10th International Command and Control Research and Technology Symposium

The Future of C2

A Measurement and Monitoring System for Tracking and Visualizing Collaboration Metrics in Real-time and for Later Analysis

Topic: **Assessment, Tools, and Metrics**

Peter R. Bono, Ph.D. **

Peter R. Bono Associates, Inc.
PO Box 1047
Niantic, CT 06357 USA

Phone: +1-860-446-8100
Fax: +1-860-446-8750

pbono@prba.com

**** Corresponding Author**

George S. Carson, Ph.D.

GSC Associates
2727 Xanthia Court
Denver, CO 80238 USA

Phone/Fax: +1-303-388-6355

carson@gscassociates.com

Abstract

A substantial body of literature proposes metrics that purport to predict or explain certain aspects of human or system behavior during planning and operations. But it is costly to devise and conduct experiments to test the usefulness of such metrics empirically. At present, there are no computer-based tools that integrate with existing computer-based systems to gather raw data and analyze them effectively. Experimenters need to build a custom set of tools along with designing and conducting the experiment. This is costly and time-consuming—deterring all but the most determined and well-funded researchers.

In military command and control, we are particularly interested in ad hoc collaborations formed and then disbanded in days, weeks, or months—collaborations whose purpose is to answer a set of questions, formulate a plan, or analyze a problem. Increasingly, these collaborations are multicultural and multi-organizational in nature: the participants are from different countries with different traditions and educational and value systems. We have designed and implemented a Measurement and Monitoring System (MMS) to support empirical studies of any system in which it makes sense to gather, calculate, store, and visualize summary information (abstracted as metrics) derived from the data exchanged among the components of the system (abstracted as messages).

Overview

Social scientists, cultural anthropologists, and other researchers have proposed models that purport to predict, and sometimes explain, aspects of human behavior relevant to collaborations integral to planning for effects-based operations [6][8]. These models are often represented by or abstracted as a set of one or more metrics—that is, numerical values calculated over some collection of raw data (email messages, biographical information, etc.). In the command and control domain, we are particularly interested in ad hoc collaborations formed and then disbanded in days, weeks, or months—collaborations whose purpose is to answer a set of questions, formulate a plan, or analyze a problem. More and more, these collaborations are multicultural and multi-organizational in nature: the participants are from different countries with different traditions and educational and value systems.

Before incorporating these metrics and models into operational systems, we would like to have more than anecdotal evidence that they are, in fact, effective and useful predictors of human behavior. However, empirically demonstrating their utility is difficult and expensive. There are presently no computer-based tools that integrate with existing computer-based systems to gather raw data and analyze it effectively. One is usually confronted with the need to build a custom set of tools along with designing and conducting the experiment. This is costly and time-consuming—deterring all but the most determined researchers

We have designed and implemented a **Measurement and Monitoring System (MMS)** that can be used to support empirical studies of any system in which it makes sense to gather, calculate, store, and visualize summary information (abstracted as *metrics*) derived from the raw data exchanged among the components of the system (abstracted as *messages*). The architecture of MMS is modular, with components exchanging information via standardized protocols and

interfaces. In particular, we have implemented MMS as an assembly of loosely-coupled independent processes running on one or more servers. We use Web Services [11] to hide all implementation-specific details such as database table structures, field names, and the like. Similarly, all calculations of metrics over the collection of messages known to the MMS are encapsulated into special-purpose *modules*—each module responsible for the calculation of one or more closely-related metrics. New metrics can be added easily and incrementally into the system without causing any reworking of the other modules. Once a new metric is added, all other functions of MMS automatically work with the new metrics.

In the current version of MMS, which was utilized in the 2004 preliminary phase of DARPA's Integrated Battle Command (IBC) project [7], there are modules that:

- Capture and store messages from web-page editing sessions, email, and chat.
- Calculate ten different metrics (e.g., percent of collaboration participants who have sent at least one message in the past 24 hours) over the message collection and store these metric values along with their calculation times in a metric history database.
- Generate bar charts of groupings of metrics and line graphs showing the time history of the values of a single metric.

In addition, MMS can be used in two modes: real-time and post-exercise reconstruction. In real-time mode, the various modules are running concurrently during collaboration. As messages are generated, they are captured and stored in the message collections. The collaboration forum moderator can monitor the metrics and take action to facilitate the forum if s/he detects negative trends and problems that need mitigation. In post-exercise reconstruction mode, a previously-captured message collection is available for processing. At user-specified intervals, all the metrics are calculated for the appropriate subset of messages gathered to date. Then a sequence of snapshot images is generated. A Snapshot Viewer module allows a researcher to step through the sequence of charts and graphs looking for significant trends. The observed trends can be correlated with the known outcome of the collaboration to evaluate the effectiveness of existing or newly-defined metrics at predicting the state of a collaboration. Metrics validated in this manner can be used in subsequent real-time collaborations to improve collaboration outcomes.

Previous Research

Groupware to support collaborative work has been the subject of extensive research. The Intelligence Community Collaboration Baseline Study [5], completed in 1999, provides a summary of this research and an extensive set of references. It is recognized that collaborative efforts may fail due to factors such as lack of participation, suggesting that monitoring groupware use is essential [5]. Many researchers including Samarasan [10] have shown that collaboration may be modeled as a communication activity among the participants and suggested that collaboration may be studied by analyzing communications among participants.

It is widely recognized that measurable individual and group characteristics have profound influences on the effectiveness of collaborations. Hofstede [6] provides an excellent summary of much of this research. Among the many characteristics, trust is widely recognized as central to effective virtual teams [5]. Jarvenpaa and Leidner [8] [9] have investigated trust in small group collaborations and have demonstrated that the degree of trust in such groups can be accurately

predicted from characteristics of the group's communications including the delay in responding to others and the number of individuals in the group who communicated on a regular basis. Even though Jarvenpaa's and Leidner's work was based on after-the-fact analysis of logged communications, it suggests that similar measures of effectiveness could be collected and evaluated in real-time to give feedback on effectiveness to the moderator of an ongoing collaboration. Recent work by Bradley and White [3] has shown that interpersonal intervention (including intervention by a moderator) has a predictable effect on team performance. Bonito [2] has shown that models of participation in small groups can be effective predictors of group performance. Finally Alavi and McCormick [1] have reviewed past studies of team member collective orientations and suggested different measures of team member's collective orientations.

Definitions

In the sequel we make use of the following specialized terms:

A *collection* is an unordered set of *messages* obtained by instrumenting and capturing some stream of actions or communications.

A *domain* is a subset of a collection that has meaning within the context of an experiment. For example, in a planning experiment, each separate collaboration set up to work on a different aspect of the experiment (e.g., political, economic, military) could be considered a domain.

A *metric* is a value calculated over some domain: that is, over some subset of the whole collection of *messages*. A metric value may be an integer, a real number, or one value of an enumerated type. Each metric has an associated unit of measure such as minutes, percentage, or messages per hour.

A *message* is a raw quantity of information gathered, processed, and stored by the *MMS*. While messages are most easily understood as familiar communications items (e.g., email or chat messages), in the MMS context, a message can be any quantity of information belonging to a collection of like-items over which we wish to calculate, track, and display metrics.

A *system-under-test* is a collection of software and hardware that is being instrumented to provide the collection of messages over which the metrics will be calculated.

The *MMS* is a system that accomplishes four principal tasks:

1. gathering,
2. calculating,
3. storing, and
4. visualizing.

MMS is partitioned into two independent subsystems: the Measurement Subsystem and the Monitoring Subsystem.

The *Measurement Subsystem* is responsible for gathering raw messages and storing them for later use by the rest of MMS. This requires instrumentation of the *system-under-test* so that messages can be gathered at the appropriate points during an experiment. The Measurement Subsystem also contains the domain-specific logic involved in calculating the metrics. Each problem domain will have different metrics that are of interest and are the focus of specific experiments.

The *Monitoring Subsystem* is responsible for storing metrics and making them available for viewing by the users of the system. All visualization takes place in the Monitoring Subsystem of MMS.

The two subsystems interact when the Measurement Subsystem requests that the metrics be stored after they have been calculated and the Monitoring Subsystem allows these metrics to be retrieved for subsequent analysis and display.

Characteristics of MMS

The desired characteristics of an MMS are:

- Simplicity. MMS should be easy to install and operate. While advanced users require fine control over the installation and operation, novice users should be able to get good results with the system defaults.
- Scalability. MMS should scale both in performance and in the volume of messages and metrics that can be handled. Arbitrary limitations should be minimized or eliminated. The database must have the throughput to handle the expected volume of transactions. It should be possible to distribute the functions of the MMS across multiple computers (servers) in order to increase performance and responsiveness.
- Robustness. MMS should not fail except under the most extreme situations, such as loss of power or failure of disk hardware. Most or all software failures should be handled gracefully, so that experimental data gathered to date will not be lost and the experiment can proceed. Automatic recovery should be implemented where possible. In addition, it must be possible to restart the MMS and pick up from where the MMS was last operating properly. The database must be robust and not be the source of system failures.
- Modularity. The MMS components should be loosely coupled; that is, they should not share in-memory data structures and should communicate with each other only via programmatic interfaces such as Web Service calls and/or via information stored in the database. While there will be a minimum set of modules required to perform all the functions of the MMS, it should be possible to test the MMS even if not all the modules are operating.
- Configurability. MMS should allow components to be added or removed without affecting the operation of other components. Where necessary, components should be able to detect the presence or absence of other components and make adjustments in their behavior accordingly.
- Extensibility. MMS should support the definition of new metrics, new message gathering methods, or even new message types in the Measurement Subsystem without adversely affecting the operation of the Monitoring Subsystem. Likewise, MMS should allow new

metric visualizations (e.g., gauges, sliders) without adversely affecting the operation of the Measurement Subsystem.

- Generality. While the initial target experimental scope is collaborative frameworks with email, chat, shared whiteboards and documents, etc., the Monitoring Subsystem should be completely independent of the meaning of the metrics. For each new experimental scope, MMS should allow implementation of new instrumentation techniques and new metric calculation components in the Measurement Subsystem. However, the overall architecture will be retained and much of the actual software that implements the various Measurement Subsystem components should be able to be repurposed in the new experimental scope. The fundamental concepts of time stamps, durations, message bodies, attachments, originators, addressees, etc. should carry over to many other scopes.
- Intuitiveness. The visualization techniques used to display the metrics—both their current (instantaneous) values and their histories over time—should be immediately comprehensible to the user. Little or no training should be required to use the Monitoring Subsystem. It is expected that some hours of training will be required to install, configure, and start up the whole MMS, but that, once operating, little or no training is required to keep it running and to use its results.
- Multiple-levels of Detail. For experiments generating a large volume of messages or employing a large number of metrics, it is desirable that the level of detail of the information presented to the user is controllable by the user.
- System-independence. The implementation modules should, as much as possible, be able to be hosted on different computing platforms and interoperate across LANs and WANs. Internet standards such as HTTP, SOAP, XML, and SQL should be used wherever possible.
- Minimal demands on users (clients). System defaults should allow the novice user to obtain useful results. System configuration and maintenance should be via interactive application programs that give visibility to all configuration parameters. Drop-down boxes, radio buttons, and other such familiar controls should be used to limit the possibility of user error. Wherever possible, the user should be allowed to choose among permitted alternatives rather than being expected to know and type the appropriate value into a text field.

Implementation Overview

MMS is based on a modular system architecture whose components exchange information via standardized protocols and interfaces. In particular, MMS is implemented as an assembly of loosely-coupled independent processes running on one or more servers. Web Services are used to hide all implementation-specific details such as database table structures, field names, and the like. Similarly, all calculations of metrics over the collection of messages known to the Measurement Subsystem of MMS are encapsulated into special-purpose components—each components responsible for the calculation of one or more closely-related metrics. New metrics can be added easily and incrementally into the system without causing any reworking of the other components. Once a new metric is added, all other functions of MMS automatically work with the new metrics.

Architecture

Figure 1 shows the main components of the MMS architecture. The components whose names contain the word "Manager" are implemented as Web Services. The two subsystems—Measurement and Monitoring—are separated by the dashed line in Figure 1. The function of each of the principal components is described in the following section.

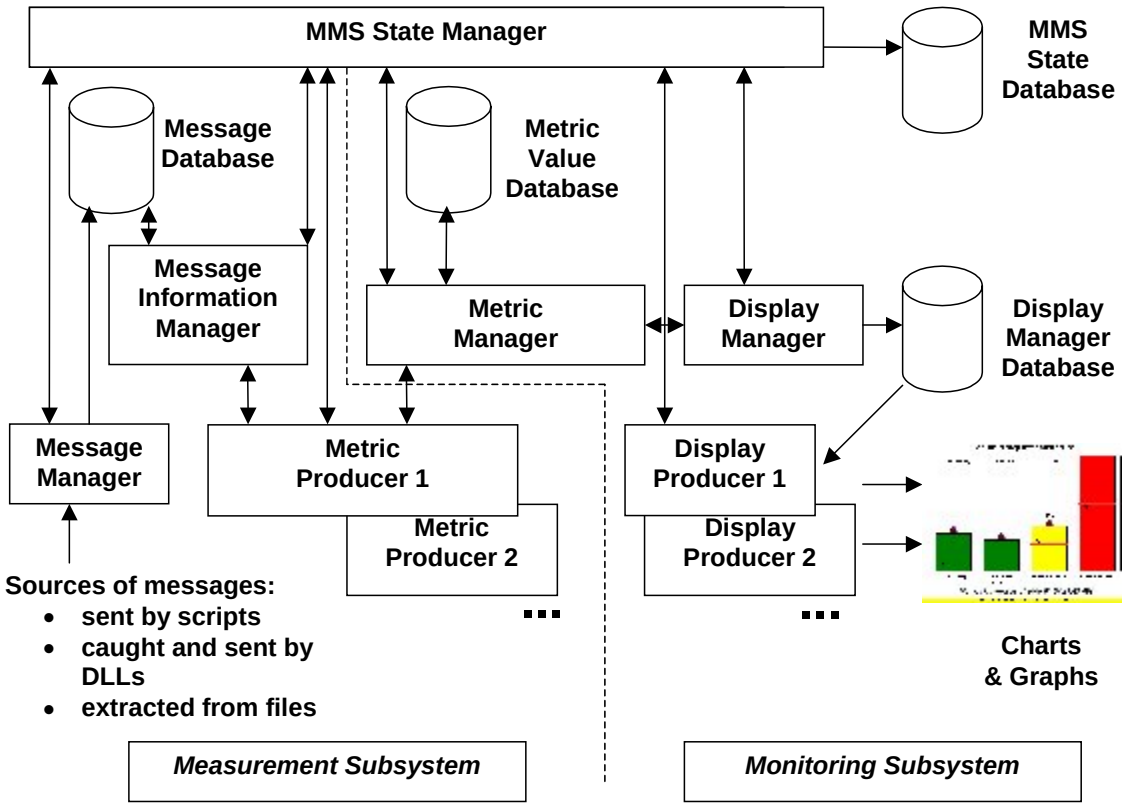

Figure 1. MMS Architecture

Description of the Main Components

The **Message Manager** is a Web Service responsible for:

- Parsing the raw messages and storing their information (sender, recipient list, subject, priority, body, attachments, etc.) in the Message database.
- Keeping track of subsets of the whole message collection (called *domains*) and keeping track of which users are participating in which domains.
- Keeping track of lists of related messages (called *threads*)—subsets of a domain. Threads are a succession of messages on the same topic. Not all messages can be assigned to a thread.
- Building counts and indices to facilitate search and retrieval by the Message Information Manager.

The **Message Information Manager** is a Web Service that:

- Calculates some basic information about the collection of messages and participants in different domains. These so-called *external metrics* are based on information contained in the message headers and do not depend upon the specific content found in the message bodies, subject line, or attachments.
- Retrieves counts on demand and makes them available to the Metric Producers.
- Provides filters so that the external metrics returned by the Message Information Manager are calculated over different domains and windows of time.

The **Metric Producers** are applications that are responsible for the calculation and storage of one or more closely-related metrics. They run periodically and, each time they run, they retrieve information from the Message Database by calling the Message Information Service, re-calculate their assigned metrics, assign a timestamp to that new value, and request that the new metric values be stored in the Metric Value Database. They accomplish the latter by calling the Metric Manager Service with the newly-created metric objects.

The **Metric Manager Web Service** hides the details of the Metric Value Database from the other components in the system. It contains basic functions for storing and retrieving sets of metric values. Either the current values of each metric or all the values for a single metric can be retrieved. The Metric Manager also provides filters so that the metric values returned by the Metric Manager are obtained over different domains and windows of time.

The **Display Manager** is an application responsible for retrieving specified groups of metrics and preparing them for display in charts and graphs. The Display Manager reads an XML-formatted *metrics description file* to assist it in making the following decisions for each metric:

- What is the maximum value expected for the metric (this determines the top value charted on the Y-axis of the visualization)?
- For the bar chart display, what are the histogram bins associated with the range of values for the metric? That is, how many bins are there and what is the top value in the range for each bin.
- What color should be used to display values that fall within each histogram bin?
- Is there a value that should trigger the generation of a user-alert? If so, what is the criterion required to generate the alert (e.g., >10, <=50%).

Periodically, the Display Manager calls the Metric Manager Service to obtain the latest metric values, tracks the changes in value from one instance of a given metric to the next, and records whether the new value is “no change”, “higher” or “lower” as compared with the previous value.

The Display Manager stores its intermediate results in the Display Manager Database.

The **Display Producers** are applications that directly access the Display Manager Database and produce the visualizations viewable by the user. Each Display Producer can produce one type of visualization (*graphic*). Each Display Producer reads an XML-formatted *window description file* to assist it in making the following decisions about each graphic:

- What is the size of the resulting graphic?
- What is its title?
- How are the axes labeled?
- Is there a legend?
- Should the alert values and criteria be included in the graphic?
- Should the trend indicators be included in the graphic?

For each graphic, a full size image and a thumbnail image are produced and stored in a known folder on the host server.

Periodically, the Display Producers redraw and store the graphics using the most current values of the metrics. Thus, over time, the graphics change reflecting the changes in the metrics.

The Display Producers learn what metrics are desired by accessing MMS State information via the MMS State Manager. In particular, each Display Producer can be instructed to produce one or more bar charts and line graphs. Each bar chart can consist of one or more metrics, each metric plotted as a single bar chart. Each line graph is the time history of a single metric.

The final major system component is the **MMS State Manager Web Service**. It manages all system configuration parameters, such as:

- *paths* to key resources such as database files, metric description files, and windows configuration files.
- *module-specific constants* such as whether debug tracing should be on or off, what are the maximum number of metrics supported by the system, and default file names. Other such constants control how often the various modules are scheduled for execution
- *display producer configuration information* specifying which metrics across which domains should be turned into graphics and saved.

The MMS State Manager also keeps track of two items of system state:

- A *list of "active" domains* along with the time the domain was started. Knowledge of this information allows the Display Producers to start their time lines at an appropriate point for each time history metric plotted.
- A *list of "running" processes*. This allows the MMS to know which components to "kill" when the user requests the MMS to stop.

Visualizations

In principal, any visualization technique could be utilized to display the information contained in the Display Manager Database. MMS implements two Display Producers:

- Display Producer 1 produces a bar chart of a subset of the available metrics (see Figure 2). The current values of each metric are obtained. Each metric is mapped to a single bar. The color of the bar is the color associated with the histogram bin correlated with the current value of the metric. The metrics are labels on the X-axis; the maximum values and units for each metric are given above the bar. A trend indicator can be shown, and alert values can also be specified.

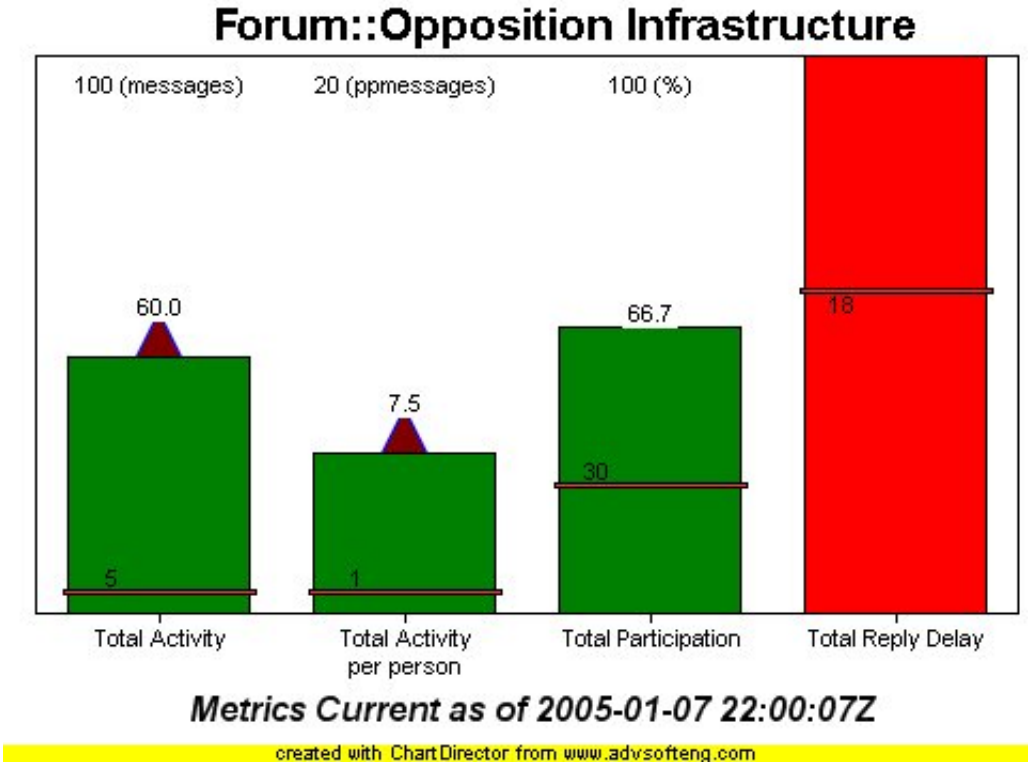

Figure 2. Sample Bar Chart of four metrics produced by Display Producer 1.

- Display Producer 2 produces a line graph of the time history of a single metric (see Figure 3). All the values of that metric from the time the domain became active are displayed. Each point is labeled with its value as well as the time when this value was calculated by the corresponding Metric Producer. The color of the line is the color associated with the histogram bin correlated with the latest value of the metric. The alert value and criterion is displayed as a band. Whenever the line graph falls into that band, the corresponding metric value was in alert status. The X-axis shows time increasing to the right; time is measured as an increment from the domain start time. The Y-axis contains the units label and the legend specifies which metric is being displayed.

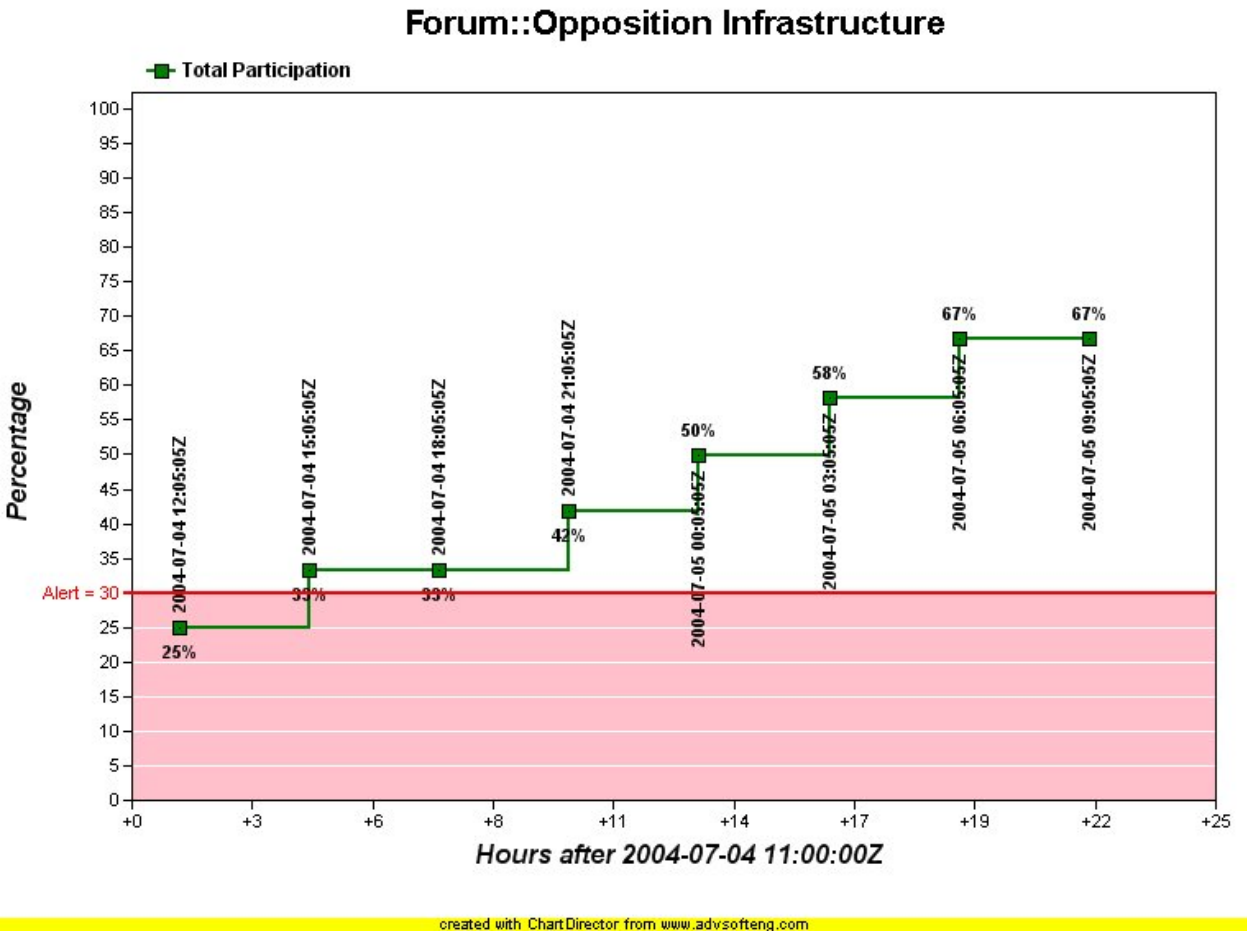

Figure 3. Sample Line Graph of the Total Participation metric produced by Display Producer 2.

Current Implementation

In the current version of MMS, which was utilized in the 2004 Rolling Start of DARPA's Integrated Battle Command (IBC) project [7], there are components that:

- Capture and store messages from web-page editing sessions, email, and chat by calling the Message Manager Web Service.
- Calculate ten different metrics (e.g., percent of collaboration participants who have sent at least one message in the past 24 hours) over the message collection and store these metric values along with their calculation times in a metric history database. The first six metrics are encapsulated into Metric Producer 1, and the remaining four metrics are encapsulated into Metric Producer 2.
- Generate bar charts of groupings of metrics and line graphs showing the time history of the values of a single metric. As explained above, bar charts are created by the Display Producer 1 module and line graphs (time histories) by Display Producer 2.

The current implementation of MMS can be used in two modes: *real-time* and *post-exercise* reconstruction.

In real-time mode, the components of MMS are running concurrently during collaboration. As messages are generated, they are captured and stored in the message collections. Periodically, the metric-producing components look over the collection and calculate and store updated values for the various metrics. As these metrics are updated, they are propagated through the MMS system until they are picked up by the display producing components that proceed to update the charts and graphs for which they are responsible. The collaboration forum moderator can monitor the metrics and intervene in the forum if s/he detects negative trends and problems that need mitigation.

In post-exercise reconstruction mode, the entire message collection is available for processing. At user-specified intervals, all the metrics are calculated for the appropriate subset of messages gathered to date. Then a sequence of snapshot images is generated. A Snapshot Viewer component permits a user to step through the sequence of charts and graphs looking for significant trends. The observed trends can be correlated with the known outcome of the collaboration to evaluate the effectiveness of existing or newly-defined metrics at predicting the state of collaboration. Metrics validated in this manner can be used in subsequent real-time collaborations to improve collaboration outcomes.

Future Research

The current implementation provides useful functionality and proves the concept of MMS; however, additional features will be added. These include:

1. interfaces to other sources of messages including other e-mail and chat servers;
2. components to calculate additional metrics,
3. additional automated configurability, and
4. a “drill down” capability for exploring the data behind the metrics.

In addition, the usefulness of the current set of 10 metrics needs to be evaluated by experiment. The experiments conducted as part of the initiation phase of DARPA’s IBC program were of short duration and did not involve a large enough set of collaborations nor did they result in sufficient message traffic to validate the currently-implemented set of 10 metrics.

Commercialization

Measurements of system status and predictions of future behavior are widely used in planning and operations throughout government and industry. MMS is well suited for these purposes. The present implementation on the Microsoft Platform—XP, Windows 2000 Server, SQL Server, ASP.NET, and Web Services should be attractive as a commercial product to the majority of customers. A Java-based, Unix-hosted implementation could be created if there were sufficient demand.

References

1. Alavi, S. B. and J. McCormick (2004). "Theoretical and Measurement Issues for Studies of Collective Orientation in Team Contexts." Small Group Research 35(2): 111-127.

2. Bonito, J. A. (2002). "The Analysis of Participation in Small Groups: Methodological and Conceptual Issues Related to Interdependence." Small Group Research 33(4): 412-438.

3. Bradley, J., B. J. White, et al. (2003). "Teams and Tasks: A Temporal Framework for the Effects of Interpersonal Interventions on Team Performance." Small Group Research 34(3): 353-387.

4. Hardy, C., N. Phillips, et al. (2003). "Resources, Knowledge and Influence: The Organizational Effects of Interorganizational Collaboration." Journal of Management Studies 40(2): 321-347

5. Hall, Tamara. (1999) “Intelligence Community Collaboration Baseline Study Final Report.” Available from the www address: http://collaboration.mitre.org/prail/IC_Collaboration_Baseline_Study_Final_Report/toc.htm .

6. Hofstede, Geert. (2001) *Cultures Consequences*. Second edition, Sage Publications, Thousand Oaks, California.

7. Integrated Battle Command. DARPA Solicitation BAA-0514. See www address: http://www.darpa.mil/ato/solicit/IBC/ .

8. Jarvenpaa, Sirkka L., Kathleen Knoll, and Dorothy E. Leidner. (1998) “Is Anybody Out There? Antecedents of Trust in Global Virtual Teams.” Journal of Management Information Systems, 14(4): 29-64.

9. Jarvenpaa, Sirkka L. and Dorothy E. Leidner. (1999) “Communication and Trust in Global Virtual Teams.” Organization Science, 10(6): 791-815.

10. Samarasan, Dhanesh K. (1988) "Collaborative modeling and negotiation." Proceedings of the Conference on Supporting Group Work, Palo Alto, California: 9-12.

11. W3C Web Services Activity, Available from the www: http://www.w3.org/2002/ws/ .