12th ICCRTS

Adapting C2 to the 21st Century

Use of a Systems Information Broker to Aide in the Dynamic Interfacing of C2 Nodes

Networks and Networking
C2 Metrics and Assessment
C2 Technologies and Systems

Dagohoy H. Anunciado [STUDENT]
Dagohoy H. Anunciado
SPAWARSYSCEN 24226, San Diego, CA & Naval Postgraduate School, Monterey CA
53605 Hull St Bldg A-33
San Diego CA 92152-5001
619-553-5604/619-553-6025 FAX
doug.anunciado@navy.mil

Abstract

Missions assigned to military forces will change as world events occur. Recent events like the Indian Ocean Tsunami and Hurricane Katrina in the United States required a massive humanitarian effort that included military forces. Information about the event needed gathering, distributing, and analyzing to determine how best to use resources to help the people in the devastation. Once observers gather information, establishing communications is needed before information can be distributed. Command and Control (C2) node functions perform one or all of the tasks of information gathering, distribution, analysis, decision making, and distribution of decisions. C2 nodes in these situations will be mobile or fixed and will come and go as a mission unfolds. Interfacing of C2 nodes may be hampered when the interface mechanisms are not worked out before an event and would take time to manually work out, which delays rescue and relief efforts. This research defines a framework and methodology for dynamically interfacing C2 nodes to a C2 enterprise to accomplish large missions such as responding to operations other than war (OOTW), e.g., nature and man-made disasters, peacekeeping, and counter drug operations.[1] Regional conflicts and general war are other situations requiring C2 enterprise accomplish a large mission.

1. Introduction (Motivation)

Missions assigned to military forces will change as world events occur. Events like the Indian Ocean Tsunami[2] devastated coastal regions of Indonesia, Sri Lanka, India, and Thailand, and also affected Somalia, Myanmar, the Maldives, Malaysia, Tanzania, Seychelles, Bangladesh, South Africa, Yemen, Kenya, and Madagascar. Hurricane Katrina in the United States created a storm surge that caused severe and catastrophic damage along the Gulf coast, devastating the cities of Mobile, Alabama, Waveland, Biloxi, and Gulfport in Mississippi, and New Orleans and other towns in Louisiana. Levees separating Lake Pontchartrain from New Orleans were breached by the surge, ultimately flooding 80% of the city and many areas of neighboring parishes for weeks.[3] Both events required a massive humanitarian effort that included military forces.

When these types of events occur, information about the event needs gathering, distributing, and analyzing to determine how best to use resources to help the people in the devastation. Once observers gather information, establishing communications is needed before information can be distributed. Command and Control (C2) node functions perform one or all of the tasks of information gathering, distribution, analysis, decision making, and distribution of decisions. C2 nodes in these situations will be mobile or fixed and will come and go as a mission unfolds. Since C2 systems may have pieces of information that a decision maker will need, interfacing with other C2 systems is necessary in order get a big picture for decision makers to formulate their decisions. The goal of this research is to present and implement a systematic method for the dynamic interfacing of C2 systems. The core of this method is an entity called the Systems Information Broker (SIB). The SIB serves as an arbitrator that will determine whether the interfacing is feasible, and as a uniform interfacing platform to support the interfacing of real-time and non-real-time systems. To aid in the interfacing feasibility, a pre-formulated set of methods will be determined that are predicted to yield interfaces among systems. The focus of this research is determining the constraints on the methods used in interfacing systems that will allow a static calculation that can predict that an interface between systems is feasibility.

2. Basic Architecture and Framework

The goal of this research is to present and prototype a systematic method to facilitate the dynamic interfacing of real-time and non-real-time systems. The core of the method is an entity called an Systems Information Broker (SIB). We propose to breakdown the responsibility of the SIB into two parts with this research focusing on the first part.

1) SIB will serve as an arbitrator that will determine whether the interfacing is feasible by considering the satisfaction of timing constraints and resource usage.

    a) SIB serves as an arbitrator taking into account the reconfiguring issues involved in the enterprise of systems supporting forces and units used to fulfill a new mission. We will need to create a calculation or mechanism for determining whether the proposed methods for interfacing systems are schedulable as systems are dynamically added, deleted, and immigrated.
    b) In addition, we will need to determine the optimality goals and constraints for the resources used on the interfacing systems. Before determining the optimality goals and constraints, we will need to determine the metrics and calculation mechanisms that

determine the current resource use. Once current resource use is know, one can then chose optimality goals and constraints for the interfaced resources. A mechanism will be needed to adjust the resource use to comply with the goals and constraints when resources are being over used. The SIB will use these mechanisms to determine if resources are used properly and make adjustments to comply with goals and constraints, but since we will not have complete information of the global state of the interfaced systems these mechanisms will only give suboptimal resource use. The goal is to still provide effective use of resource, but not necessarily optimal resource use.

2) SIB will also serve as a uniform interfacing platform to handle data interoperability and timing constraints to support the interfacing among systems. SIB will be used in an operational mode and serves as a uniform platform to handle the scheduling of system interactions, and interoperability among systems.

Figure 1 is the framework of the systematic method to facilitate the dynamic interfacing of real-time and non-real-time systems.

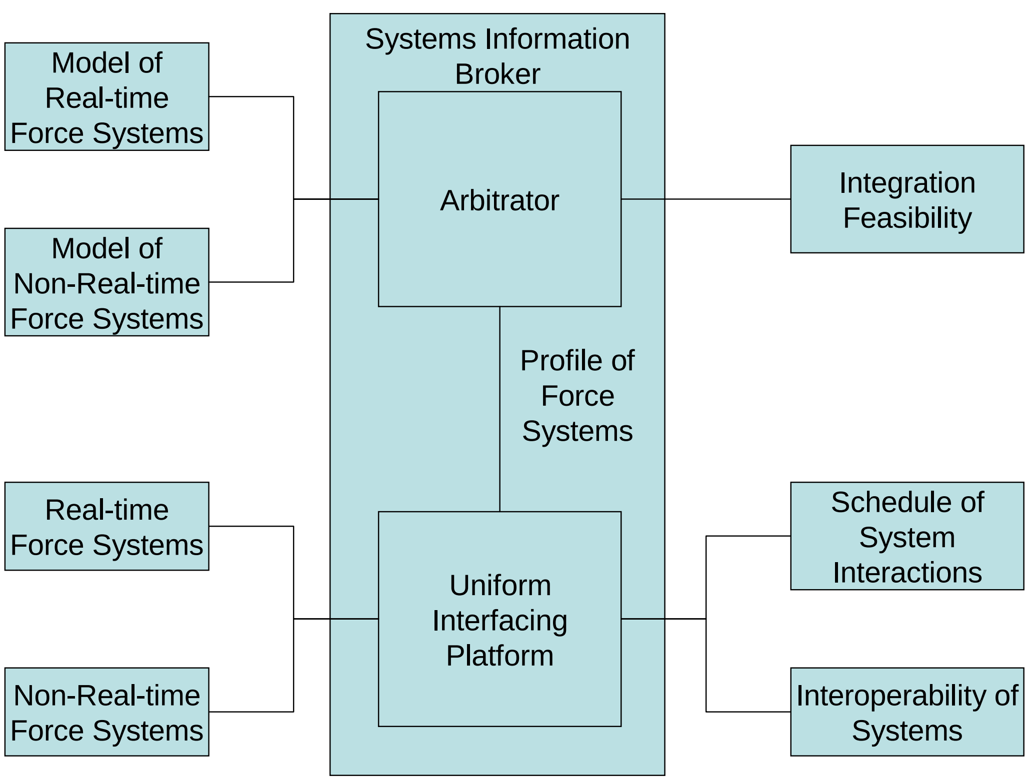

Figure 1 Framework of the proposed method

3. Mechanism for Determining If Systems Can Interface

We plan to develop a criterion that the SIB uses to determine whether interfacing systems is feasible. The SIB will use this criterion to determine if the enterprise of systems is schedulable after some systems are added, deleted, or immigrated from the enterprise. We imagine that this schedulability determination will be similar to embedded real-time schedulability but with a higher magnitude of timing constraint values due to network delays and jitter. The immigrating capability will be limited to non-real-time systems. To develop this criterion, a dynamic scheduling analysis algorithm for distributed systems with non-real-time and real-time tasks needs to be presented. Breaking down the mission the enterprise is required to accomplish into phrases of operation and placing systems to address one phase may provide a way to bound the timing constraints that the systems handling a phase needs to meet. Calculating probabilities of moving from one phase to another may provide an additional means of bounding the timing constraints among real-time and non-real-time systems.

Another criterion that the SIB uses is to determine whether the enterprise of systems still has good resource usage after some systems are added, deleted, or immigrated from the enterprise. To develop this criterion, a resource usage metric needs to be defined and a method will be developed to compute this metric.

This framework will model systems with the base component being a single system. The model will not go much below the system level. A description of a system will include characteristics of the system and applications running on the system.

3.1. Alternative Methods for Interfacing Systems

The idea is to have several methods to choose from when interfacing systems. Methods will be ranked by scoring criteria that is explained in the next section, with the top scoring method being the primary interfacing method and the remaining methods as alternatives.

3.1.1. Modeling Systems Being Interfaced

Applications used on a system vary in degrees of complexity from simple to very complex. On the complex end are applications that require many data sets, perform a high number of calculations on a subset of the data sets, and graphically render the calculated results. Without these data sets, desired results may not be precise enough to be useful. Many times developers will have an idea of what data their applications need to ingest, but these data sets are not well documented such as what the data is making up the data set, the way data is being gathered, what organization is maintaining the data sets, or how to get access to the data sets.

Table 1 contains the attributes that can determine the complexity of an application. An application that is real-time and performs any of the other attributes would be considered a very complex application. An application that is non-real-time, calculation intensive, data intensive, and graphical, or has only the real-time attribute is a complex application. A semi-complex application would be non-real-time and has two of the three remaining attributes. A simple application would be non-real-time and has only one of the remaining attributes.

Table 1 Complexity Attributes for an Application

Complexity	Real-time	Calculation Intensive	Data Intensive	Graphical
	Timing constraints bounding computed results	Requiring high number of math calculations	Requiring high number of data points	Requiring high use of graphics to rendering information
Simple				X
Simple			X	
Simple		X		
Semi-Complex			X	X
Semi-Complex		X		X
Semi-Complex		X	X	
Complex		X	X	X
Complex	X			
Very Complex	X			X
Very Complex	X		X	
Very Complex	X		X	X
Very Complex	X	X		
Very Complex	X	X		X
Very Complex	X	X	X	
Very Complex	X	X	X	X

Without knowing what data sets or information one needs or an application, process, or system needs, one cannot perform processing with the data sets in order to get results from a formula or model. Before processing formulas or models, data is needed and depending on the use of the formulas or models, continuous processing of formulas or models may also require continuous access to the data sets or results to be relevant to a user. Relevant means that a user will be able to take actions to avoid harmful consequences.

Understanding what data sets are used is a good starting point to keeping an application relevant to its users. Keeping the data sets organized, and knowing who and where to get updated data sets also adds to an application relevance to a user.

Making the data sets organization simple to understand may make its maintenance easier.

Information base used to keep an enterprise of systems working may do well when the data sets making up the information base is organized.

Having a flat organization of data sets may be an ideal way to understand the data sets that an application or system will otherwise need.

Real-time systems used by the forces will be modeled using the following attributes:

1) Tasks
2) CPU Cycles used
3) Network resource usage
4) Time Constraints
   a) Finish Within
   b) Maximum Execution Time
5) Periodic, Event Driven, or Both for Task Execution

Non-real-time Force systems will be modeled similar attributes as follows:

1) Tasks
2) CPU Cycles used
3) Network resource usage
4) Periodic, Event Driven or Both for Task Execution

3.1.1.1. Breaking down the Resources

Resources used by systems will be broken down into system resources and network resources. Systems resources are further decomposed into CPU resources, memory resources, and I/O resources. Tied with each resource are resource concerns that have the potential of diminishing the quality of service of the resource.

Table 2 and Table 3 below have the resources being modeled plus the resource concerns for each resource.

Table 2 Real-time System Resources Modeled

Resource	Resource Concern
System	
CPU	Lack of CPU cycles to complete task calculation that causes task to miss its deadline.
Memory	Lack of memory causing task to miss deadline.
I/O	Waiting for I/O resources that causes task to miss deadline.
Network	
Bandwidth	Lack of Bandwidth that causes task to miss deadline.
Quality of Service	Jitter and Latency that degrade information flow and causes task to miss deadline.

Table 3 Non-real-time Systems Resources Modeled

Resource	Resource Concern
System	
CPU	Lack of CPU cycles prevents task from completing computations in a usefully timeframe.
Memory	Lack of memory prevents task from completing computations in a usefully timeframe.
I/O	Waiting for I/O resources prevents task from completing computations in a usefully timeframe..
Network	
Bandwidth	Lack of Bandwidth case prevents task from completing computations in a usefully timeframe.
Quality of Service	Jitter and Latency that degrade information flow and prevents task from completing computations in a usefully timeframe.

3.2. Scoring Criteria of Interfacing Methods

Interfacing methods are scored using criteria of the interfacing latency, capacity, and quality of service which includes availability and reliability. Other criteria may include cost of using the interface.

3.2.1. Maintaining System Schedule

Interfacing of systems must not interfere with individual systems processing to meet task schedules. System schedule is first determined by the time-budget of mission threads that the system supports. Development of time-budget allocations for time-critical mission threads is a recommendation of the Committee on C4ISR for Future Naval Strike Groups.[4]

3.2.2. Maintaining Optimality Goals and Constraints with Respect to Resource Usage

The goal is to use system individual system resources to close to maximum, at least 80%, with 20% to surge processing. But the emphasis is to maintain the system processing schedule even at the cost of underutilized resource use.

4. Conclusion

[1] Joint Doctrine for Military Operations Other Than War, Joint Pub 3-07, 16 June 1995.
[2] 2004 Indian Ocean Earthquake, Wikipedia, http://en.wikipedia.org/wiki/2004_Indian_Ocean_earthquake, Accessed on 12 November 2006.

[3] Hurricane Katrina, Wikipedia, http://en.wikipedia.org/wiki/Hurricane_Katrina, Accessed on 12 November 2006.
[4] C4ISR for Future Naval Strike Groups, Committee on C4ISR for Future Naval Strike Groups, National Research Council, 2006, http://www.nap.edu/catalog/11605.html, Accessed on 13 November 2006.