ASN.1 Is Reaching Out!

John Larmouth

Salford University Professor
1 Blueberry Road, Bowdon, Altrincham, Cheshire WA14 3LS, England.
j.larmouth@salford.ac.uk

Abstract. This paper takes a light-hearted look at the life of ASN.1 and it relationships with other languages from the time of its birth to the year one of the third millennium – the information age.

1 Birth and Marriages

ASN.1 [1,2,3,4,5], SDL [6], and TTCN [7,8] were all born in the 1980s, in the hey-day of Open Systems Interconnection (OSI). SDL would claim a birth-date of 1976, 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 [9] into the world, a twin brother, and the dominant partner in child-hood. X.400 was soon to be joined from the same parentage by X.500 [10], 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 liaisons with those Recommendations that have 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

R. Reed and J. Reed (Eds.): SDL 2001, LNCS 2078, pp. 241–249, 2001.

dramatically extending its reach into legacy protocols and is developing new relationships with the fledgling XML [11] 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 to 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 an develop. Viva SDL, ASN.1, and TTCN!

2 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 if 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" (TLV) 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 for 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.