ASN.1 ENHANCEMENTS TO SUPPORT TACTICAL DATA COMMUNICATIONS

Christopher D. Bonatti
Booz·Allen & Hamilton Inc.

ABSTRACT

This paper presents methods of adapting the upper OSI layers for enhanced performance in a reduced bandwidth environment, such as that found in tactical military systems. The paper describes several techniques for reducing the encoding overhead imposed by the ASN.1 BER and examines several existing proposals for alternative encoding rules. The paper also illustrates the benefits of the new techniques with complex data structures.

Analytical methods were used to determine the extent of the ASN.1 BER overhead for complex PDU structures. To provide a representative level of complexity, various X.400 APDU structures were used. The BER encoding overhead for an X.400 message includes tag octets, length octets, and pad bits for a complete P3 APDU.

As might be expected, the amount of encoding overhead varies considerably with the size of the message content. For small message sizes (i.e., fewer than 10,240 octets), such as those used for tactical C^3, the amount of overhead is at least 11 percent. For smaller messages the overhead is much greater. For 5120 octet messages the overhead figure is approximately 30 percent. For a 2048 octet message, which is not unreasonable in the tactical environment, the encoding overhead exceeds 60 percent.

These figures should not be confused with X.400 protocol overhead, which consists of data carried in protocol fields, and typically adds 200 to 300 percent overhead to user-supplied message bodies fewer than 10,240 octets in size. Intelligent profiling can reduce this figure significantly, but that is beyond the scope of this paper.

Several techniques can be used to minimize encoding overhead including: tag and length octet omission, intelligent

ordering of fields in the abstract syntax, nesting omission, numeric range adjusting, and bit alignment.

This paper analyses the overhead savings achieved by using improved encoding techniques. Overhead estimates are given using the described techniques, including overviews of the PER, LWER, and MBER proposals. These estimates were derived using examples of PDUs of varying complexity drawn from the X.400 base standards.

To promote reduced encoding rules, several actions are recommended in the standardization community. The PER and LWER specifications are being progressed as CD 8825-2 and 8825-4, respectively. They should both soon be available in DIS form. These efforts should be monitored carefully and actively supported by national standardization bodies. A proposal incorporating bit alignment, such as MBER, should be introduced to ISO and considered as a long term solution.

1. INTRODUCTION

This paper examines sources of overhead in OSI* applications and proposes techniques for minimizing overhead due to ASN.1* encoding. Because of the low bandwidth typically available in tactical C^{3}* environments, overhead related to OSI services and protocols is a major concern. It is possible, however, to adapt the upper OSI layers to provide enhanced performance in reduced bandwidth environments.

This paper describes several techniques for reducing ASN.1 encoding overhead, analyzes the relative performance of those techniques, and provides an overview of several

---

*OSI: Open Systems Interconnection
ASN.1: Abstract Syntax Notation One
C^{3}: Command, Control and Communications

existing proposals that use these techniques in varying degrees.

2. TECHNICAL BACKGROUND

In the OSI reference model defined by IS* 7498-1 [Ref. 1], the presentation layer is allocated three important high-level functions. The presentation layer must ensure that communicating applications agree to common semantics for any data to be exchanged. The presentation layer is responsible for ensuring that transmitted information correctly conveys the agreed end-to-end semantics. The presentation layer is also responsible for selecting an optimum mutual transfer syntax for each association.

Two key OSI components that are available to perform the functions of the presentation layer are the presentation protocol and ASN.1. The presentation protocol provides mechanisms for negotiating both the abstract syntax used by the application and the transfer syntax used between presentation entities. An abstract syntax is a complex data structure that organizes information in a manner suitable to the application. A transfer syntax is the format used to transfer the application data via the session layer. Collectively, a particular abstract syntax and transfer syntax pair is referred to as a *presentation context*.

The set of presentation contexts supported by each OSI end system is defined during setup procedures. The presentation user (i.e., the application layer) specifies a set of presentation contexts that is supported for a

---

*IS: International Standard

particular association. This information allows the presentation protocol to negotiate a presentation context that is mutually acceptable for each instance of communication.

2.1 ASN.1 Overview

Different computer architectures use different internal representations of information. For the presentation layer to enable communication between end systems with different data representations, a conversion function must be employed. Generally, two architectural options are available: explicit format conversion and common format conversion. These options are characterized in Figure 1. Explicit format conversion employs a different conversion process for each different machine type it communicates with. This has the advantage of converting data only once, but it requires each end system to support conversion to each different machine format. The common format conversion approach uses a standardized transfer format and performs conversion to and from the local machine formats. This has the advantage of requiring only one type of conversion per end system, and also eliminates the need for end systems to have knowledge of the data representation used by a remote system. These considerations are paramount in an open systems environment. The only disadvantage to the common format approach is that two conversions are always applied to the data, even if two end systems use the same local data representation.

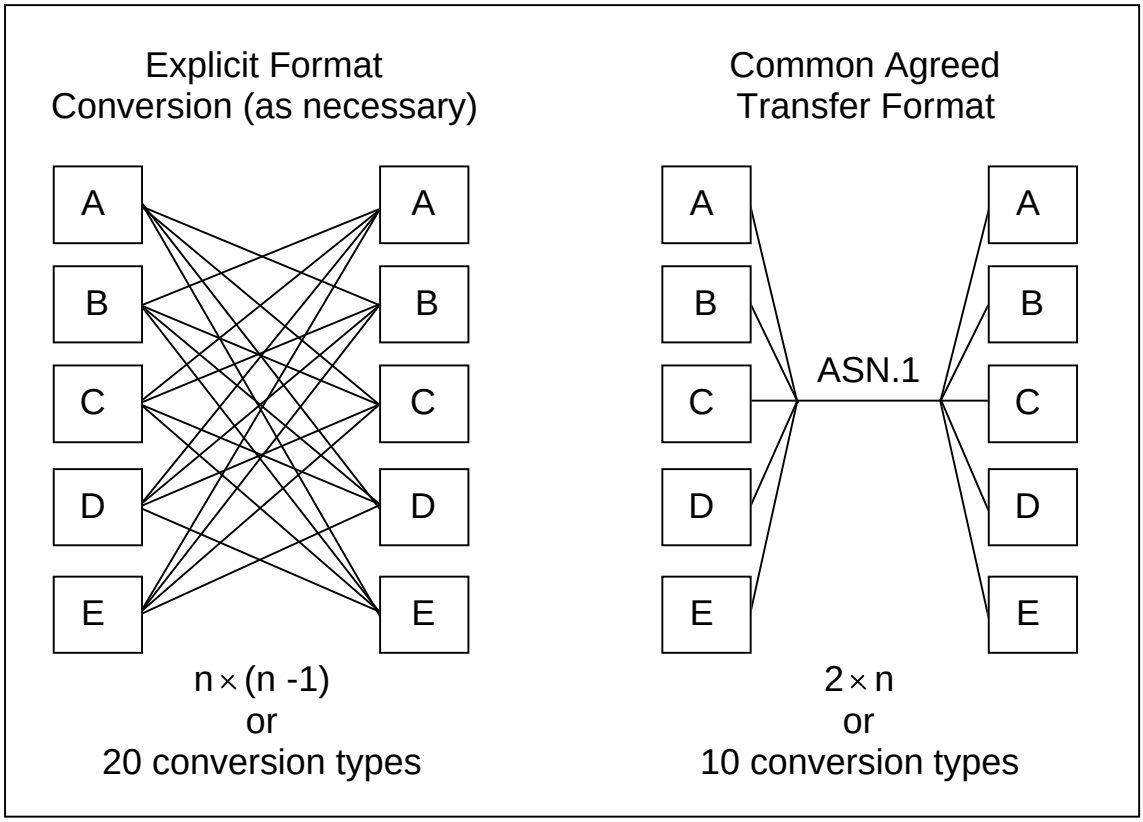

Figure 1 – Data Representation Conversion Alternatives

The ASN.1 standards, IS 8824 [Ref. 2] and X.208 [Ref. 3], were developed by the ISO* to provide a standardized method of defining the abstract syntaxes used by applications in a way that was independent of local data representation. The ASN.1 allows applications to agree to a common picture of the data structure, regardless of how the information is locally stored or processed. The notation includes provisions for specifying various primitive data types including boolean values, integer values, bit strings, octet strings, real values, and enumerated types. It also

---

** ISO: International Organization For Standardization*

provides mechanisms for defining constructed types, including several useful type definitions derived from the primitive data types.

2.2 BER Overview

The original ASN.1 standard was codeveloped with the BER standards, IS 8825 [Ref. 4] and X.209 [Ref. 5]. These encoding rules provided a standardized mechanism for representing complex data structures in a simple data stream suitable for transmission over a communications channel. The BER define this transformation by specifying octet-aligned encoding rules for each data element type in the abstract syntax. Rules were also given for forming compound encodings for ASN.1 constructed types.

For primitive data types, each BER encoding element is composed of three fields: a type identifier, a length indicator, and content octets. This type-length-value, or TLV*, structure is common to many data encoding techniques. The structure of a BER encoded type is shown in Figure 2.

Constructed data types may also be encoded using the TLV form, or they may use an indefinite length form. The indefinite length form uses a repeated "End of Content" octet to delineate the end of the encoding. Indefinite length encodings are particularly useful when data transmission must begin before the total content length is known.

---

* *TLV: Type Length Value*

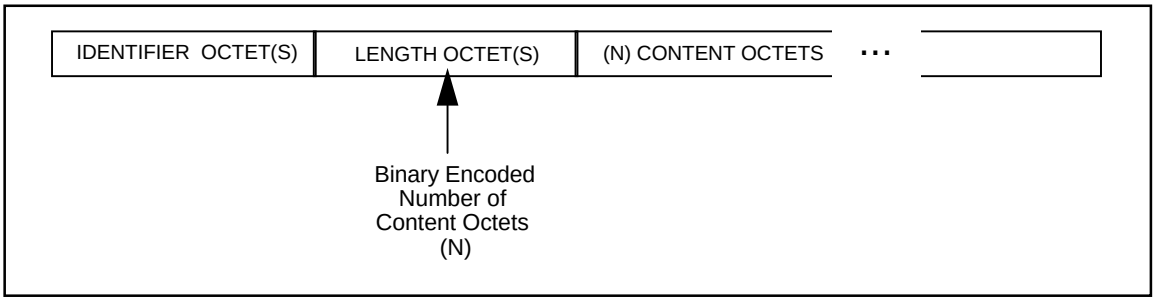

Figure 2 – Structure of BER Elements

The type identifier field contains an integer tag value, a 2-bit subfield that indicates the semantics of that value, and a 1-bit flag that indicates whether the type is primitive or constructed. There is no limit to the magnitude of the tag value that can be encoded in the type identifier field, but tag values less than or equal to 30 have the advantage of being encoded in a single octet. The four possible values of the 2-bit subfield each indicate one of the following classes for the tag value: universal, application, context-specific, or private. The universal class is composed of tag types defined in the ASN.1 base standard. Application tag types are those defined in other international standards. Context-specific tags are defined in a particular instance of abstract syntax and have no meaning outside that particular data structure. Private tag types are defined for a particular enterprise or implementation, and are never assigned within international standards.

Unlike the type identifier field, the length indicator field cannot accommodate an arbitrarily large value.

However, the maximum content length is so large as to be considered virtually infinite.

2.3 ASN.1 in an X.400 Context

The CCITT* X.400 [Ref. 6] and IS 10021 [Ref. 7] base standards use ASN.1 extensively for abstract syntax definition. Three major application layer protocols, P1, P3, and P7, are specified as exchanges of abstract syntax defined in ASN.1. The three content type structures defined in the base standards (i.e., P22, PEDI, and VMP*) are also specified using ASN.1. The structure of X.400 PDUs* is shown in Figure 3.

Although the protocol specifications of X.419 [Ref. 8] and IS 10021-6 [Ref. 9] avoid any mention of encoding rules for P1, P3, and P7, the definitions of the standard content types in X.420 [Ref. 10] and IS 10021-7 [Ref. 11] specifically require the use of BER. Because the negotiation of coding rules does not take place between two X.400 UAs*, the standards must specify the encoding of content types to ensure interoperability. As a consequence of this arrangement, some X.400 implementors have chosen to hard code the format of content types even when their protocol engine uses a more general purpose ASN.1 encoder.

** CCITT: International Telegraph and Telephone Consultative Committee*

VMP: Voice Messaging Protocol

PDU: Protocol Data Unit

UA: User Agent

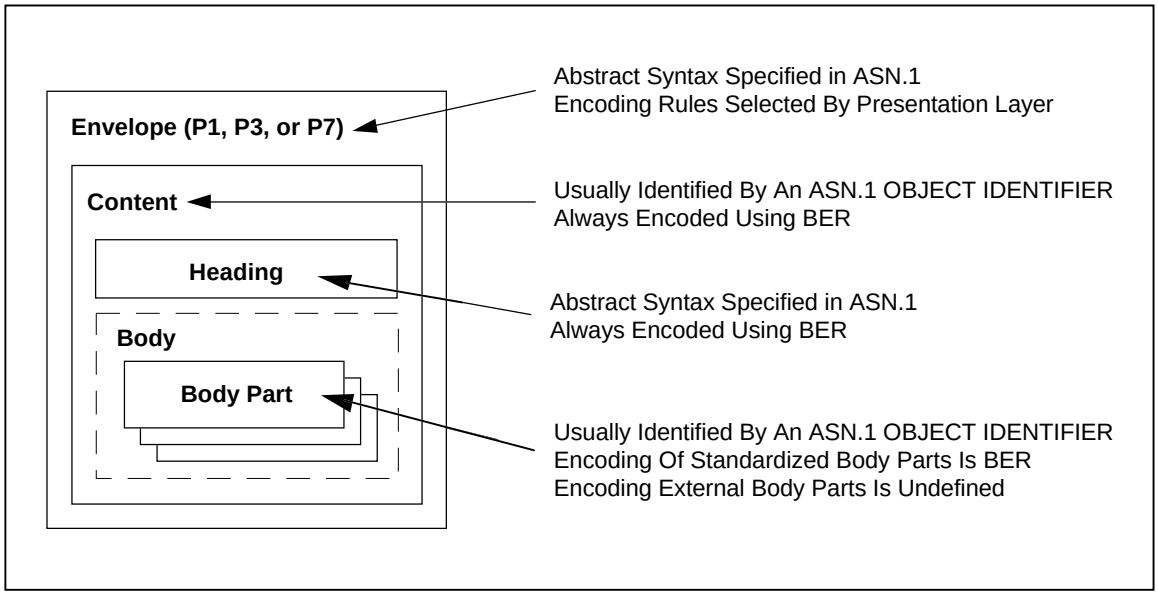

Figure 3 — Structure of X.400 PDUs

2.4 Recent Developments

The ISO has recently completed DIS* versions of revised ASN.1 and encoding rule standards. The new ASN.1 standard consists of multiple parts. Basic descriptions of the ASN.1 notation and guidelines about how to apply it are now covered in DIS 8824-1 [Ref. 12]. The three new parts, DIS 8824-2, DIS 8824-3, and DIS 8824-4 [Refs. 13, 14, and 15], provide mechanisms for defining object classes, exception values, and parameterization that are intended to replace the MACRO notation used in the second edition of IS 8824. The new DIS version also reduces the possibility of syntactic ambiguity by eliminating the ANY and ANY DEFINED

** DIS: Draft International Standard*

BY types and by making identifiers mandatory in SEQUENCE, SET, and CHOICE types.

The ASN.1 encoding rules standard has likewise been rewritten as a multipart document. The existing BER are now specified in DIS 8825-1 [Ref. 16]. The series has been extended to incorporate new alternative encoding rules. The new DIS 8825-3 [Ref. 17] defines DER* that use a TLV format that is similar to BER, but which eliminates the many alternative encodings that are possible with BER. The DER is useful in cases, such as cryptography, where independently generated encodings must be identical. The PER* are defined by CD* 8825-2 [Ref. 18] to provide a more compact form of TLV encoding that is faster to process. The WD* 8825-4 [Ref. 19] defines non-TLV LWER* that are much faster to process because they are designed to generate encodings similar to commonly used local storage formats. Both PER and LWER are expected to reach DIS status in the near future.

The set of different encoding types is intended to be open ended, and will therefore likely expand in the future. The civil aeronautical community has already proposed one such standard called MBER* [Ref. 20]. The presentation layer protocol is capable of ensuring continued interoperability as long as one common set of encoding rules

---

*DER: Distinguished Encoding Rules
PER: Packed Encoding Rules
CD: Committee Draft
WD: Working Draft
LWER: Light Weight Encoding Rules
MBER: Minimum Bit Encoding Rules*

is universally implemented. For this reason, it is important that all future ASN.1 encoders continue to support BER.

3. OVERHEAD EXTENT

In developing OSI protocols for tactical use, there is a great concern regarding the large amount of overhead imposed by OSI. For OSI application protocols, this overhead consists of two different types: protocol overhead and encoding overhead. Reduction of both types of overhead is necessary to reduce application bandwidth requirements. However, the techniques for reducing the two types of overhead are quite different.

To examine the impact of these types of overhead, it is necessary to focus on specific protocols. It is unclear whether a "typical" OSI application exists. Yet, it is certain that particular OSI applications will be used in both strategic and tactical environments. The X.400 Message Handling System is one such application. Therefore, it is appropriate to examine the impact of overhead on the transfer of X.400 APDUs*. Where message content is considered, this paper will consider the IPMS* content type.

To analyze the overhead X.400 PDUs, it is first necessary to define the characteristics of the average message that will be sent. The makeup of this average message is determined based on a variety of sources. The message body size and subject field size were based on a

---

**APDU: Application Protocol Data Unit*
IPMS: Interpersonal Messaging System

statistical analysis of 1,000 randomly selected SMTP* messages. Although these messages do not accurately reflect use of the X.400 fields, they are believed to accurately represent user content requirements for character-oriented messages in an strategic environment. Statistics for the sample messages are shown in Table 1. These figures were used in lieu of any tactical statistics because no population was felt to be characteristic of the projected tactical messaging environment. The *sensitivity*, *importance*, and *languages* fields of the IPM heading were assumed to be unused for this analysis.

Table 1
Sample Message Content Statistics

Statistical Quantity	Body Size (octets)	Subject Length (octets)
Aggregate Size	2,058,015	38,121
Mean Size	2058	38
Standard Deviation	4991	42
Minimum	0	0
Maximum	41,973	121
Median	803	22

Note that for both the Body Size and Subject Length columns, the minimum value recorded was zero. Because it is assumed that a tacical message will always contain some user

---

SMTP: Simple Mail Transfer Protocol

data, further calculations will assume an arbitrary minimum user data size of 20 octets.

Two approaches were used to evaluate the impact of X.400 service selection on overhead:

(a) A min-max approach was used to determine the service selections that would yield the worst and best overhead results.

(b) Typical values were determined for each field based on assumptions about the tactical environment.

No security services were considered for the purposes of this investigation.

3.1 Protocol Overhead

Protocol overhead is composed of any additional information that is transferred for the purpose of supporting the services provided by an application. This generally includes any data elements that are exchanged between application entities that are not input by the user. It excludes any user data that are conveyed by the protocol. Protocol overhead also excludes any overhead incurred during encoding.

It can be difficult to determine what data elements should be included in a measurement of protocol overhead. For example, addressing fields are usually supplied by the user, but they are clearly integral components of the protocol and are rarely tightly coupled to the data being transferred.

In X.400 terms, protocol overhead includes all protocol elements in the P1, P3, and P7 envelopes, with the exception of the message content. Within the content itself, determining which elements contribute to overhead is much more difficult. Although all IPMS heading fields provide information to the recipient UA, many of these fields support functions that are unique to IPMS. Generally, these fields should be included in the protocol overhead. Only those fields that convey user information that qualifies the meaning of the body should be excluded. By these criteria, the following fields should be considered as user data:

(a) subject (conveys a brief summary of the body)

(b) importance (conveys the precedence or priority of the body)

(c) sensitivity (conveys a rudimentary security label for the body)

(d) languages (identifies the languages used in composition of the heading and body).

A study of the protocol overhead for X.400 PDUs revealed that, for the assumed average message previously described, the typical protocol overhead for a P3 PDU was 351.1 percent. This overhead excludes any additional overhead added by ASN.1 encoding. A summary of all protocol overhead is shown in Tables 2 and 3.

Table 2
Summary of Computed Protocol Overhead

Sample Set	User Data (octets)	P22 Heading Overhead (octets)	P3 Envelope Overhead (octets)
Minimum	20	22	45
Maximum	42,094	12,649	20,131
Typical	2096	2837	4521

Table 3
Percentage of Protocol Overhead

Overhead Category	Minimum User Data (%)	Typical User Data (%)	Maximum User Data (%)
Minimum	335.0	3.2	0.2
Typical	367.9	351.1	17.5
Maximum	163,900.0	1563.9	77.9

Clearly, this degree of overhead is excessive. However, it is expected that this overhead can be greatly reduced by applying restrictive profiling to limit what protocol fields may be used.

3.2 Encoding Overhead

A significant portion of any ASN.1 encoded PDU is occupied by overhead bits, such as tag values, length octets, and pad bits. By determining the extent of the

overhead and better understanding its causes, it may be possible to minimize its effects by developing better encoding rules.

Encoding overhead for BER can take three forms. Specifically, these forms are as follows:

(a) Identifier Octets
(b) Length Octets
(c) Inefficient Storage of Data Values.

Examples of these forms are shown in Figure 4.

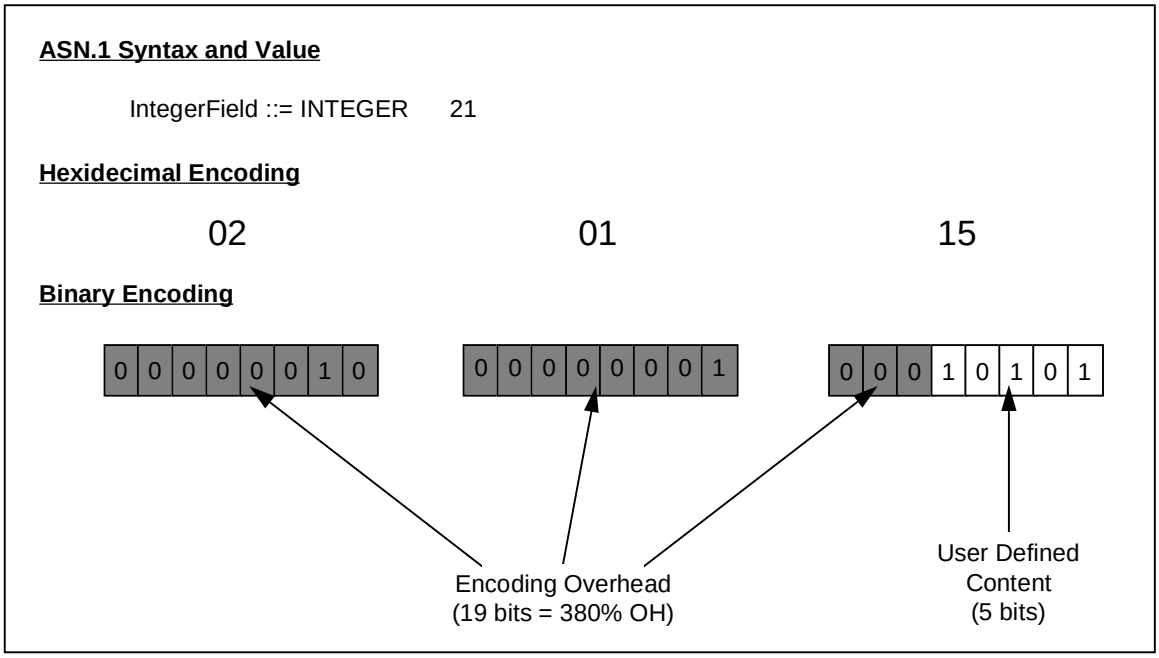

Figure 4 – Sources of Encoding Overhead

The number of identifier octets for each field is easy to determine. If the tag value is between 0 and 30, as are

most tags, then only one identifier octet is used. If the tag value is between 30 and 16,383, then three tag octets are used. Tag values above 16,383 are seldom encountered but would require commensurately more octets.

The number of length octets used for an encoding obeys a different relationship. Fields whose content is between 0 and 127 octets in length, carry only one length octet. Content lengths between 128 and 255 require two content octets. Three content octets are able to encode content between 256 and 65,535 octets in length. Each additional length octet increases the potential size of the content by a factor of 256.

Inefficient storage of data values is a less obvious source of encoding overhead. All values encoded with BER occupy a whole number of octets. Therefore, a BOOLEAN field always incurs 7 bits of encoding overhead. A field of type BIT STRING incurs varying amounts of overhead depending on the length of the bit string in question. A 7-bit string incurs less overhead than a 4-bit string. Note that BIT STRING encodings always incur an additional octet of overhead because an integer value containing the number of unused bits is always appended to the beginning of the content. An INTEGER field incurs a varying amount of overhead depending on the range of values the field is intended to hold. For example, if the abstract syntax specifies an INTEGER field to contain a value between 99 and 101, then the value can be unambiguously encoded using 2 bits. Using BER, an entire octet is necessary. A similar problem occurs with REAL values, except the amount of overhead also varies with the degree of accuracy required.

In the X.400 PDUs, encoding overhead was determined analytically based on an assumed average message. The results of the analysis showed a significant amount of overhead attributable to BER encoding. For the typical X.400 message characterized above, the approximate encoding overhead for a P3 APDU was 19.7 percent. As message sizes decrease, however, the encoding overhead becomes far more significant because the number of overhead octets is relatively independent from the size of the body. A summary of the overhead figures is shown in Table 4.

Table 4
Summary of Computed Encoding Overhead

Sample Set	Content Size* (octets)	P3 Encoding Overhead (octets)
Minimum	452	67
Maximum	55,155	2522
Typical	5345	1054

* = Includes protocol and encoding overhead from P22 heading

Examination of the intersections among these relatively independent figures is enlightening. As Table 5 and Figure 5 show, the extent of encoding overhead becomes formidable for small messages. For large message sizes, the extent of the overhead is insignificant.

Table 5
Percentage of Encoding Overhead

Overhead Category	Minimum User Data (%)	Typical User Data (%)	Maximum User Data (%)
Minimum	14.8	1.3	0.1
Typical	233.2	19.7	1.9
Maximum	558.0	47.2	4.6

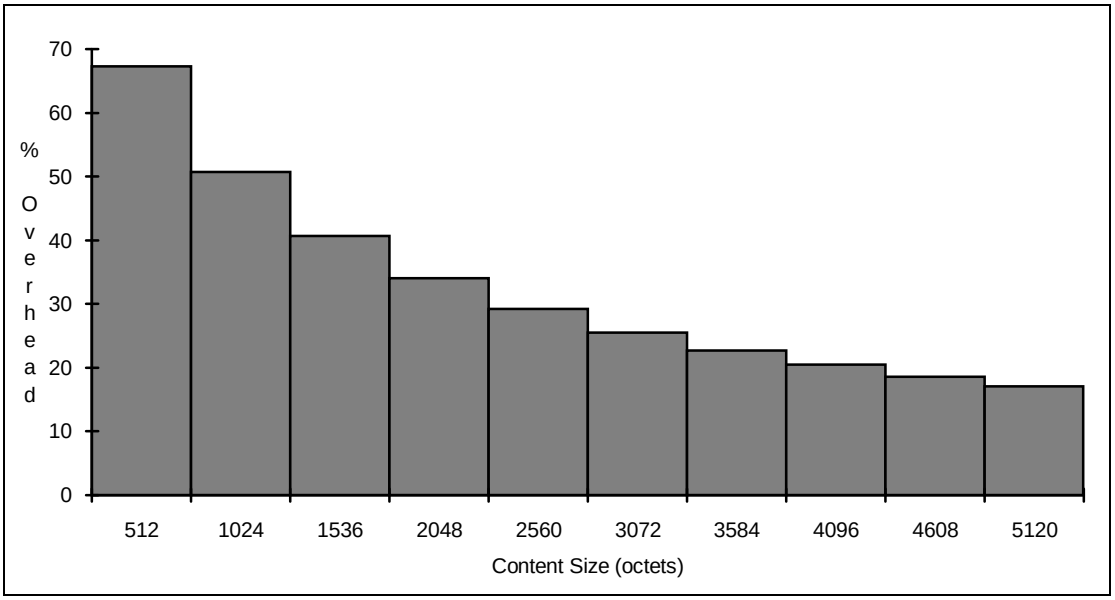

Figure 5 – Encoding Overhead for Small Messages

4. ENCODING OVERHEAD REDUCTION TECHNIQUES

The extent of the overhead imposed by the X.400 protocols and ASN.1 encoding rules illustrates the need for

mechanisms to reduce overhead. Several techniques exist to reduce the demands of OSI applications on the communications stack. Intelligent profiling can be used to limit the extent and variability of protocol overhead. In addition, data compression techniques could be employed at the presentation layer to reduce the overall size of the PDUs. To reduce encoding overhead, a more efficient set of encoding rules than BER must be adopted. This section focuses on techniques that reduce the size of ASN.1 encodings.

4.1 Tag Omission

Tag values may seem to be a vital part of the encoding, but in fact they are frequently not necessary. Because both sending and receiving parties are aware of the original abstract syntax, the type and identity of a particular element can often be inferred. This makes the identifier octets unnecessary in many circumstances. To achieve this type of operation, it is necessary to have clear rules for when tags should be omitted.

The rules for identifier octets must require tags in several circumstances. If an encoding is a SET construct, then the order of the elements cannot be inferred from the abstract syntax. Similarly, if the syntax includes a CHOICE field, the type of the element cannot be inferred. The presence of any OPTIONAL or DEFAULT fields in the syntax can cause ambiguity in the ordering of the elements; therefore identifier octets should always be included for these fields.

4.2 Length Code Omission

Length octets frequently can be omitted from encodings when the length of the element can be inferred from the abstract syntax. Any BOOLEAN, ENUMERATED, or bounded INTEGER field has an implicit length based on the abstract syntax. For example, the following syntax defines a field whose encoding has an implicit length in octets.

```
SampleElement ::= SEQUENCE {
     field1    INTEGER (0..15),
     field2    INTEGER
          (low (0), medium (1), high (2)) }
```

The first field of the SEQUENCE is limited to a 4-bit value. The second field can always be represented by 2 bits. The length of both fields, therefore, can be inferred as 1 octet.

4.3 Elimination Of Nesting

Many OSI applications define their abstract syntax using many layers of nested element definitions. In some cases, the nesting is necessary to adequately describe the data to be transferred; in other cases, the nesting adds clarity for implementors. Excessive nesting, however, has a negative effect on the encoding overhead.

Often, it is possible to omit levels of nesting during encoding. Most uses of the SEQUENCE construct can be ignored during encoding without introducing ambiguity.

4.4 Numeric Range Adjustment

Values that are limited to a minimum value greater than zero can often be reduced to a zero-aligned value during encoding. This technique can save considerable encoding overhead under the right circumstances. For example, the following syntax defines a field that is not zero-aligned.

```
SerialNumber ::= INTEGER (100502000..100503000)
```

In such a case, BER would require 4 octets to encode the serial number (plus another 2 for identifier and length octets). By adjusting the apparent range to (0..1000) for purposes of encoding, the value can instead be encoded in 2 octets. This technique is an application of a limited form of finite set data compression.

4.5 Bit Alignment

The BER incur a significant amount of encoding overhead by maintaining octet alignment for every encoding. To minimize this overhead, it is possible to ignore octet boundaries and encode data values using only the number of bits they require. For example, examine the following syntax.

```
SampleElement ::= SEQUENCE {
     field1    INTEGER (0..15),
     field2    INTEGER
          (low (0), medium (1), high (2)),
     field3    INTEGER (1..32) }
```

Using octet alignment, the encoding of this syntax requires twice as much space as with bit alignment.

Certain fields, such as OCTET STRING and BIT STRING, tend to gain little and lose much with bit alignment. Because they usually have large data values, strings do not benefit as much from an overhead savings of less than 7 bits. Also, OCTET STRINGS and other character-oriented strings need to be octet aligned for local processing. Pad bits, therefore, should be used to octet align such strings.

For bit alignment to be fully effective, it is necessary to specify how certain fields should be aligned. Unfortunately, there is not a suitable construct for this in the current ASN.1 abstract syntax rules. Such a construct should be added to the base standard. In the future, abstract syntax definitions should be written under the assumption that their encodings may be bit aligned.

4.6 Element Reordering

After reviewing the above techniques, it should be clear that many of these methods yield overhead savings only under the right circumstances. One way to improve the performance of these techniques is to reorder the ASN.1 fields before encoding to improve the likelihood of overhead savings. To do so, it is vital that the reordering algorithm be deterministic so that the other end systems can also determine the correct element order for decoding.

In particular, this approach can benefit the bit alignment technique. By grouping all string types together

in the encoding, the overhead of needless pad bits is avoided.

5. REDUCTION BENEFIT

To assess the quantitative impact of these techniques on real PDUs, the X.400 APDUs examined in Section 3 were re-analyzed using the improved encoding techniques. The results of this new analysis are shown in Table 6.

Table 6
Overhead Using Modified Encoding Rules

Overhead Category	BER Overhead (octets)	Modified Overhead (octets)	Overhead Reduction (%)
Minimum	67	62	7.5
Maximum	2522	1311	48.0
Typical	1054	726	31.1

In many instances, the techniques could not be applied because of the limitations of the X.400 abstract syntax. For example, the X.400 abstract syntax makes frequent use of the SET construct, thus limiting the benefit of tag omission. If these techniques were applied to new OSI applications written with encoding efficiency in mind, much greater overhead savings could be achieved.

6. EXISTING ENCODING RULE PROPOSALS

Several proposals for alternative encoding rules — in addition to DER, which has already reached DIS status — are currently progressing through ISO. Key among these is the specification for PER, which is designed to reduce encoding sizes. Another proposed set of encoding rules is LWER, which is designed to optimize the speed with which encodings can be processed. A former proposal, which has been largely subsumed by PER, is the MBER. MBER offers some unique features and ideas that are still worth promoting.

6.1 Packed Encoding Rules

The premise of PER is to provide a standard set of encoding rules that will minimize the size of the resulting encodings. The PER are identified by a single new transfer syntax object identifier, which can be used for negotiation via the presentation protocol.

The PER specifications are essentially an octet-aligned TLV encoding scheme, although the type and length identifiers frequently are omitted. The coding scheme uses many of the techniques described in Section 4, including:

(a) Tag Omission
(b) Length Code Omission
(c) Numeric Range Adjustment.

The PER standard recommends against use of the SET construct and OPTIONAL fields in abstract syntax definitions. It also defines when the various encoding forms (i.e., short definite, long definite, and indefinite) shall be used. The

PER specifies that, when indefinite encodings are used, the end-of-contents octets shall be dropped whenever the length can be determined from the syntax.

Another concept introduced in PER is BIT STRING packing. This rule specifies that, whenever a construct is composed only of BIT STRING and LOGICAL fields whose tag, length, and unused bit values are not necessary, the data values shall be concatenated at the bit level for maximum packing efficiency. Note that this is only possible for fixed length fields.

Although the PER implementations will be capable of generating smaller encodings of ASN.1 abstract syntax, the improved overhead performance is not without cost. The rules that make up PER are complex compared to BER and, therefore, are likely to require greater processing resources in implementations.

6.2 Light Weight Encoding Rules

The LWER is designed to maximize the processing speed of the encoder at the expense of flexibility, extensibility, and compactness. It does so by defining the transfer syntax to match the local representation format as closely as possible. To this end, LWER defines six new transfer syntaxes that cover a broad spectrum of representation formats. These syntaxes define rules with varying word sizes and octet ordering schemes. Maximum processing speed usually will be achieved when both end systems are operating in their "native" context.

The LWER encodes data types similarly to the way they are stored on many computer systems. Most primitive types, such as INTEGER, REAL, ENUMERATED, and BOOLEAN, are encoded as fixed-length fields of one word. The number of octets in the word depends upon the transfer syntax agreed during connection establishment. String fields, such as BIT STRING and OCTET STRING, are encoded as a length code and a pointer. The pointer value indicates the position of the string data later in the encoding sequence. Tag values are never included in LWER encodings.

Use of LWER will generally increase the encoding overhead significantly. This approach has merit in high-speed applications where bandwidth is plentiful, or where processing resources are extremely limited. This latter case may apply to some tactical scenarios (e.g., man-pack SATCOM*).

6.3 Minimum Bit Encoding Rules

The MBER proposal was originated by the civil aeronautical community. Many of the aeronautical requirements behind the proposal are held in common with the tactical environment. Although its aim is similar to that of PER, its approach is more aggressive.

In addition to specifying many of the same techniques as PER, MBER employs bit alignment to further reduce the size of resulting encodings. To support this encoding form, the proposal suggests a new ALIGN constraint for the ASN.1 notation. This constraint would allow abstract syntax

---

SATCOM: Satellite Communications

developers to specify particular alignment requirements (e.g., octet alignment, dibit alignment) for any field.

The MBER proposal offers significantly lower encoding overhead than PER, yet it seems to have been subsumed by the PER development within ISO. Although the specifics of MBER are not sacred, the tactical community should strongly encourage and support the development of some type of bit aligned encoding rules.

7. CONCLUSION

This paper has demonstrated that the ASN.1 BER encoding overhead is significant, and it can be reduced through application of alternate encoding rules. The percentage of encoding overhead increases dramatically as message sizes decrease. This makes encoding overhead a serious concern for the tactical environment, where many messages are expected to be small and highly perishable.

To promote reduced encoding rules, several actions are called for in the standardization community. Specifications for PER, which are being progressed as CD 8825-2, should soon be available in DIS form. These encoding rules would offer significant bandwidth savings over BER. The PER effort should be monitored carefully and actively supported by national standardization bodies. A proposal incorporating bit alignment, such as MBER, should be introduced to ISO and considered as a long-term solution.

REFERENCES

1. **ISO/IEC: CD 7498-1, Open Systems Interconnection - Basic Reference Model, November 1991 (unclassified)**

2. **ISO/IEC: IS 8824, Information Technology - Open Systems Interconnection - Specification of Abstract Syntax Notation One (ASN.1), second edition, 15 December 1990 (unclassified)**

3. **CCITT: X.208, Specification of Abstract Syntax Notation One (ASN.1), 1988, (unclassified)**

4. **ISO/IEC: IS 8825, Information Technology - Open Systems Interconnection - Specification of Basic Encoding Rules for Abstract Syntax Notation One (ASN.1), second edition, 15 December 1990 (unclassified)**

5. **CCITT: X.209, Specification of Basic Encoding Rules For Abstract Syntax Notation One (ASN.1), 1988 (unclassified)**

6. **CCITT: X.400, Message Handling, System and Service Overview, 1988 (unclassified)**

7. **ISO/IEC: IS 10021, Information Technology - Text Communication - Message-Oriented Text Interchange System (MOTIS), first edition, December 1990 (unclassified)**

8. **CCITT: X.419, Message Handling Systems, Protocol Specifications, 1988 (unclassified)**

9. **ISO/IEC: IS 10021-6, Information Technology - Text Communication - Message-Oriented Text Interchange System (MOTIS) - Part 6: Protocol Specifications, first edition, December 1990 (unclassified)**

10. **CCITT: X.420, Message Handling Systems, Interpersonal Messaging System, 1988 (unclassified)**

11. **ISO/IEC: IS 10021-7, Information Technology - Text Communication - Message-Oriented Text Interchange System (MOTIS) - Part 7: Interpersonal Messaging System, first edition, December 1990 (unclassified)**

12. ISO/IEC: DIS 8824-1, Information Technology - Open Systems Interconnection - Abstract Syntax Notation One (ASN.1) - Part 1: Specification of Basic Notation, 8 October 1992 (unclassified)

13. ISO/IEC: DIS 8824-2, Information Technology - Open Systems Interconnection - Abstract Syntax Notation One (ASN.1) - Part 2: Information Object Notation, 8 October 1992 (unclassified)

14. ISO/IEC: DIS 8824-3, Information Technology - Open Systems Interconnection - Abstract Syntax Notation One (ASN.1) - Part 3: Constraint Specification, 8 October 1992 (unclassified)

15. ISO/IEC: DIS 8824-4, Information Technology - Open Systems Interconnection - Abstract Syntax Notation One (ASN.1) - Part 4: Parameterization of ASN.1 Specifications, 8 October 1992 (unclassified)

16. ISO/IEC: DIS 8825-1, Information Technology - Open Systems Interconnection - Specification of ASN.1 Encoding Rules - Part 1: Basic Encoding Rules, 8 October 1992 (unclassified)

17. ISO/IEC: CD 8825-2, Information Technology - Open Systems Interconnection - Specification of ASN.1 Encoding Rules - Part 2: Packed Encoding Rules, July 1991 (unclassified)

18. ISO/IEC: DIS 8825-3, Information Technology - Open Systems Interconnection - Specification of ASN.1 Encoding Rules - Part 3: Distinguished Canonical Encoding Rules, 8 October 1992 (unclassified)

19. ISO/IEC: WD 8825-4, Information Technology - Open Systems Interconnection - Specification of ASN.1 Encoding Rules - Part 4: Light Weight Encoding Rules (unclassified)

20. D. Blum, R. Rice: Minimum Encoding Rules (MBER) for the Abstract Syntax Notation (ASN.1), 4 April 1990 (unclassified)