

White Paper

ASN.1 is Reaching Out!

By John Larmouth

Birth and marriages

ASN.1, SDL, and TTCN were all born in the 1980s, in the hey-day of OSI. (SDL would claim a birth-date of 1980, ASN.1 of 1984, and TTCN a year or so later.) ASN.1 was conceived by Xerox putting seminal ideas into their Courier protocol, but the birth pangs accompanied X.400 into the world, a twin brother, and the dominant partner in childhood. X.400 was soon to be joined from the same parentage by X.500, which quickly established itself on the worldwide stage, and rampaged through the worlds of security and e-commerce. It should have been certified at birth!

ASN.1 was a brilliant child, widely adopted and embraced by many long-dead OSI Standards, and by some that still retain major importance today.

In its youth, it was wooed and wedded by SDL and TTCN, forming a comfortable liaison with those Recommendations that has stood the test of time. Together they have formed important partnerships that have not only brought civilization to the world of protocol specification, but have (perhaps more importantly!) brought profits to many tool-vendors and "I sleep-well-a -nights" to many implementors.

To quote from the SDL Forum Web-site: "The use of the object model notation of SDL-2000 in combination with MSC, traditional SDL state models and ASN.1 is a powerful combination that covers most aspects of systems engineering."

The combination of SDL, TTCN, and ASN.1 is powerful indeed. SDL will be well known to everyone in this audience as providing what is probably still today the most powerful and complete means of specifying (and with SDL tools, implementing) the required behaviour of communicating systems. TTCN provides the means for the specification and implementation of test suites.

As maturity develops, both TTCN and SDL are developing new acquaintances and alliances, but ASN.1 remains a major support and platform for their activities.

This presentation will identify some of the exciting new developments in the second youth of ASN.1. The ASN.1 platform for SDL and TTCN is quite dramatically extending its reach into legacy protocols and is developing new relationships with the fledgling XML world, and perhaps soon with UML.

This in turn gives new opportunities for SDL and TTCN, and for their supporting tools. Together they can enable what were hitherto non-ASN.1 protocols (such as Bluetooth - surely to be the buzz-word of the next decade) to be easily and precisely specified, and their implementations rapidly produced and tested.

There has been a long feud between the Montagues and Capulets (character-based protocols with ad hoc BNF versus binary-based protocols with formal notations and tools), but the "middle-way" of XML-based encodings now seems an attractive proposition to many.

Will the children of the 1980s live a long old age in 2080? Time will tell, but in human terms, they are now at their prime - mature, with plenty of real-world experience, but still young enough to adapt and develop. Viva SDL, ASN.1, and TTCN!

Protocol specification through the ages

Computer communications technology has been developing for approximately the last 1.5 billion seconds, with major advances every 150 million seconds.

We can identify a very clear stone-age era with crude syntaxes and very rudimentary tools, a bronze-age era when tools were sharpened and refined, through to the present day of advanced technology and clear insights.

There are, of course, three main thrusts to protocol development: syntax specification, procedure specification, and testing methodologies and suites. As this is a presentation on ASN.1, please excuse concentration on the former.

The earliest work (Montague's stone-age) was quite crude and simple: fields of encodings were fixed length, always present, and never repeating. Pretty pictures drawn on the walls of caves sufficed to specify all that was needed.

Capulet's stone-age was not much better: simple command lines with some sort of terminator, ASCII (a dinosaur of there ever was one) encoded, but Capulet matured to the bronze-age very rapidly through the insights of Bachus and Naur, but has not progressed much since then.

Montague progressed from simple pictures to "tabular notation", based on a "type, length, value" approach, showing the first real signs of an awareness of the problems of "extensibility" - the need for easy interworking between deployed version 1 systems and enhanced version 2 systems developed many years later.

But it was still another 150 million seconds or so before realization dawned that a clear separation of abstract syntax specification from encoding specification provided real advantages of re-usability, and good profits for tool vendors.

And it took another three hundred million seconds before there was realization that extensibility did **not** require a TLV style of encoding. The world could have very efficient, compact encodings, and **still** have interworking between deployed systems and new systems, with minimal effort, and in a bug-free manner.

We will ignore the rather short Iron Age and the Agricultural and Industrial Revolutions and proceed to the modern era. There is now much better understanding of the problems to be solved in protocol design, and the options that are possible, with significant recent advances in understanding ways of achieving extensibility without prejudicing compact encodings.

Tools are widely available with increasing flexibility in what they can support.

So what does ASN.1 offer?

There are many well-known features of ASN.1, some of which have been present from its earliest beginnings:

- A simple notation for abstract syntax definition, independent of the complexities of encoding (transfer syntax) design.
- Full support in all encoding rules for interworking between version 1 and version 2 systems, and for vendor-specific extensions of a base protocol through the open type, information object class, and relational constraint mechanisms.
- Easy mappings with good tool support from abstract syntax definition to C, C++, Java, on a very wide range of platforms, enabling rapid development of implementations incorporating much reusable and bug-free code.
- Highly robust and mature TLV-style encodings giving plenty of support for extensibility.
- Highly compact encodings for low-band-width applications, but still with extensibility support, and available at the touch of a button from an increasing number of ASN.1 tools.
- Mature notation and tools deployed in a very wide range of industries from telecommunications through to intelligent transportation systems, parcel delivery, and news networks.

- Fully canonical encoding rules for specialised applications, such as those supporting digital signatures and certificates, used by many security-related protocols (for example SET).
- Strong links to system design and testing methodologies such as SDL and TTCN.

And what are the main ASN.1 developments today?

There are two pieces of important new work which are near completion as extensions to ASN.1, and which are expected to increase considerably the range of applications in which it is employed. This will not only increase the market sector for ASN.1 tools, but also for associated tools such as SDL and TTCN.

The two developments are very different, and provide very different opportunities and challenges, but interestingly the greatest benefits may come from a synergy between the two developments.

The first to be described is the Encoding Control Notation (ECN). The second is the XML Encoding Rules and other support for XML. There is also strong interest in linking ASN.1 with UML, but that work is still in its infancy, and will not be discussed further in this presentation.

With these together we can add the bullets:

- The ability to use ASN.1+ECN tools to support any protocol, whether initially defined using ASN.1 or not.
- The ability for ASN.1 tools to interact with XML tools, including standard Web browsers, to input and display ASN.1 values in a very human-readable form.

Encoding Control Notation (ECN)

The origins of ECN were in a dinner conversation during an ISO/ITU-T meeting in Lannion between Frank Schramm (Siemens) and Colin Wilcock (Nokia). The idea (like most good ideas) was very simple.

It had been known for some time that the ASN.1 notation could easily be used to describe the abstract syntax for pretty well any protocol. Put another way, with the mapping of ASN.1 type definitions to C, C++ and Java data-structures, ASN.1 could be used to define an API for any protocol, hiding all details of "auxiliary fields" (length and choice determinants, mechanisms for extensibility, etc) and details of actual encoding.

Unfortunately, the application of the standardized encoding rules to such a specification rarely produced the bits-on-the-line that these "legacy protocols" were using.

With the development of the ASN.1 Packed Encoding Rules (PER), it was often possible to get the correct encodings, but only by including these "auxiliary fields" within the ASN.1 definition. This lost most of the advantages of information hiding that ASN.1 offered, and put the onus of generating correct length and choice determinants (and other aspects of encoding) back onto the application. This was very unattractive!

The ECN work aimed to provide a notation for encoding control that could not only specify the way in which primitive types in ASN.1 would be encoded, but would also enable auxiliary fields to be added in a separate specification. These goals have been achieved.

Put simply, it is now possible to:

- Specify any existing "legacy" protocol using ASN.1, including in the ASN.1 only fields which carry application semantics, hiding all aspects of encoding detail.
- Specify separately (with considerable flexibility) the details of the encoding rules to be applied to produce any desired form of encoding.

Application to Bluetooth

ECN has been applied in the complete definition of the Bluetooth Service Discovery Protocol, and in the partial definition of a number of other protocols such as Hiperlan and Tetra. Here we concentrate on the Bluetooth Service Discovery protocol.

Service Discovery is at the heart of Bluetooth, enabling a mobile device moving into a new geographic environment to discover the existence of other Bluetooth devices in that environment. It is key to the fully automatic configuration that is needed for communication by domestic devices.

The protocol has many similarities with features of ASN.1 BER and PER, but of course uses totally different encodings at the detailed level. Many messages are encoded in a fixed-length PER-style, but contain an "attribute id" and "attribute value" field which corresponds very closely to an ASN.1 "open-type".

Bluetooth uses a tabular notation to define attribute ids and types, but also provides for vendor-specific additions of new types using constructors in much the same way as ASN.1 types are defined. For these types, Bluetooth uses a BER-like TLV structure, but of course with different encodings.

It has similar concepts of universal class tags for the "T" part (but without user over-ride of tag values), and of a number of built-in primitive types and constructors.

Encoding rules have been written using ECN that can be applied to **any** ASN.1 type definition that can be written using the Bluetooth primitive types and constructors. This provides facilities that enable tool-support for encoding Bluetooth applications, and means that the Bluetooth Service Discovery Protocol can now be regarded as an ASN.1-based protocol, with important implications for tools which support ASN.1.

For ECN to be useful, it is not necessary for standardised protocols to be re-defined using ASN.1+ECN. The necessary ASN.1+ECN specifications can be provided by ASN.1 tool vendors to implementors (for example, of Bluetooth, Hiperlan or Tetra), with the correctness of the ASN.1 and ECN warranted and supported by the tool-vendor in the same way as correct encodings are normally warranted and supported by the tool. This enables implementation teams to use a single familiar tool for all protocols.

Examples of ECN

Examples of two pieces of ECN used in the specification of the Bluetooth data element sequence are:

```
bluetooth-tag-encoding #TAG ::=
    {ENCODING SPACE SIZE 8
     EXHIBITS HANDLE "Bluetooth tag" AT {0..7} }

length -delimited-repetition
    {< REFERENCE: length >} #REPETITION ::=
        {ENCODING
         {REPETITION-SPACE
          SIZE variable-with-determinant
          MULTIPLE OF octet
          USING length } }
```

Status of ECN

The base standard is ITU-T Rec. X.692 | ISO/IEC 8825-3. It has completed its ISO FCD ballot, and final text is now ready for ITU-T Last Call. It is expected to undergo only editorial changes, and to be finally approved before the end of this year.

There is also an amendment giving extensive support for a range of extensibility mechanisms. The FPDAM ballot for this is underway in ISO, and final approval is also expected before the end of the year.

Syntax checkers are available, and a more complete tool is expected to become available during the summer.

There is a web site at <http://asn1.elibel.tm.fr/ecn> which contains a complete Word ECN tutorial with examples, and presentation slides in PowerPoint.

XML Support

There are several pieces of XML work planned, but the most mature, and arguably the most important, is the provision of ASN.1 XML Value Notation, and XML Encoding Rules (XER).

The addition of XML Value Notation means that within any ASN.1 module, in test specification, or as input and output to ASN.1 tools, a new value notation can be used which is XML mark-up.

The tags used are based on the identifiers in ASN.1 SEQUENCES and SETS, and on type reference names where necessary. An example is:

An example of ASN.1 XML Value Notation

With the ASN.1 type:

```
Invoice ::= SEQUENCE
{
  number      INTEGER,
  name        UTF8String,
  details     SEQUENCE OF
              SEQUENCE {
                part-no  INTEGER,
                quantity INTEGER},
  charge      REAL,
  authenticator BIT-STRING}

```

we can have the value:

```
this-invoice ::=
<Invoice>
  <number>32950</number>
  <name>Funny name with &lt; and &gt; in it</name>
  <details>
    <part-no>296</part-no>
    <quantity>2</quantity>
    <part-no>4793</part-no>
    <quantity>29</quantity>
  </details>
  <charge>397.65</charge>
  <authenticator form=hex>
    EFF8 E976 5403 629F

```

</authenticator>
</Invoice>

For the XML Encoding Rules, it is a UTF8 encoding of a value of the outer-level type that is transmitted down the line, with a short XML preamble and postlude.

Advantages of the XML Value Notation

This new ASN.1 facility provides:

- The ability to send values of ASN.1 PDUs to browsers for easy monitoring or display of received messages.
- The ability to generate ASN.1 PDUs (probably for conversion by an ASN.1 tool to PER encodings before transmission) from an XML tool.
- A more readily understandable (although more verbose) value notation for input and output of values to and from ASN.1 tools, for monitoring and testing, and for the specification of test suites.
- A human-readable text-based (but verbose) encoding format suited to browsers, which can be easily converted to or from a PER encoding by ASN.1 tools.

This is quite powerful support in itself, but combined with ECN provision, it enables, for example, the input and display of Bluetooth Service Discovery Protocol messages (for testing and monitoring purposes) using standard XML tools. The whole can be much greater than the two parts.

There is further XML work planned in the area of mapping XML schema languages and ASN.1 schemas (type definitions) but neither the precise focus of this work, nor an exact understanding of the user demand for it, is decided yet. This work will probably begin to mature during 2002, and is not discussed further.

Status of the XML work

The work involves two new amendments (one to ITU-T Rec. X.680 | ISO 8824-1 and one to ITU-T Rec. X.681) to support the new value notation, and one new (quite small) standard (ITU-T Rec. X.693 | ISO 8825-4) to define use of the XML Value Notation to provide the XML Encoding Rules (XER).

These are all complete and under FPDAM and FCD ballot, and are expected to be finally approved in late 2001. ASN.1 tools supporting the new value notation and XER are expected to become available during this summer, together with tutorial material.

Conclusion

ASN.1 has come a long way from the ten page X.409 produced in 1984, and is showing an ability to adapt to changing user requirements which should hopefully give it a long life yet, and reinforce its suitability as a base platform for communications protocol specification.

The recent merger of the main standards bodies involved in ASN.1 work (ITU-T SG7) and in SDL and TTCN work (ITU-T SG10) to form the new (ITU-T SG17) should enable integrated support for these new features across the complete spectrum of protocol specification, system definition, and testing, both at the standards level and within tools, to the benefit of both standardisers and implementors.

We have finally reached the Information Age!

About John Larmouth

John Larmouth was the Founding Director of the Information Technology Institute at the University of Salford, where he has worked for over twenty years. A graduate of Cambridge University, he has been involved with ASN.1 since its introduction as an ISO standard in the early 1980s. He served as Editor of the Standard for the first ten years of ASN.1's existence and has been ISO Rapporteur for ASN.1 for the past decade.