EVALUATING THE OVERHEAD OF OSI STACKS IN INTEROPERABLE DIS NETWORKS

Margaret Loper- Institute for Simulation and Training
M. A. Bassiouni- Department of Computer Science
G. Bulumulle- Institute for Simulation and Training
Ming Chiu- Department of Computer Science

University of Central Florida
Orlando, Florida

ABSTRACT

OSI (Open Systems Interconnections) communication stacks can be used to interconnect heterogeneous DIS machines and eliminate their incompatibilities. However, the interoperability benefit of OSI stacks could be offset by the computational overhead associated with the complex data transformation process of OSI upper layers. It is feared that an OSI implementation utilizing the transformation process would be too slow to meet the real-time requirements of DIS networks. In this paper, we present the results and conclusions of a detailed performance evaluation study which we have recently conducted to measure the overhead of the OSI transformation process, assess its impact on the delay encountered by DIS PDUs, and evaluate the benefits of using lightweight transfer syntax implementations.

ABOUT THE AUTHORS

Margaret Loper received her masters degree in Computer Engineering from the University of Central Florida in 1991. She is currently a Systems Engineer for the Institute for Simulation and Training, Orlando. Ms. Loper has been actively involved in research on simulation networking, OSI protocols, and multicast communications. She is an active member of the DIS Communication Architecture and Security Subgroup and is also a member of ANSI working groups on OSI Upper Layers and Network & Transport Layers.

M. A. Bassiouni received his Ph.D. degree in Computer Science from the Pennsylvania State University in 1982. He is currently an Associate Professor of Computer Science at the University of Central Florida, Orlando. He has been actively involved in research on computer networks, distributed systems, concurrency control, and relational databases. Dr. Bassiouni is a member of IEEE and the IEEE Computer and Communications Societies, the Association for Computing Machinery, and the American Society for information Science.

G. Bulumulle received his masters degree in Computer Engineering from the University of Central Florida in 1992. He is currently an Associate Engineer at the Institute for Simulation and Training and is involved in research related to DIS networks.

M. Chiu received his masters degree in Computer Science from the University of Central Florida in 1991. He is currently a Ph.D. student and Research Assistant in the Department of Computer Science, UCF.

EVALUATING THE OVERHEAD OF OSI STACKS IN INTEROPERABLE DIS NETWORKS

M. Loper, M. Bassiouni, G. Bulumulle, and M. Chiu

University of Central Florida
Orlando, Florida

INTRODUCTION

In large scale distributed interactive simulation (DIS) systems[6,10], various heterogeneous computing nodes are used as vehicle simulators and as control and data logging elements. Today, there are two great standards shaping the architectural principles and the technology of connecting large number of heterogeneous computing machines, namely, the OSI (Open Systems Interconnections) reference model and the Internet protocol suite (colloquially known as TCP/IP). The OSI model uses the principle of layered architecture in a more rigorous way, but the efficiency of the initial OSI implementations has been traditionally lower than that of TCP/IP. Both standards have similar lower-level communication (Physical and Data Link) layers that can employ Ethernet, token ring, FDDI, and other LAN/WAN protocols. Notable differences exist in the Transport layer of the two standards (i.e., the OSI transport protocol and TCP). Some of the functions of the Transport layer include: segmentation and reassembly of user messages, routing and flow control, recovery from data loss, and congestion avoidance. The OSI model defines three distinct layers above the Transport layer. These are the Session layer (which organizes the structure of the message exchanges), the Presentation layer (which allows a mutually acceptable transfer syntax to be established between the communicating entities), and the Application layer. The Internet protocol stack does not have this upper layer structure; rather, the functions of the Sessions and Presentation layers are built into the Application layer as needed. The placement of the implementation of the *transfer syntax* as well as the common abstract language upon which it is based (e.g., ASN.1, XDR, Xerox's Courier) represent only one of the differences between OSI and Internet.

OSI-compliant communication stacks can be used to interconnect the heterogeneous DIS machines and eliminate their incompatibilities as will be explained shortly. OSI, or the corresponding Government mandate (GOSIP), is a network protocol architecture consisting of seven layers. The upper layers of OSI are the place to implement any common syntax for OSI-compliant networks. Specifically, the OSI Standards place the functionality of the transfer syntax in the Presentation layer of the OSI stack, and make it a selectable feature that is not required for compliance. OSI upper layers therefore may perform a complex transformation step which produces a common transfer syntax for the exchanged messages. The interoperability benefit of this aspect of the OSI stack is, however, offset by the computational overhead associated with producing and managing the common transfer syntax. It is feared that an OSI implementation utilizing the common transfer syntax for DIS networks would be too slow to meet the real-time requirements of interactive training and would therefore degrade the realism of the training exercise. The objective of this paper is to present the numerical results and conclusions of a detailed performance evaluation study which we have recently conducted to measure the overhead of the OSI transfer syntax, assess its impact on the delay encountered by DIS PDUs, and evaluate the benefits of using lightweight transfer syntax implementations.

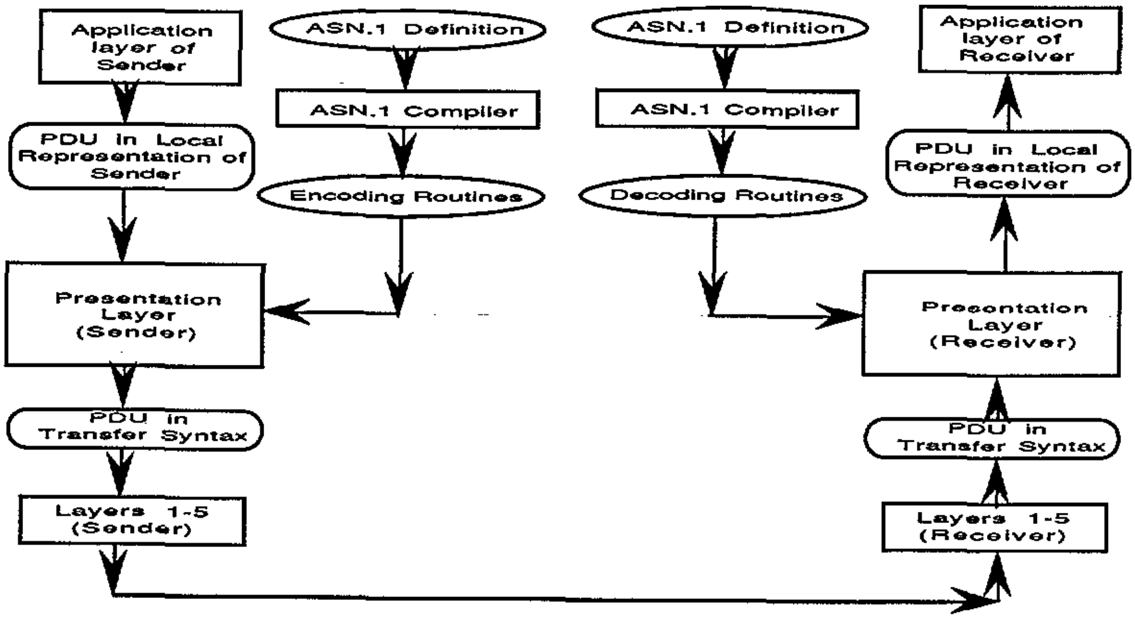
The performance experiments were conducted using the DIS/OSI Testbed at the Institute for Simulation and Training. The OSI stack was provided by the ISO Development Environment (ISODE)[7,8], a widely used suite of software primarily designed for fast implementation and testing of OSI upper-layer protocols. Using ten PDU types of the DIS Standards, our tests enabled us to evaluate the throughput delay encountered by a DIS PDU with and without the OSI transfer syntax overhead. The ten PDU types used in our tests, from DIS Version 1.0[10], are:

1) Entity State
2) Fire
3) Detonation
4) Collision
5) Service Request
6) Resupply Offer
7) Resupply Received
8) Resupply Cancel
9) Repair Complete
10) Repair Response

For each PDU type, several experiments were performed to compute various performance measures (e.g., the average value, the standard deviation, and the coefficient of variation of the end-to-end delay, the degradation ratio due to the overhead of the OSI Presentation layer, etc.). Consistent results have been observed when the tests were repeated using four different hardware platforms. In the following sections, we describe the OSI/DIS performance experiments and present the performance results and conclusions.

CHARACTERIZATION OF OSI/ASN.1 OVERHEAD

Using OSI/ASN.1, two dissimilar DIS nodes can exchange protocol data units (PDUs) as illustrated by the following diagram.


Application layer of Sender
ASN.1 Definition
ASN.1 Compiler
Encoding Routines
PDU in Local Representation of Sender
Presentation Layer (Sender)
PDU in Transfer Syntax
Layers 1-5 (Sender)
ASN.1 Definition
ASN.1 Compiler
Decoding Routines
Application layer of Receiver
PDU in Local Representation of Receiver
Presentation Layer (Receiver)
PDU in Transfer Syntax
Layers 1-5 (Receiver)

Basically, the exchange mechanism is carried out as follows:

1) The PDU is first transformed from its local representation at the sending host to a transfer syntax using a set of

transformation rules called the *encoding rules.*
2) The PDU in transfer syntax is transmitted down the communication stack of the sending host, and is delivered in the same transfer syntax to the receiving host.
3) The PDU in transfer syntax is transformed to the local representation of the receiving host using a set of rules called the *decoding rules.*

For OSI-compliant networks[3], two standards have been so far proposed and adopted by ISO/ANSI: i) The *Abstract Syntax Notation One* (ASN.1)[2,4,9] is used to solve the problem of heterogeneous local representations of data, and ii) ASN.1 *Basic Encoding Rules* (BER)[5] are a set of encoding rules used to produce a transfer syntax for the exchanged data based on ASN.1. Consider for example the communication of a simple PDU between two machines: the sending machine A encodes characters in ASCII and uses a 2's complement scheme for integers; the corresponding representations in the receiving machine B are EBCDIC and 1's complement. The PDU transmitted by the application layer of machine A has two fields: an integer of value -5, represented in 2's complement, and an octet string of value "USA", represented in ASCII. For each of the two fields, the BER code in the Presentation layer of machine A generates a sequence of three components: 1) unique tag, 2) length identifier, and 3) the value of the field represented in a common transfer syntax. These Tag-Length-Value (T-L-V) sequences are transmitted down the stack of machine A and ultimately received by the Presentation layer of machine B. The BER decoding routines in machine B uniquely decipher the T-L-V sequence of each field and then passes the value -5 in 1'complement and the string "USA" in EBCDIC to its application layer.

The following are the different types of time overhead incurred by the implementation of OSI/ASN.1.

a) Encoding Overhead (EO): which is the time needed for the execution of the BER encoding routines.

b) Sender Processing Overhead (SPO): which is the extra processing time (excluding the encoding overhead) in layers 1 through 6 due to the representation of data in transfer syntax.

c) Decoding Overhead (DO) and Receiver Processing Overhead (RPO): these are defined analogously to EO and SPO.

d) Total Time Overhead (TTO): which is the sum of the above components. In the DIS environment, TTO represents the extra end-to-end delay encountered by a DIS PDU when the OSI transfer syntax is introduced.

PERFORMANCE EVALUATION EXPERIMENTS/RESULTS

To assess the impact of using OSI in DIS communication networks, several experiments were conducted using ISODE[7,8]. The following is a high-level description of these experiments.

The Isolation Model

The purpose of this experiment is to compute the encoding and decoding overhead, EO and DO, associated with OSI in DIS simulators.

The Network Model

In this experiment, measurements are taken with respect to the end-to-end delay between two hosts and the total time overhead TTO is recorded.

Table 1 gives the average end-to-end delay in milliseconds for each DIS PDU type with and without the overhead of OSI/ASN.1. Identical Sparc machines were used both as sender and receiver. The column labeled "without transfer syntax" gives information about the end-to-end delay encountered by a PDU when it is transmitted between two hosts without invoking the OSI/ASN.1 encoding or decoding routines (i.e., the PDU is treated like a single stream of binary data which is transmitted without transformation). Each entry in Table 1 was obtained by transmitting the same PDU sixty times and computing the average value of the end-to-end delay and the corresponding coefficient of variation, denoted C.o.V., which is obtained by dividing the value of the standard deviation over the average value. The 60 samples used in computing the average delay were found to be statistically sufficient for obtaining accurate results (care was taken to avoid sampling the initial few transmissions in which higher delay is observed due to the cost of connection set-up). We also define the degradation ratio as the ratio between the increase in the average delay due to OSI/ASN.1 and the original average delay (without OSI/ASN.1). Figure 1 shows the histogram of the degradation ratio for two different hardware configurations: the first configuration uses only Sparc machines and the second uses only Motorola machines.

Table 1. Impact of OSI transfer syntax on the average end-to-end delay

PDU Type	without transfer syntax		with transfer syntax	
	Avg.	C.o.V.	Avg	C.o.V.
1	8.193	0.022	24.395	0.022
2	7.609	0.016	18.815	0.033
3	8.178	0.021	25.078	0.017
4	7.533	0.037	15.023	0.038
5	7.454	0.051	13.634	0.031
6	7.385	0.026	13.831	0.229
7	7.472	0.025	18.274	0.029
8	7.345	0.029	10.154	0.012
9	7.357	0.027	10.642	0.035
10	7.370	0.032	10.649	0.029

LIGHTWEIGHT OSI IMPLEMENTATIONS

In this section, we present our ongoing analysis and evaluation of light-weight OSI implementations, e.g., the skinny enveloping scheme. The latter approach is based on limiting the functionality of the OSI transfer syntax implementation to what is needed by the application and eliminating unused features. In the Presentation and Session layers, the approach works by pre-coding invariant octet-sequences for outbound messages. At the receiving end, the inbound messages are matched against the invariant octet-sequences for direct access of relevant data. If the match fails, the expensive process of general parsing (e.g., ASN.1 parsing) of the incoming octets is performed. In the best scenario, the match would be all what is needed to handle the envelope carrying the wrapped data. The skinny stack doctrine does not provide any guidelines regarding which fields can be encoded as "invariant octet sequences" or which fields can be ignored; the choice is basically application dependent. In the DIS Entity State PDU, for example, one may consider the fields : "protocol version" and "exercise-id" as invariant values (since the same value is used throughout one training exercise). Some or all of the various padding fields, and other entity dependent fields (e.g., "country", "category", "domain", etc.) may be practically ignored since they are not needed in every ESPDU transmission. Our performance tests were also used to determine the level of improvement that can be achieved when all nodes of a DIS network use a skinny OSI implementation. For each PDU type, experiments were performed to determine the end-to-end delay for the skinny enveloping case as well as for the general parse (full stack) counterpart. Our tests were executed on four different "sending host/receiving host" hardware configurations denoted by S/S, M/M, S/M, and M/S where S stands for a Sparc machine and M stands for a Motorola machine. Figure 2 shows the values of the end-to-end delay (in milliseconds) for the ten DIS PDUs using the S/S configuration.

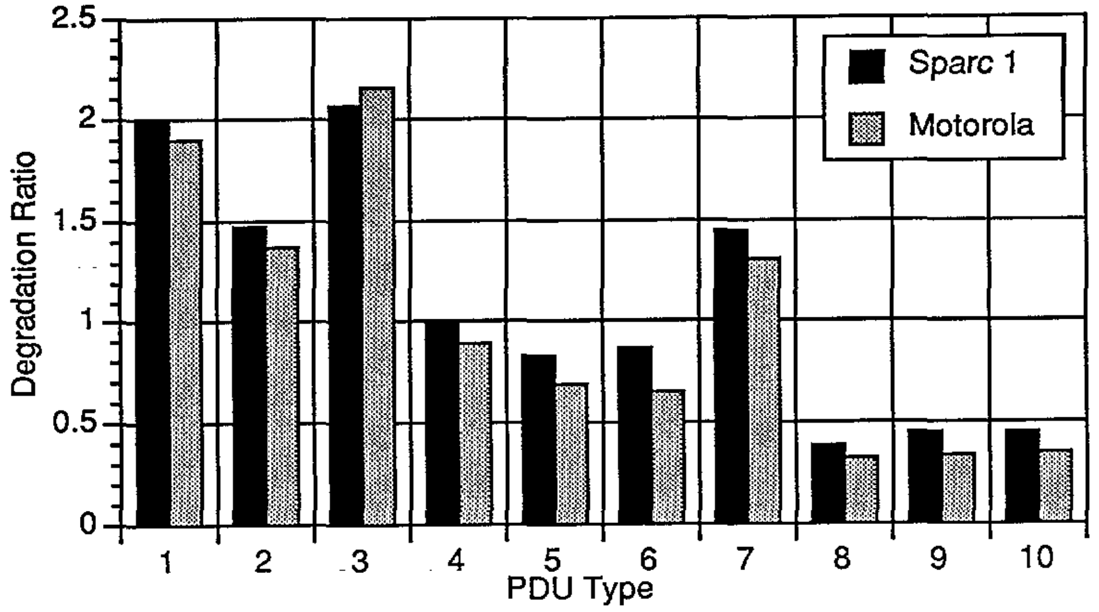


Figure 1. Degradation ratio for two configurations

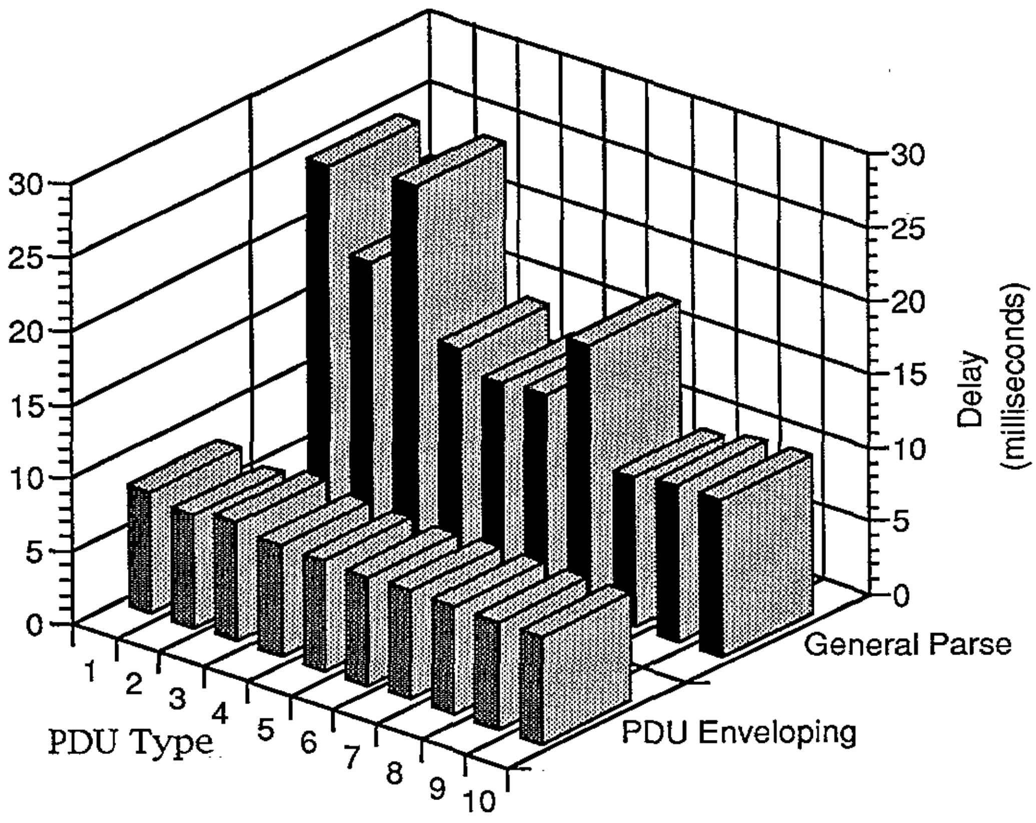


Figure 2. End-to-end delay for the *S/S* configuration

Figure 2 also shows that unlike the case of PDU enveloping of the skinny stack, the general parse of the full stack exhibits significant variations in the value of the end-to-end delay among the different PDUs. Furthermore, the speed-up of the skinny enveloping approach has been found to depend on the composition of the actual data transmitted. To help analyze the numerical results, and to better understand their implications, we shall introduce a simple metric that reflects the complexity of the different DIS PDUs. Table 2 shows the composition of these PDUs based on the description of their contents in the DIS Standards.

Table 2. Composition of PDU types in the DIS application

PDU Type	# integers	# real values	# octet strings
1	42	9	6
2	27	7	1
3	39	9	2
4	13	7	2
5	18	1	2
6	17	1	2
7	17	1	2
8	9	0	1
9	10	0	2
10	10	0	2

Let

I_k = the average number of integers in a PDU of type k

R_k = the average number of real values in a PDU of type k

S_k = the average number of octet strings in a PDU of type k

and define C_k to be a metric for the complexity of processing (e.g., parsing) a PDU of type k. A simple choice of C_k is the following linear relation:

$$C_k = a*I_k + b*R_k + c*S_k$$

where a, b, and c are constants. The bubble chart of Figure 3 shows the relationship between the speed-up and the complexity of the PDU for the Sparc hardware (the corresponding results for the Motorola hardware are quite similar and are not given in this paper). The chart contains a bubble for each PDU type such that the size of the bubble is proportional to the complexity metric of the corresponding PDU type (assuming a=1, b=4, and c=4). The value of the vertical displacement (Y-axis) of the center of the bubble is equal to the speed-up achieved by the enveloping scheme. The speed-up is defined to be the ratio between the end-to-end delay of the DIS PDU using a full stack and the corresponding end-to-end delay using the skinny enveloping scheme. In general, the larger the size of the bubble, the higher the corresponding speed-up value. The choice a= 1, b=4, and c=4 in Fig. 3 to represent the complexity of integer, real, and string variables, respectively, was simply made based on the size we expected for these variables (many integers in DIS PDUs are short integers of size one byte; a real value is usually encoded in four bytes; and many string fields are used for padding and are of size four bytes). We have also experimented with other reasonable choices of a, b, and c. The results were not significantly different from those presented in the paper. Figure 4 shows the values of the speed-up for the four different hardware configurations. The minimum speed-up in Figure 4 has a value of 1.28 and corresponds to the Repair Complete PDU in the S/M configuration. The maximum speed-up has a value of 3.43 and corresponds to the Detonation PDU in the M/S configuration. Notice that the most frequent PDU in DIS (namely, the Entity State PDU) suffers from a very high ASN.1 overhead and would therefore benefit the most from lightweight transfer syntax implementations. Finally, it should be noted that the simple complexity metric derived from Table 2 didn't differentiate between integers and short integers and didn't take the length of individual octet strings into account. Using more sophisticated metrics is a topic that is worthy of further investigation.

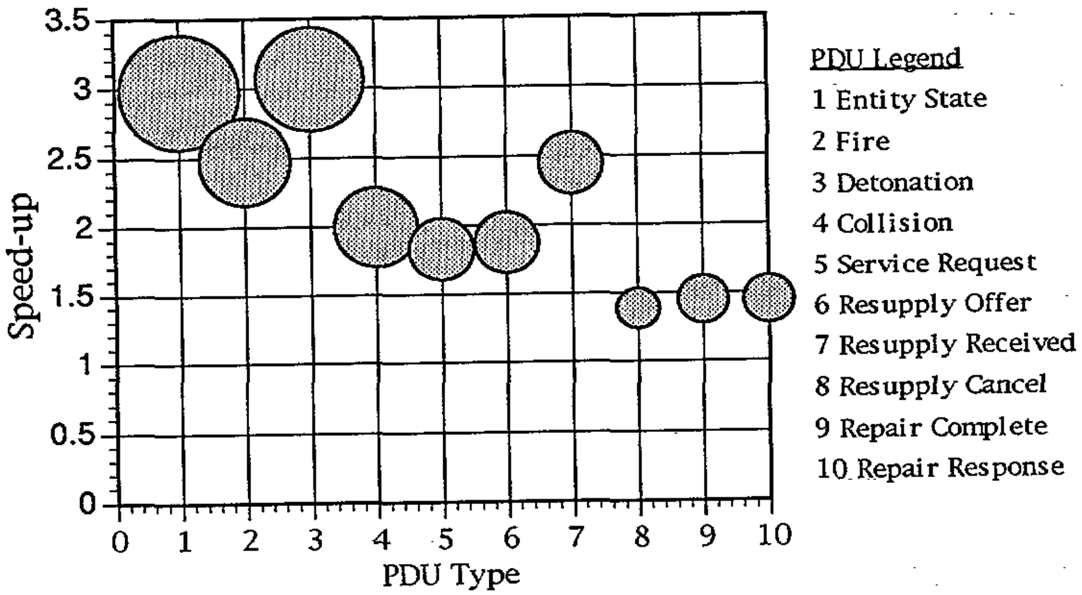


Fig. 3. PDU complexity vs. speed-up
(Note: bubble size is proportional to complexity)

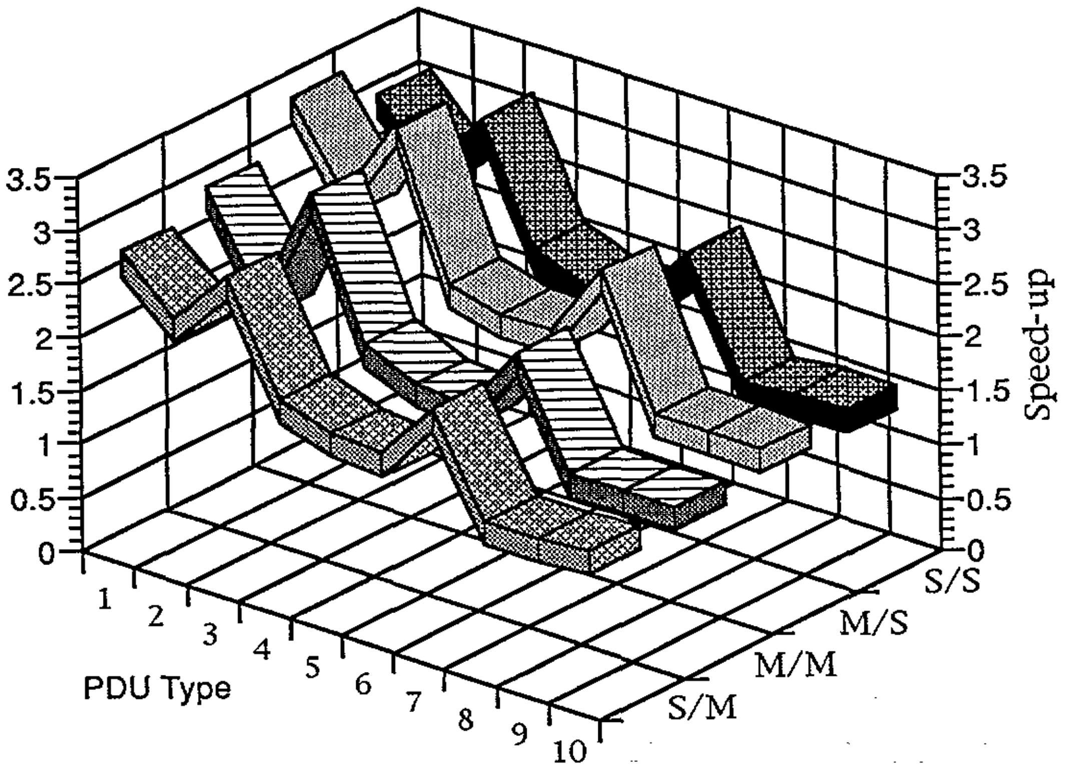


Fig. 4. Speed-up values for four hardware configurations

The range chart of Figure 5 gives the average speed-up value for each PDU (the average is computed over the four hardware configurations). The minimum and maximum values of the speed-up are also depicted.

In addition to the ten DIS PDUs discussed earlier, the Network Model was also used to measure the end-to-end delay of a PDU consisting of a sequence of m integers (as was done in previous work[11]). Figure 6 plots the relationship between m and the average end-to-end delay (in milliseconds using Sparc machines). The delay shown in Figure 6 is the overall delay encountered by a packet when traveling from one host to the other.

In general, the end-to-end delay was found to closely follow the linear equation

$$d = c_0 + c_1 * m$$

where c_0 and c_1 are constants, d is the delay in milliseconds, and m is the number of integers in the PDU ($c_0 = 7.582$ and $c_1 = 0.205$ for the Sparc hardware used in Figure 6). The corresponding delay, d', without invoking the OSI/ASN.1 routines can also be approximated by a linear equation

$$d' = c_0 + c_2 * m$$

where c_2 is a constant whose value is orders of magnitude smaller than that of c_1. A good approximation of the OSI/ASN.1 overhead TTO can therefore be obtained as follows

$$\begin{aligned} TTO &= d - d' \\ &\approx c_1 * m \end{aligned}$$

In other words, the relationship of TTO versus m is similar to that shown in Figure 6, but shifted vertically by the value c_0.

CONCLUSIONS

In this paper, we presented the results of our performance evaluation experiments to measure the interoperability overhead of the OSI transfer syntax in DIS networks. The tests showed that the end-to-end overhead of OSI's ASN.1 is significant and can therefore compromise the proper operation of the DIS application. Our experiments also gave preliminary insight into the possible speed-up of the lightweight skinny approach. The tests showed a speed-up of up to 3.4. In most

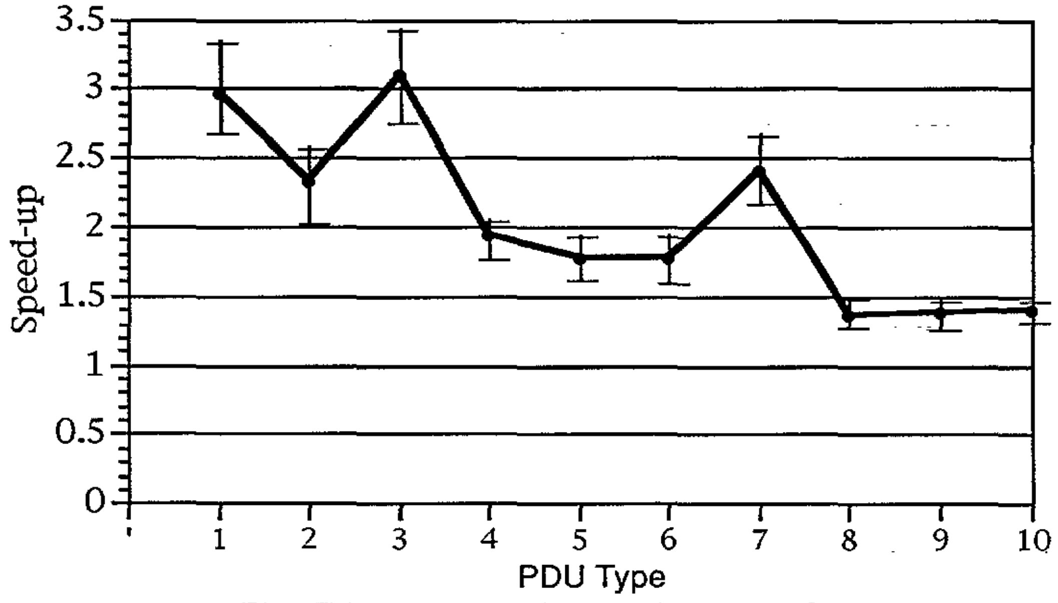


Fig. 5. Average value and range of speed-up

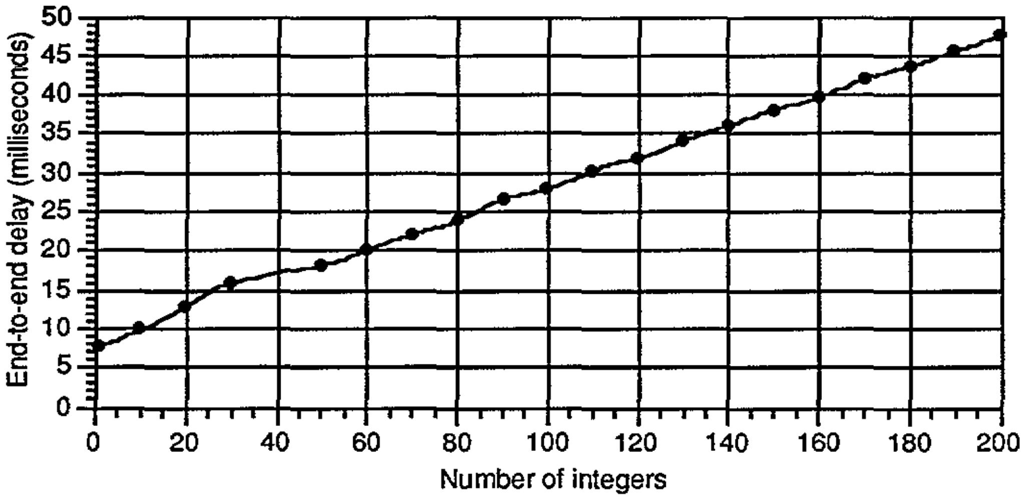


Figure 6. End-to-end delay for a PDU vs. number of integers

PDUs and hardware configurations, the speed-up is well below 3 implying that the skinny enveloping scheme in DIS is at most three times faster than the full stack. The minimum speed-up observed in our tests is 1.28. Although an actual skinny implementation for the DIS application may differ from the set-up used in our tests, the results reported in this paper clearly show that there is an evident need to develop and standardize lightweight OSI-compliant networks.

ACKNOWLEDGMENT

This work has been supported by U.S. Army's Simulation, Training, and Instrumentation Command (STRICOM) under contract N61339-91-C-0103.

REFERENCES

[1] Furniss, P. and Ashley, K. "The OSI skinny stack" Draft Report, University of London Computing Center, March 1992.

[2] Gaudette, P. "A tutorial on ASN.1" U.S. Department of Commerce, National Institute of Standards and Technology, Technical Report, NCSL/SNA-89/12, May 1989.

[3] ISO. Information Processing- Open systems Interconnection- Basic Reference Model. ISO International Standard 8072, 1983.

[4] ISO. Information Processing- Open systems Interconnection- Specification of Abstract Syntax Notation One (ASN.1). ISO International Standard 8824, 1990.

[5] ISO. Information Processing- Open systems Interconnection- Specification of Basic encoding Rules for ASN.1. ISO International Standard 8825, 1990.

[6] LORAL System Company. Strawman Distributed Interactive Simulation Architecture Document. Vol. I: Summary Description. ADST Program Office, March 1992.

[7] Rose, M. "ISODE: horizontal integration in networking" ConneXions, The Interoperability Report, Vol. 2, No. 3, March 1988.

[8] Rose, M. *The ISO Development Environment: User's Manual.* Volumes 1-5, Performance Systems International, Inc., January 1990.

[9] Steedman, D. *Abstract Syntax Notation One (ASN.1): The Tutorial and Reference.* Technology Appraisals Ltd. ISBN 1-871802-06-7, 1990.

[10] STRICOM: *Ditsributed Interactive Simulation- Standards Developement Draft 2.0*, September 1992.

[11] Neufeld, G. and Yang, Y. "The design and implementation of an ASN.1-C Compiler" IEEE Transactions on Software Engineering, Vol. 16, No. 10, 1990, pp. 1209-1220.