd2A	.90	77	334.47	2.30E-29