SIMULATION MANAGEMENT IN

DISTRIBUTED INTERACTIVE SIMULATION

Huat K. Ng, Ronald S. Klasky, Kenneth P. Kelly
Veda Incorporated
Orlando, Florida

ABSTRACT

The standardization of simulation protocols through the SIMulator NETworking (SIMNET) and Distributed Interactive Simulation (DIS) concepts has allowed the interconnection of dissimilar simulators into an electronic battlefield. A distributed simulation may encompass many different types of systems and the number of entities in an exercise can grow to thousands. As the network traffic increases, an exercise control and management system is critical in order to successfully control the scenario. In DIS, a host computer designated as the Simulation Manager (SM) performs exercise management functions via 12 SImulation MANagement (SIMAN) PDUs. Some functions performed by the SM include: Start, Restart, Maintenance, and Shutdown of an exercise. The focus of this paper is to describe an on-going design and development effort which will result in the test, validation and implementation of the 12 new SIMAN PDUs on a workstation. From this workstation, a DIS exercise manager will be able to utilize the SM software to control all of the entities (live, constructive and virtual) on the battlefield.

ABOUT THE AUTHORS

Huat K. Ng received his Bachelor's degree in Computer Engineering and Master's degree in Electrical Engineering from the University of Central Florida. Mr. Ng has worked with Veda Incorporated since 1993 on realtime simulation networking. He has been involved with Distributed Interactive Simulation (DIS) standard development and implementation of application level gateways to interface dissimilar simulation protocols.

Ronald S. Klasky is a visual systems engineer with Veda Incorporated. Mr. Klasky has a Master's degree in Computer Science, specializing in computer graphics and seven years of graphics programming experience. His duties include developing computer graphics software in support of simulation projects in such areas as terrain database display and graphical user interfaces.

Kenneth P. Kelly has over 13 years in support of various NAWC-TSD and STRICOM programs. He is the Project Manager on Veda's Reconfigurable Ground Vehicle Test Bed project and the DIS Simulation Management effort. Mr. Kelly has a Bachelor's degree in Industrial Engineering and a Master's degree in Engineering Management. He also serves as the Orlando Branch Manager for Veda Incorporated.

SIMULATION MANAGEMENT IN

DISTRIBUTED INTERACTIVE SIMULATION

Huat K. Ng, Ronald S. Klasky, Kenneth P. Kelly
Veda Incorporated
Orlando, Florida

INTRODUCTION

The standardization of simulation protocols via the Distributed Interactive Simulation (DIS) protocol has demonstrated the feasibility of interconnecting multiple distributed simulators via a local/wide area network. The Simulation, Training and Instrumentation Command's (STRICOM) Battlefield Distributed Simulation - Developmental (BDS-D) program has begun to address the extension of creating an entire electronic battlefield. The BDS-D goal is to develop the ability to network geographically separated simulation sites while preserving time/space coherence and synchronization within the limits of human perception and reaction times. In order to successfully manage the entire electronic battlefield of the BDS-D program, an exercise control management capability must be added to the electronic network.

A distributed simulation may encompass many different types of simulator systems and the number of entities in an exercise can grow to the hundreds or thousands of entities. As the number of entities expands, an exercise control and management system is critical in order to successfully control the scenario. In DIS, a host computer designated as the Simulation Manager (SM) performs exercise management functions. Some functions performed by the SM include: start, restart, maintenance, and shutdown of exercises. Other functions provided by the SM include the introduction of late players and the collection and distribution of specific types of data. In the most recent unapproved DIS standard (version 2.0.3) [1], 17 *new Protocol Data Units* (PDUs) were proposed. Among the 17 new PDUs, 12 of the PDUs will be used for simulation management activities. An initial DIS architecture has been presented in the DIS Communication Architecture Subgroup (CAS) and the simulation management activities serve to establish a portion of the Session Database. The Session Database is defined in the DIS architecture as "a standard database which includes network initialization data and simulation entity initialization and control data" [1]. SIMAN PDUs have not yet reached maturity. However, the PDUs have been defined in a manner to provide flexibility and extensibility.

Multiple SMs may exist in a distributed simulation exercise, with an individual SM controlling an individual site or simulation. The multiple SMs may be arranged in a hierarchical structure with each having different responsibilities and levels of authority. The DIS standard does not bar the use of multiple SM hosts to perform exercise control and management. An example of a hierarchical structure of multiple SMs is depicted in Figure 1.0.

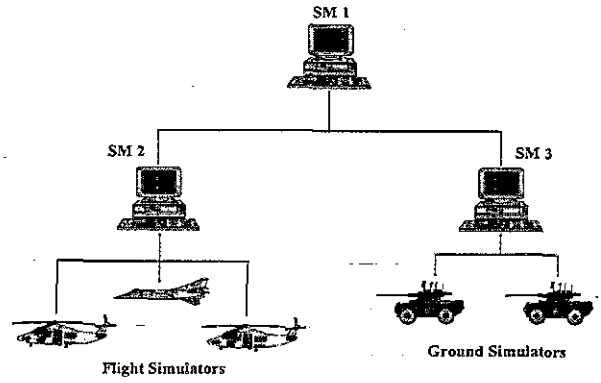

Figure 1.0 Example of a Hierarchical Structure of Multiple Simulation Managers

BACKGROUND

The DIS standard has been approved by the Institute of Electrical and Electronics Engineering (IEEE) as the IEEE P1278 standard which addresses the communication protocols employed between simulation applications. The current state of the simulation entity is conveyed to other simulators using a PDU. A simulation entity is defined as "an element of the synthetic environment that is created and controlled by a simulation application through the exchange of DIS PDUs" [1]. At present, the IEEE P1278 defines only ten PDUs. These PDUs are related to entity interaction and logistics information.

As the DIS standard evolves, new PDUs are added as required. In the most recent unapproved DIS standard, i.e. Version 2.0 Third Draft (or 2.0.3), 17 new PDUs were added to the original IEEE P1278 standard. The DIS PDUs defined in Version 2.0.3 can be logically organized into six different protocol families:

- entity information/interaction
- warfare
- logistics
- simulation management
- distributed emission regeneration
- radio communications

Among the 17 new PDUs described, 12 of the PDUs will be used for simulation management activities. The DIS 2.0.3 will be submitted to the IEEE in mid 1994. The goal of the SIMAN PDUs is to provide a centralized control mechanism of the simulation exercise.

Veda Incorporated, under STRICOM sponsorship, has developed a software package called DISMAN (Distributed Interactive Simulation MANager) to provide distributed simulation management capabilities. DISMAN adheres to the DIS standard 2.0.3. Due to the infancy of the SIMAN PDUs, the enumerated list for datum identifications is short (less than 50) and incomplete in DIS 2.0.3. The enumerated list consists of numerical values associated with each simulation attribute. The enumerated list, however, is being expanded and finalized in the DIS working groups, which consist of industry, academia, and military representatives. The standardization process is continuing with DIS standard workshops held biannually under the sponsorship of STRICOM.

The research, design and implementation of a stand-alone exercise control station, utilizing the 12 SIMAN PDUs, is the focus of this project. DISMAN will be used in the Anti-Armor Advanced Tactical Demonstration (A^2ATD) Number One. Experiments are also being discussed for the 1994 Interservice/Industry Training Systems and Education Conference (I/ITSEC) to provide exercise control, initialization of entities, and automated data collection. The design goals of DISMAN is to provide the flexibility to add future requirements such as the Session Manager. The Session Manager is a planned future expansion of the SM.

SIMULATION MANAGEMENT PDUs

The SIMAN PDUs are described in document [1] to support managerial chores in a simulation exercise. DISMAN will utilize the 12 SIMAN PDUs and will serve to establish a portion of the Session Database for simulations participating in a DIS exercise. The Session Database defines its contents in two major categories: Network data and Simulation Entity data. Within the Session Database, DISMAN falls into the category of Simulation Entity data. The Simulation Entity data category includes the information needed to initialize all of the simulation entities with the required parameters and data to support a specific exercise.

The 12 SIMAN PDUs are as follows:

- Create Entity
- Remove Entity
- Start/Resume
- Stop/Freeze
- Acknowledge
- Action Request
- Action Response
- Data Query
- Set Data
- Data
- Event
- Message

Functionalities of DISMAN

The primary functionality of DISMAN is to use the SIMAN PDUs to actively manage simulation hosts and applications. Management of a simulation includes:

1. maintaining and updating the local simulation session databases on simulation hosts,
2. configuring the simulation applications and entities it simulates,
3. performing diagnostics and other maintenance procedures on simulation hosts and applications,
4. coordinating the entity's involvement in an exercise, and
5. monitoring the applications and entities for signs of problems which the manager should address.

Using the SIMAN PDUs, the manager can set, control, initiate an action, automate data collection, and report an event on an entity or a group of entities. Examples of the usage of the SIMAN PDUs follow:

- Parameters that can be set by DISMAN include location, velocity, orientation, dead-reckoning algorithm, force identification, and entity type. These parameters are defined in the Set Data PDU.
- Controlling an entity's state is accomplished by using the Stop/Freeze, and Start/Resume PDUs.
- An action such as turning on or off VV&A data collection by a simulation asset is accomplished *via the Action Request PDU.*
- Automating data collection for after action review and analysis is accomplished with the Data Query PDU. The Data Query PDU provides the flexibility in setting the frequency of a periodic data response by the simulator. DISMAN can also be used to set event conditions, so that information that is of interest will be sent from the simulator onto the network when a condition is flagged true.

DIS is an application layer protocol, and to verify reliable protocol handshaking there is a corresponding pair of PDUs for each simulation management transaction. For example, the Acknowledge PDU will be used by DISMAN to acknowledge the simulator's receipt of the Start/Resume and Stop/Freeze PDU. Figure 2.0 illustrates the transaction involving the Stop/Freeze and Acknowledge PDU.

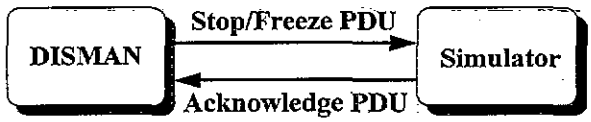

Figure 2.0 Example of a Simulation Management PDU Transaction

Simulation management functions in the DIS protocol are not strictly realtime. Functions such as initialization and configuration management transpire before an entity joins a realtime DIS simulation exercise. To begin an exercise, DISMAN will transmit a Start/Resume PDU with the real-world and simulation time embedded into the PDU. The real-world time corresponds to the time (with respect to Greenwich time) at which the entity is to start/resume the exercise. The simulation time corresponds to the time of day in the simulated world at which the entity will start/resume the exercise. In management functions that involves monitoring and exercise coordination, DISMAN allows the user to specify a lead-time in the corresponding SIMAN PDU. For example, if the manager of an exercise would like to monitor the fuel quantity of a simulator, a Data Query PDU will be sent by DISMAN to the simulator. The Data Query PDU will include a set time interval which specifies the time interval between issues of Data PDUs by the simulator.

Simulation management functions typically do not generate a vast amount of data traffic. Unlike the Entity State PDUs, the SIMAN PDUs do not follow a dead-reckoning algorithm to determine the frequency of PDU transmission. Simulation management functions are transaction oriented. A request from DISMAN invokes a single response from each managed entity. The only exception is in the case of a periodic data response to a Data Query PDU. However, even then, the frequency of transmission is still not as high when compared to a moving entity's amount of Entity State PDU traffic. Due to the non-realtime, low traffic characteristics of the simulation management protocols, DISMAN does not need to be

defined in a manner that minimizes PDU sizes or processing requirements. Therefore, DISMAN is designed with the goal of providing the manager maximum flexibility and extensibility.

Data Requirement Set

The 12 SIMAN PDUs are defined in the proposed DIS Standard 2.0.3. Each SIMAN PDU is preceded with a simulation management header which consists of a PDU header (similar to other PDU families), the originating entity, and the intended receiving entity. The overall length of a simulation management PDU header is 28 bytes.

In order to convey the information between DISMAN and simulation applications, data are represented as identity/value pairs. Each identity/value pair is encapsulated in a fixed datum or variable datum record. To provide the flexibility in communicating information, SIMAN PDUs (Action Request, Action Response, Data Query, Data, Set Data, Event Report, and Message) are defined in a manner that allows concatenation of a variable number of datum records.

The identity of the individual datum fields are defined in [2]. This field is defined as a 32-bit enumeration. For example, the ammunition quantity identity is currently defined in DIS 2.0.3 to be enumeration value 40. Upon receiving a datum identity of value 40, the simulator recognizes that the correponding datum value is the ammunition quantity. The enumerated list is currently minimally defined, ie. only 50 items. However, as the standard evolves, enumeration values can easily be added without redefining the overall PDU structure.

DISMAN SOFTWARE DESIGN

Several factors were taken into consideration in the software design for DISMAN. The software must provide an interface that is easy and intuitive to the manager, yet still provide all the functionalities required by the defined simulation management protocols. DISMAN must be capable of supporting large simulation exercises and must convey important managerial information to the DIS manager. Although the simulation management protocols are not strictly realtime, the manager needs to obtain information regarding the simulation entities in a quick and efficient manner.

To accommodate the list of enumerated datum identities in the DIS 2.0.3 simulation management protocols, DISMAN provides a Graphical User Interface (GUI) to the user. The simulation management parameters are configurable and are entered into DISMAN in a format consistent with the SIMAN PDUs. With the aid of a pull-down menu system the user can easily navigate and select individual datums without the need to refer to the entire defined enumerated list. This feature hides the details of the PDU construction, but still provides the user with a powerful tool.

A Plan-View Display (PVD) is provided to the manager. The PVD displays a terrain database with UTM grid lines. Some of the map features that can be selected for display are trees, roads, lakes, rivers, and contour lines. The coordinate system can be selected to be map-defined UTM or user-defined pseudo-UTM. Simulation entities on the network appear on the PVD as icons with or without their entity identification (site id, application id, entity id) as labels. The mouse is used to select individual entities for the purpose of conveying information via PDUs. Entities can be individually selected or grouped by a "click-and-drag" feature of the mouse. Figure 3.0 illustrates the PVD.

Overall Architecture

The utility of a layered model in describing the overall architecture of DISMAN promotes an architecture that clearly defines each layer's definition and functionalities. In the field of network communications, layering concepts such as the International Standards Organization (ISO) Open Systems Interconnection (OSI) model provides a solution for interoperability of different systems.

The overall DISMAN architecture is composed of three layers: application, assembly, and network. The application layer implements the simulation management functions. The application layer also interprets and performs any acknowledgements to SIMAN PDUs received from the network. A GUI/PVD is provided to the user for ease of accomplishing the SIMAN applications. The

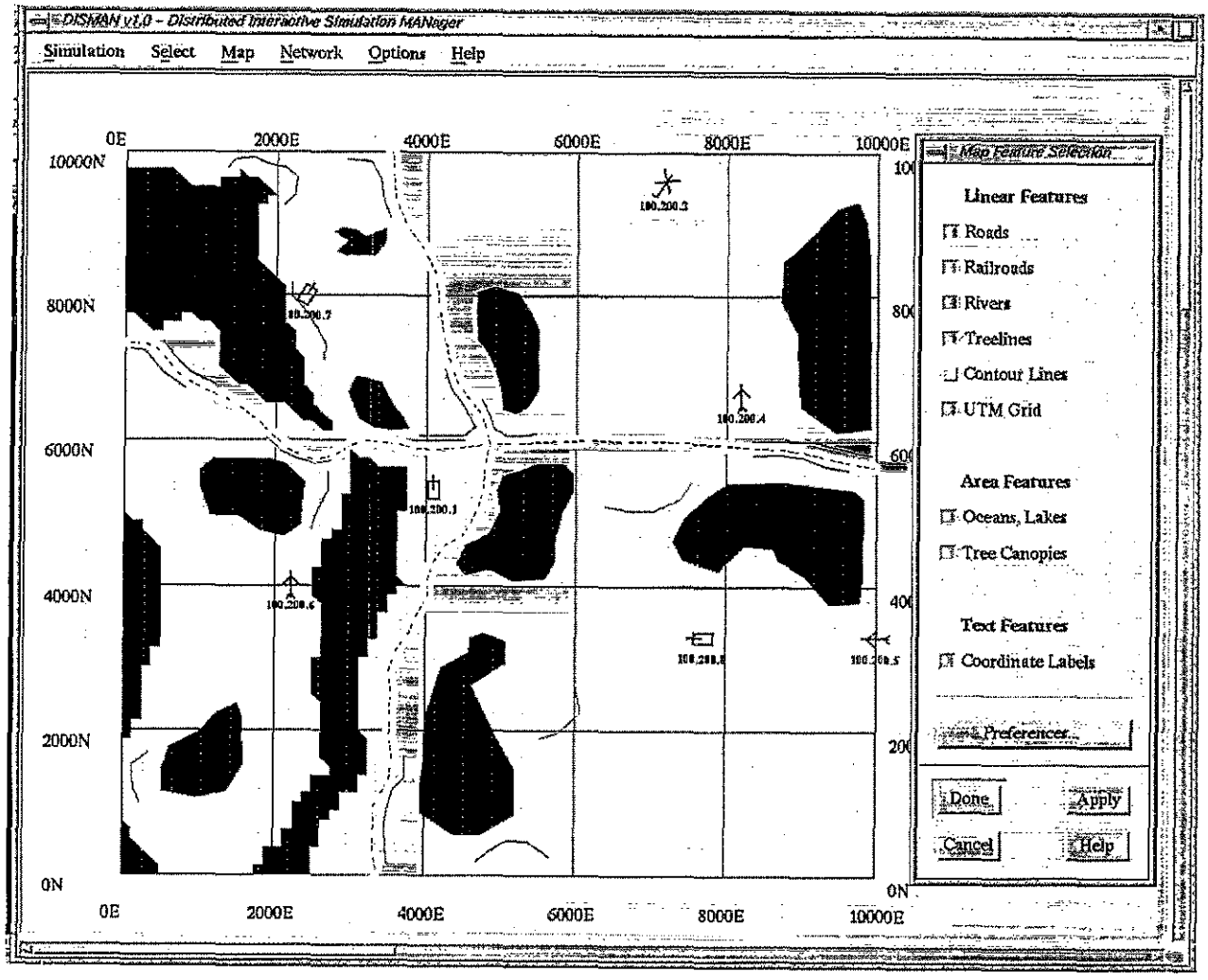

Figure 3.0 DISMAN's Plan-View Display

assembly layer performs the construction and decomposition of the SIMAN PDU. Data inputs from the application layer will be assembled into a SIMAN PDU structure as defined in the DIS 2.0.3 standard. Similarly, the assembly layer will disassemble a DIS PDU (SIMAN and Entity State PDUs) from the network into an internal data structure for further processing by the application layer. The assembly layer provides the interface between the application and network layer via service access points. Finally, the network layer performs the functions necessary to support the communication protocol stack. The network layer provides the necessary communication protocols such as UDP/IP/Ethernet. The Internet protocol stack of UDP/IP/Ethernet follows the recommended Phase 0 DIS communication architecture [3].

The advantage of utilizing a layered approach is to provide a clear definition in functionalities and to encourage software modularity. The value added to by modularizing the software is the ability to break up the software design into manageable pieces that are loosely coupled with each other.

Software Approach

To provide maximum code reuse, the design of DISMAN employs modern software practices. Issues such as code modularity, strong prototyping, and

code readability are major criteria in the software approach. With the tremendous advances in today's computing hardware, care is being taken in the software approach to promote code portability and reusability between computer systems. Software designs that effectively alienate the software solution from the hardware to the maximum extent possible can reduce the cost of maintaining the software across different computing platforms. Software that is designed to be extensible, and capable in absorbing changes to requirements such as the changes resulting from the evolution of the DIS standard is highly desirable.

DISMAN is written in ANSI-C on a Silicon Graphics Indy workstation. The software approach employs standard Unix networking libraries. The socket system call is used to obtain a socket descriptor for performing networking tasks. Standard Interprocess Communication (IPC) libraries are utilized to provide communications between processes. DISMAN is divided into four concurrent processes:

1) Low-Level I/O
2) Protocol Translator
3) Entity Update
4) Graphical User Interface

Figure 4.0 illustrates the four processes that reside in the DISMAN software.

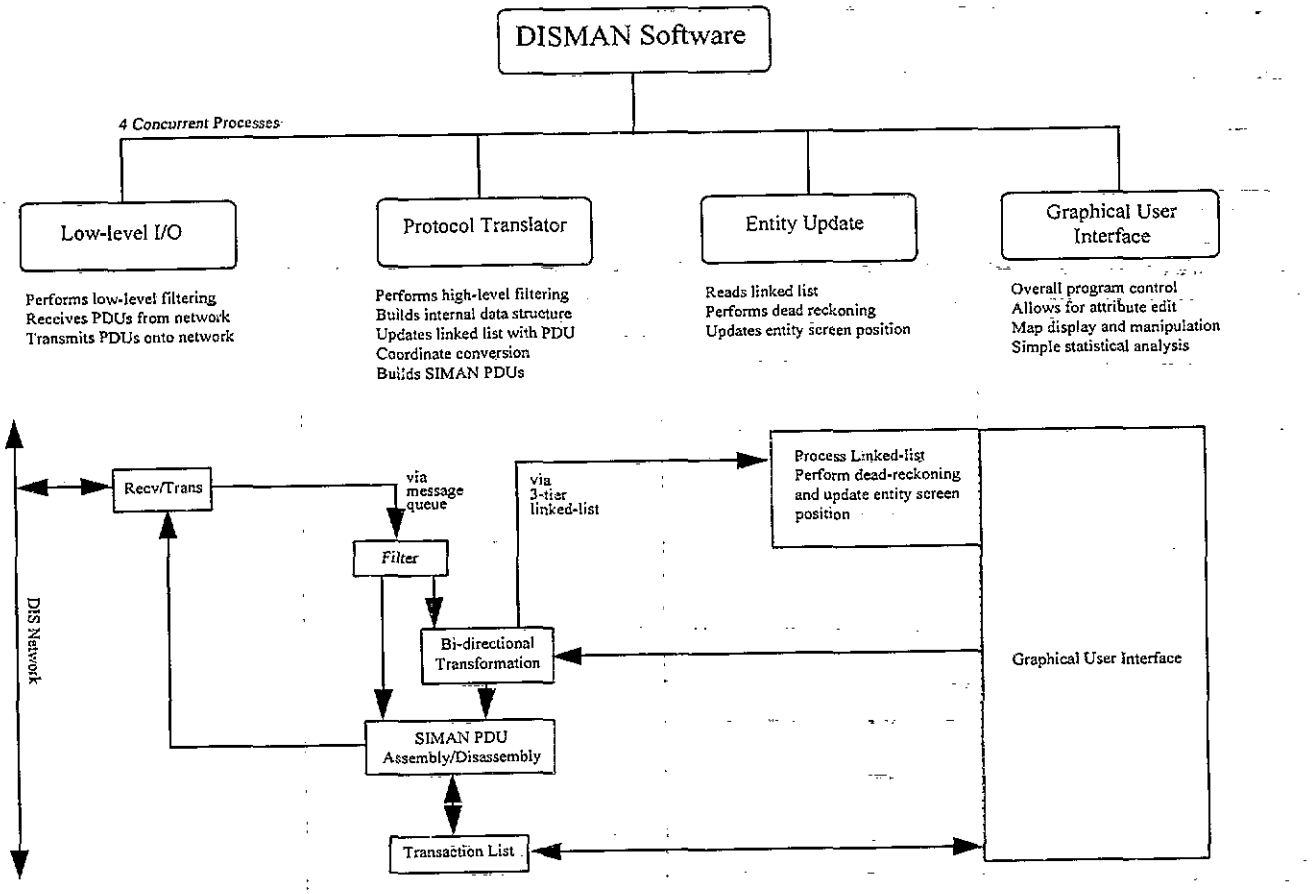

Figure 4.0 DISMAN Software Architecture

The Low-Level I/O performs filtering of PDUs based on the port identification number (DIS uses UDP port number 6993). Any data received other than DIS data will be discarded without further burden to the CPU. On the output side, when DISMAN transmits a SIMAN PDU, the PDU will be encapsulated by a UDP/IP/Ethernet frame before transmission onto the network.

The Protocol Translator process performs several functions such as high-level filtering, bidirectional transformations, updating the DISMAN internal link list, and builds the appropriate SIMAN PDUs. The most CPU intensive operation in the Protocol Translator process is the bidirectional transformation between DIS PDUs and internal DISMAN data structure format. Once the data are translated, they are entered into a three-tier linked list data structure for the Entity Update process. Similarly, in translating from a DISMAN data structure into a SIMAN PDU, the reverse process in coordinate conversion is applied.

The Entity Update process traverses the three-tier linked list data structure and performs dead reckoning and entity screen position updates on each item. The advantage of using a three-tier linked list is to provide another level of filtering based on entity identification (site, application, entity) values. DISMAN can selectively group the entities based on any combination of site, application and entity identities. Figure 5.0 illustrates the three-tier linked list data structure utilized in DISMAN.

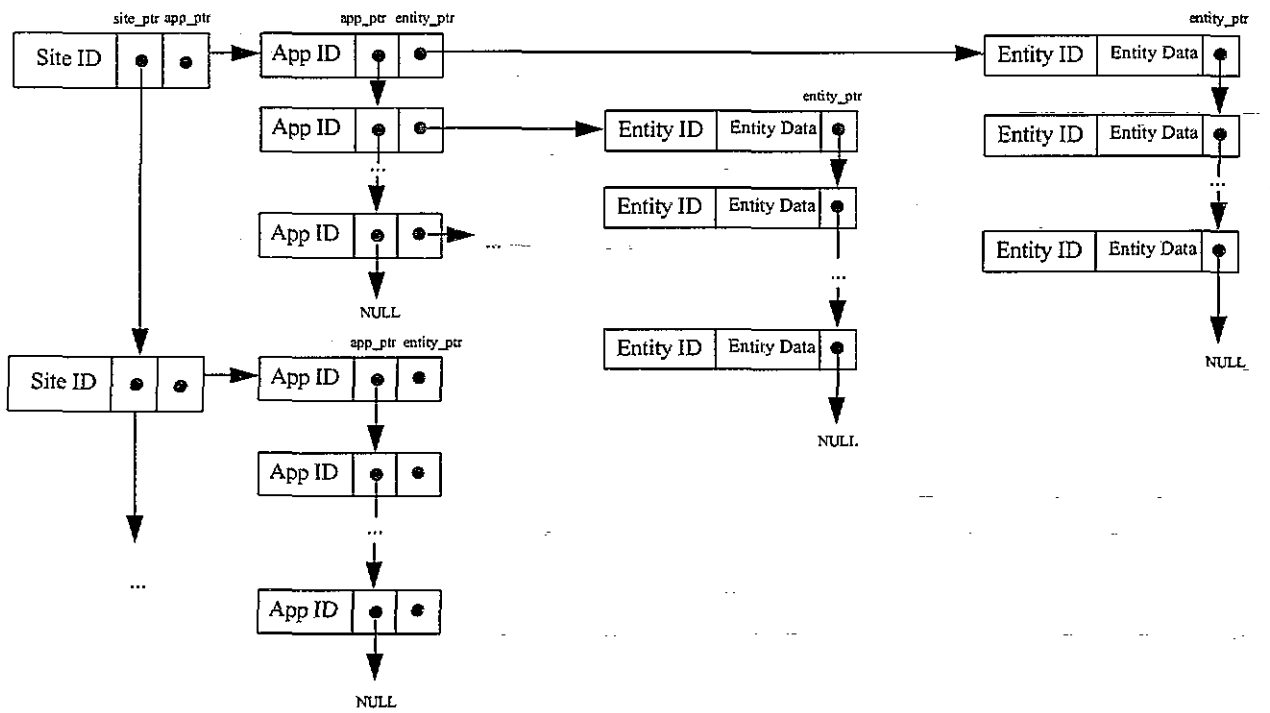

Figure 5.0 The Three-Tier Linked List Data Structure in DISMAN

The Graphical User Interface (GUI) process provides the user with a front-end system to perform overall program control, attribute edits, map display, and statistical analysis. The GUI is written in C utilizing standard X/Motif graphical libraries. Map control features such as contour lines, tree lines, tree canopies, rivers, UTM grid lines, panning, zooming are provided to the user. Pull down and pop-up menus are provided along with the map to enter entity attributes. The GUI provides the user an intuitive way in which to run DISMAN without the need to read software manuals or to understand the SIMAN PDU formats.

CONCLUSION

The DISMAN software utilizes standard software libraries to promote code reusability across different computational platforms. Modular software coding practices are employed in the software approach to allow ease in changes and add-on features. With the DIS standard still evolving, changes to DISMAN enumerated datum list will occur in the near future. DISMAN can be easily modified to meet these changes due to the modularity of the design. By providing the user with a simple-to-use GUI feature, DISMAN hides the details of the SIMAN PDUs, yet equips the manager with a powerful management tool.

The DISMAN program has provided the DIS community with a simulation management tool. By automating the managerial tasks by utilizing the 12 SIMAN PDUs, DISMAN has provided the manager with a systematic approach to simulation management. DISMAN will be used in the first A^2ATD demonstration to control exercise scenario. As of the date of this paper, testing of the DISMAN is still in process and the upper limits (number of entities) that DISMAN can control have not been determined. A^2ATD will be the first experiment which incorporates DISMAN as the simulation manager/controller. In A^2ATD, DISMAN will be used to control manned simulators, such as the M1A2. The next series of planned experiments will test DISMAN with Semi-Automated Forces (SAF) systems such as STRICOM's ModSAF (Modular Semi-Automated Forces) and CGF (Computer Generated Forces). These tests will show the utility and necessity of simulation management tools in controlling manned and automated forces.

LIST OF ACRONYMS

A^2ATD	Anti-Armor Advanced Tactical Demonstration
BDS-D	Battlefield Distributed Simulation - Developmental
CAS	Communication Architecture Subgroup
CGF	Computer Generated Forces
DARPA	Defense Advanced Research Projects Agency
DIS	Distributed Interactive Simulation
DISMAN	Distributed Interactive Simulation MANager
GUI	Graphical User Interface
IEEE	Institute of Electrical and Electronic Engineers
ISO	International Standards Organization
ModSAF	Modular Semi-Automated Forces
OSI	Open Systems Interconnection
PDU	Protocol Data Unit
PVD	Plan-View Display
SAF	Semi-Automated Forces
SIMAN	SImulation MANagement
SIMNET	SIMulation NETwork
SM	Simulation Manager
STRICOM	Simulation, Training and Instrumentation Command
UTM	Universal Transverse Mercator

BIBLIOGRAPHY

[1] "Proposed IEEE Standard Draft Standard for Information Technology - Protocols for Distributed Interactive Simulation Applications, Version 2.0, Third Draft", *Technical Report IST-CR-93-15,* Institute for Simulation and Training, University of Central Florida, May 1993.

[2] "Enumeration and Bit Encoded Values for Use with Protocols for Distributed Interactive Simulation Applications", *Technical Report IST-CR-93-19,* Institute for Simulation and Training, University of Central Florida, June 1993.

[3] "Communication Architecture for Distributed Interactive Simulation (CADIS)", *Technical Report IST-CR-93-20,* Institute for Simulation and Training, University of Central Florida, June 1993.