To appear in Proc. MILCOM '97, IEEE Military Communications Conference, Monterey, CA (Nov. 1997).

A HIERARCHICAL MANAGEMENT FRAMEWORK FOR BATTLEFIELD NETWORK MANAGEMENT

Adarshpal S. Sethi
Pramod Kalyanasundaram
Christopher M. Sherwin
Dong Zhu

Department of Computer and Information Sciences
University of Delaware, Newark, DE 19716
{*sethi, kalyanas, sherwin, dzhu*}*@cis.udel.edu*

ABSTRACT

Effective management of battlefield networks requires a hierarchical network management architecture wherein managers can dyanmically delegate management tasks to intermediate managers. In this paper we describe a novel paradigm called the spreadsheet paradigm which extends the traditional flat SNMP management model to a hierarchical architecture. We present a spreadsheet MIB, language, and event model that is located at an intermediate manager controllable by a manager. The spreadsheet-based scripting environment presents an easy to use and comprehend abstraction and interface to extend a standard management framework. We are applying this framework to problems in the management of tactical battlefield networks.

Keywords: Network Management, Hierarchical Management, SNMP, Tactical Internet.

INTRODUCTION

Management of battlefield networks poses unique challenges: rapid configuration and setup, quick adaptation to constantly changing conditions, critical response times, the need for survivable multi-level distributed management and control, and flexible routing and congestion control. Our research in network management at the University of Delaware has led to the development of a hierarchical framework for the management of a complex distributed system. This framework provides a flexible, dynamic management solution for the management of battlefield networks by permitting distribution of control and management functions over a hierarchical management structure. In this framework, management authority can be delegated to Intermediate Managers which can react quickly to changing battlefield scenarios.

Prepared through collaborative participation in the Advanced Telecommunications/Information Distribution Research Program (ATIRP) Consortium sponsored by the U.S. Army Research Laboratory under the Federated Laboratory Program Cooperative Agreement DAAL01-96-2-0002.

To effectively manage large distributed systems and complex internetworks, a management model must go beyond the traditional flat model of a single manager communicating with a large number of agents. Hierarchical management and peer-to-peer communication among managers are attractive alternatives that need to be explored for distributed system management [1]. The need for hierarchical management is particularly acute in tactical battlefield networks which are expected to have tens of thousands of nodes. The Force XXI Battle Command Brigade and Below (FBCB2) of the PM Applique in its System Requirements Review (SRR II, Feb. 4-5, 1997) has identified a hierarchical management strategy for the management of the Tactical Internet as the Army War-Fighter Experiment transitions to its expanded future system versions 2, 3, and 4. In this new management approach, management information will be summarised at each level of the hierarchy and will flow upwards through the Platoon, Company, Battalion, and Brigade levels to the ISYSCON platform. Every FBCB2 will have the ability to activate a net manager function so that there is no single point of failure. Although currently mainly monitoring functions have been identified, the management structure will likely grow to include control functionality as well.

Unfortunately, the most popular management frame-

work in use today, the SNMP framework (which includes both the SNMP and the SNMPv2 protocols) [2], [3], [4], only supports the flat management model. The framework provides no means for managers to delegate tasks to intermediate managers or for peer-to-peer communication between intermediate managers during the execution of these tasks. The FBCB2 management strategy uses SNMP since it is based on the Internet protocol suite and is therefore limited by these weaknesses of SNMP.

Management by delegation is a well-known strategy [5], [6] for implementing hierarchical management, but so far the SNMP community has been unable to take advantage of it because the delegation primitives have not been integrated with the SNMP framework. Our research into new models for distributed system management has led us to propose a new paradigm – which we call the *spreadsheet paradigm* [7], [8] – that incorporates management by delegation concepts into the SNMP framework to facilitate the management of distributed systems. The spreadsheet paradigm allows a manager to offload routine management tasks to an intermediate manager and facilitates user configurability of management information and control in a *value-added* manner. This is achieved by providing a scripting MIB and language specially designed for management tasks in SNMP.

The main objectives of the spreadsheet paradigm are: (a) to introduce a powerful intermediate manager that enhances (but preserves) the existing SNMP framework. (b) to provide an environment that supports user configurability of management information. (c) to support basic primitives, events and operations via a scripting MIB and language to accomplish fairly complex network management tasks; and (d) to present an abstraction and interface that is easy to comprehend and use.

The major benefits of the spreadsheet paradigm are that it permits distribution of control between the manager and intermediate managers through the use of spreadsheet-based scripts and it allows *dynamic* creation of user-defined views of managed objects including relationships between objects belonging to different MIBs. This paradigm fits naturally into the existing Internet Management framework because a spreadsheet is in essence a two-dimensional table which is the fundamental structure supported by this framework. It relieves managers from some of the elementary lower-level chores of management thus allowing them to concentrate on higher level tasks required by a management application. Managers can dynamically decide what actions are to be performed and how they are to be executed at the intermediate manager with a high degree of flexibility.

THE SPREADSHEET PARADIGM

An essential component of the spreadsheet paradigm is an abstraction of a spreadsheet. A spreadsheet is composed of *cells* arranged in a two-dimensional matrix consisting of *rows* and *columns*. The number of rows and columns in a spreadsheet is arbitrary and is only limited by memory constraints in the implementation or platform. A cell is the fundamental unit of operation in the spreadsheet paradigm. A cell contains a *control information part* and a *data part*. The control part can serve as a repository for event specifications, relationships between management objects or references to other cells. The control part dictates the rules for collection of information or relationships between objects. The data part contains the data collected as a result of executing the control information specification. For instance, the control part may specify that the cell should contain the result of summing two or more counters in different nodes (or different MIBs). The data part contains the result of such a summation. A user of the spreadsheet is given the flexibility of creating, deleting and modifying cells. In addition, a user can view and clear data in the cells. Several cells can be combined together to form a complex event which can be stored in another cell. This structure facilitates the construction of hierarchical events.

For example, a user could create a cell which contains the following as its control part: "Is the interface between Node A and Node B of the network down?". The data part of the cell would be "true" if the interface is down and "false" otherwise. Another cell could represent the same condition but between Nodes C and D. Now if the management user wants to check the interface condition for all four nodes, such a condition can be checked by combining the first cell and the second cell and storing this condition in a third cell i.e., the condition for the third cell would be "Are Cell1 and Cell2 true?". Thus cells which have already been defined could be used

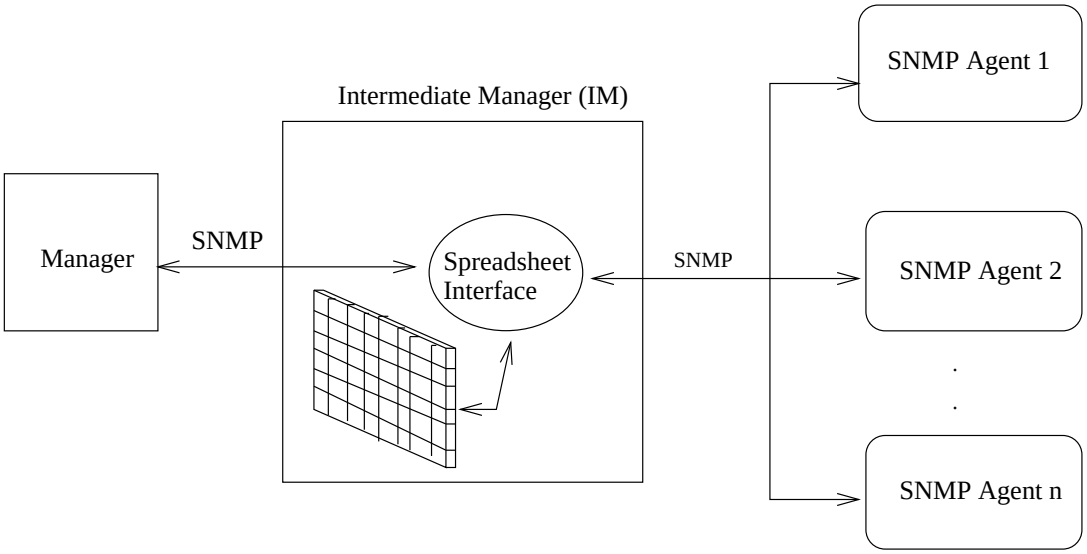

Fig. 1. An Intermediate Manager (IM) with a spreadsheet interface

to form more complex conditions.

Figure 1 shows the spreadsheet abstraction located at a node that functions as an Intermediate Manager (IM). The IM accepts SNMP requests from the manager and enters them into the spreadsheet that is maintained at the IM. The control information entered into the spreadsheet contains scripts written in the Spreadsheet Language (SSL), a specially designed scripting language for use with the spreadsheet paradigm. The scripts are interpreted locally at the IM which may result in SNMP requests to be forwarded to one or more agents. The agents' responses are processed as specified by the scripts in the spreadsheet cells. As a result of this processing, updates may occur in the data contained in one or more cells. An operation initiated by a single cell may result in a data update in the cell which initiated the operation as well as in cells that are dependent on the cell that initiated the operation. The spreadsheet abstraction can be supported using the standard SNMPv2 protocol operations.

SPREADSHEET OPERATION

Figure 2 shows the basic components contained in an implementation of the spreadsheet entity. The primary function of the spreadsheet MIB (*ssmib*) is to implement the spreadsheet abstraction using SNMP tables. User operations on cells map to operations on tables that are part of this MIB. The IM function module coordinates the activities of the various components of the Intermediate Manager. When the IM receives an SNMP request from the manager, the IM function module performs the necessary operations on the spreadsheet MIB to implement the cell abstraction. Once the request has been carried out, the IM function module responds to the manager that requested the operation. If a control value is entered into a cell, the objects that need to be polled are forwarded to the polling subsystem. Once the polling entries are set up, the IM requests them on an as needed basis, for spreadsheet processing. The IM function module interacts with the event model to perform event based processing of the spreadsheet. When the script in a cell needs to be executed, the IM function module interacts with the interpreter to process the cell control information.

The polling subsystem plays a vital role in implementing the spreadsheet abstraction. When a user sets up information in a cell, there is a need to constantly update the value(s) contained in the cell. This can be achieved by polling the managed objects referenced in the cell. In order to perform such polling, the IM maintains polling tables that facilitate the polling process that automatically updates the cell values. This allows a user to view the spreadsheets that are set up at the IM and expect the cells to contain values that are current (within a certain time granularity). The polling subsystem allows polling entries to be set up based on a cell id.

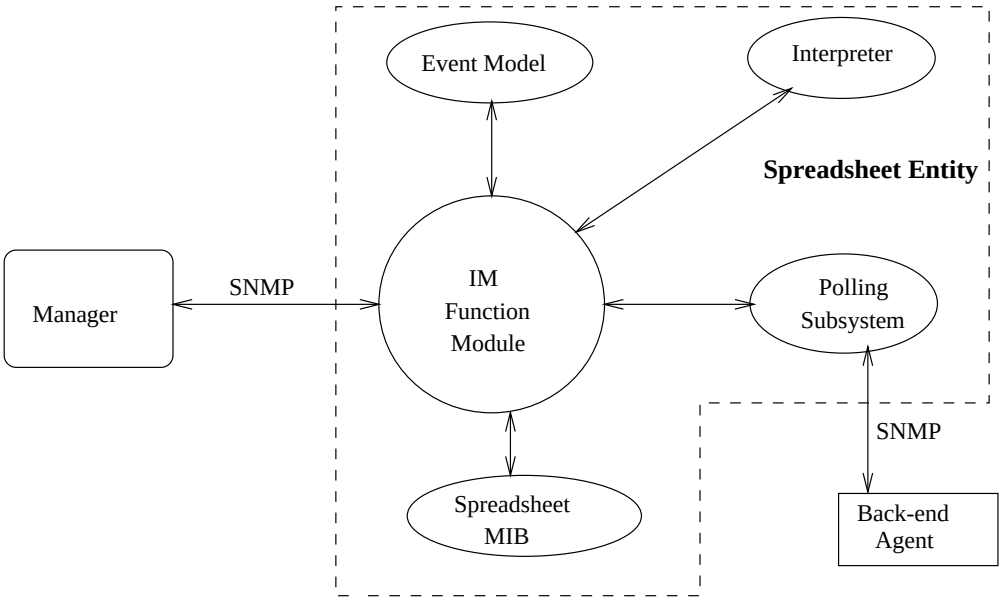

Fig. 2. Spreadsheet Entity

Thus the IM function can request the polling subsystem to set up polling entries associated with a cell and retrieve all the values that are part of a cell.

The spreadsheet paradigm supports a Spreadsheet Scripting Language called SSL that can be used by a user to set up sheets of control. The spreadsheet language is interpreted by an interpreter and scripts that are set up in the various cells can be executed under the control of this interpreter. The interpreter performs the functions of syntax checking, run time error checking, detection and reporting.

A language that targets a network management environment must be able to support features that facilitate the specification of network management tasks coupled with user flexibility and expressive power [11]. The SSL has been specially designed for this purpose. The SSL supports arithmetic operators that allow a user to perform expression evaluation. It further allows a user to specify numeric (dot notation) or symbolic form of managed objects. It allows a user to specify a fully qualified managed object i.e., the OID and the host on which the managed object resides. Since the spreadsheet paradigm supports access to managed objects on different nodes, it is important for a user to be able to specify the actual managed object qualified with a host name. It also supports cell assignment, cell local variables for temporary storage, and other cell manipulation operations. Further, the language provides the standard constructs for iteration (*forall, while*), condition (*if...else*) etc. This allows a user to setup procedural scripts in the cells. A cell can contain a procedure that can be executed from another cell.

The spreadsheet supports two modes of operation: synchronous or request/response mode and asynchronous or event mode. In the synchronous mode, the manager requests some operation to be performed using one of the standard SNMP protocol operations and the IM responds after processing the request. In the asynchronous mode, the user sets up events and actions associated with such events. These events are constantly monitored by the IM. On occurrence of any of the events being watched, the IM carries out the associated actions which may include notifying the manager. This mode of operation allows the manager to successfully delegate some of its routine tasks to the IM. In order to support asynchronous event processing, the spreadsheet entity uses an underlying event model that permits event and temporal criteria specification.

The spreadsheet paradigm defines certain basic events (poll event). The event model allows a user of the spreadsheet paradigm to set up events based on cells and the basic events supported by the paradigm. The user can also use events that have already been defined in conjunction with the SSL, to construct

more complex events. This hierarchical event building can be performed recursively until (in some cases) a complete management task can be modeled as a set of events. The event model uses ordering of events based on the local clock. This ordering obviates the need for clock synchronization. Also, the events are defined at the IM and hence any ordering other than the one based on the local clock at the IM would be inconsistent. The spreadsheet paradigm event model also supports features that allow the backward propagation of events to ensure that all the cells that depend on a particular event are processed and the spreadsheets updated.

CONCLUSIONS

In conclusion, we have presented a new paradigm for network management which will allow the SNMP management framework to be extended to hierarchical management environments such as for battlefield networks. We described the architecture that supports the spreadsheet paradigm and outlined some of the features of the language and event model that form an integral part of this paradigm.

A prototype implementation of the spreadsheet paradigm is currently in progress. The implementation includes a graphical user interface that can be used by a manager to construct, load and execute scripts in the Intermediate Manager. We are also actively examining application of our hierarchical management framework to the management of the FBCB2 Tactical Internet. Specifically, there are problems in the configuration management of routers as nodes move from one subnet to another that could benefit from the use of this framework. Our framework also provides an excellent viable solution for the task of performance monitoring, collection and reduction of data, and passing it on to higher management levels in a flexible manner.

The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government.

REFERENCES

[1] A.S. Sethi, Y. Raynaud, and F. Faure-Vincent, editors. *Integrated Network Management IV*. Chapman and Hall, London, 1995.

[2] J. D. Case, M. S. Fedor, M. L. Schoffstall, and C. Davin. *Simple Network Management Protocol (RFC 1157)*, May 1990.

[3] J. Case, K. McCloghrie, M. Rose, and S. Waldbusser. *Protocol Operations for Version 2 of the Simple Network Management Protocol (SNMPv2) (RFC 1905)*, January 1996.

[4] M. T. Rose and K. McCloghrie. *Structure and Identification of Management Information for TCP/IP based internets (RFC 1155)*, May 1990.

[5] Y. Yemini, G. Goldszmidt, and S. Yemini. Network Management by Delegation. In I. Krishnan and W. Zimmer, editors, *Integrated Network Management II*, pages 95–107. North Holland, Amsterdam, 1991.

[6] K. Arai and Y. Yemini. MIB view language (MVL) for SNMP. In A.S. Sethi, Y. Raynaud, and F. Faure-Vincent, editors, *Integrated Network Management IV*, pages 454–465. Chapman and Hall, London, 1995.

[7] P. Kalyanasundaram, A.S. Sethi, and C. Sherwin. Design of A Spreadsheet Paradigm for Network Management. In *Proceedings of the 7th IFIP/IEEE Workshop on Distributed Systems: Operations and Management*, L'Aquila, Italy, October 1996.

[8] P. Kalyanasundaram, A.S. Sethi, C. Sherwin, and D. Zhu. A Spreadsheet-based Scripting Environment for SNMP. In *Proceedings of the 5th IFIP/IEEE International Symposium on Integrated Network Management (to appear)*, San Diego, CA, May 1997.

[9] M. Hasan. An Active Temporal Model for Network Management Databases. In A.S. Sethi, Y. Raynaud, and F. Faure-Vincent, editors, *Integrated Network Management IV*, pages 524–535. Chapman and Hall, London, 1995.

[10] S. Waldbusser. *Remote Network Monitoring Management Information Base (RFC 1757)*, February 1995.

[11] D. Holden. Predictive Languages for Management. In *Integrated Network Management I*, pages 585–596. North Holland, Amsterdam, May 1989.