The Use of Formal Methods in Communications Standards

K. J. Turner

1 Introduction

Formal (i.e. mathematical) methods in computer science have been under development for the past 50 years, but the last 15 years has seen a great surge of interest in formal methods for specification. Communication systems have been a popular area for application of formal methods as they exhibit challenging technical features such as concurrency and distribution. The trend towards open systems, such as found in the ISO work on OSI (*Open Systems Interconnection*), has provided a strong impetus for methods that allow interfaces and services to be specified precisely.

In recognition of this, CCITT began work in 1976 on standardising SDL (*Specification and Description Language*, Z.100) as a formal notation for telecommunications systems. Similarly ISO began work in 1980 on standardising Estelle (*Extended Finite State Machine Language*, ISO 9074) and LOTOS (*Language Of Temporal Ordering Specification*, ISO 8807) for formal specification of open systems standards. All three languages reached a stable state in 1988, and have been widely applied to specifying communications standards.

2 Formality

A **formal system** is a mathematical theory. A formal system often defines a **formal language** that may be used to write specifications supported by that theory. A formal language consists of a set of **symbols** and a set of **rules** for combining symbols into expressions and manipulating them. A formal language has a formally defined **syntax** that gives rules for correctly formed statements, and may have a formally defined **semantics** that assigns meaning to syntactically correct statements. Formal languages are broadly categorised as **non-constructive** (meaning that they specify a system only in terms of its observable 'black-box' properties) and **constructive** (meaning that they allow a mathematical model of a system to be built).

A **formal method** is a means of developing a (computer) system by use of formal systems and formal languages. A formal method addresses all aspects of development, from initial identification of requirements to final validation. It is noticeable that design methods and design languages in computer science have become progressively more **formal** and more **abstract**. Machine code was used in the early days of computing, but this gave way to assembly languages and low-level languages. Nowadays, high-level programming languages and fourth/fifth-generation languages are commonplace. Specification languages based directly on mathematical concepts are also gaining ground. Hardware design languages have gone through a similar evolution towards formality and abstractness.

The major motivation for formal methods is precision of expression, and hence the avoidance of ambiguity. Note that this still leaves scope for implementation freedom. Since formal languages are generally abstract, they lead to concise specifications. It is important that the designers in a team have a precise notation for communication of ideas; formal approaches can be very helpful for this. Since formal methods are based on mathematics, there is the possibility of rigorous analysis and proof of formal specifications. Unfortunately, the tools and techniques for this are still not practicable for large systems, but it is still worthwhile to invest in formal methods because of this long-term prospect.

Conformance to standards is becoming a major issue in computing, so the availability of formal specifications of standards is an important step forwards. Increasingly, suppliers of computing and communications systems are being held legally responsible for the performance and correct operation of their equipment. As part of limiting liability, suppliers may have to demonstrate that they use appropriate techniques such as formal methods.

[1] Department of Computing Science, University of Stirling, STIRLING FK9 4LA (Phone: 0786-67422, Email: kjt@uk.ac.stir.cs)

Procurement standards are in future likely to insist on formal methods in areas such as safety-critical systems. Quality-critical systems and secure systems are other obvious areas of application.

Most development processes conform to a **waterfall model**. It is common experience to find that an error at an early stage is propagated to later stages. The cost of fixing an error or omission when a specification is written may be one thousandth the cost of correction when the system has been implemented and delivered. Formal methods are therefore very effective in early design and specification, where the relatively high costs of formal methods are offset by the substantial savings in later re-working. If a formal specification is constructive, it may be used as a prototype to investigate functionality. Later in the development process, a formal specification may be used to derive integration tests and conformance tests. Formal specifications are also a good basis for documentation.

Training, tools and case studies are necessary ingredients in ensuring industrial take-up of formal methods. There are text-books formal approaches such as CSP (*Communicating Sequential Processes*), VDM (*Vienna Development Method*) and Z. Academic and commercial organisations now run training courses on formal approaches such as LOTOS, VDM and Z. Tools for formal methods are appearing, often under the banner of CASE (*Computer-Aided Software Engineering*). For example, tools for the Estelle, LOTOS and SDL languages are already commercially available or nearing completion. Case studies in the use of formal methods include the VIPER chip (RSRE, University of Cambridge), CICS (IBM, University of Oxford), mobile satellite communications (ESA, University of Madrid), and Transputer chips (Inmos, University of Oxford).

3 Formal Description Techniques

The difficulties of specifying OSI standards in natural language were recognised early, motivating CCITT and ISO to work on international standards for FDTs (*Formal Description Techniques*). This led to the development of Estelle, LOTOS and SDL for producing formal specifications that are **unambiguous**, **clear**, **concise**, **complete** and **consistent**. FDTs are also intended to help in **analysis**, **implementation** and **testing**. Other standardised but non-formal techniques include ASN.1 (*Abstract Syntax Notation 1*, ISO 8824/8825, X.409) for expressing protocol formats, and TTCN (*Tree and Tabular Combined Notation*, ISO 9646-3) for expressing conformance tests. There are many other formal but non-standardised techniques that have been used to describe communications systems, including: finite state machines, nets, process algebras, logic, declarative languages and abstract data types.

There will never be an ideal formal method any more than there will ever be an ideal programming method. Factors in choosing an appropriate formal method include: the type of application, the existing user community, the maturity of the method, the degree of formality and abstractness required, the requirements for conciseness and readability of specifications, the time to become familiar with the method, and the availability of training and tools.

In communications systems, LOTOS (and to a lesser extent Estelle) has been used widely to specify most of the layers of OSI: Presentation, Session, Transport, Network and Data Link. LOTOS has also been used for space data systems (CCSDS 701). LOTOS and Estelle have been used for distributed applications such as TP (*Transaction Processing*). FDTs such as LOTOS and Z are being investigated for the standardisation work on ODP/DAF (*Open Distributed Processing/Distributed Applications Framework*). Other formal approaches are being considered for distributed systems such as ODA (*Office Document Architecture*).

4 Conclusion

The last decade has seen major advances in the development of practical formal methods and tools. Safety critical systems and communications systems have been the main areas to benefit, but it is certain that formal methods will be applied more widely in future. The availability of standardised formal techniques and commercially available training and tools will encourage the uptake of formal methods.

Selected Bibliography

Aggarwal, S. and Sabnani, K. (it eds.): Proc. of the *Eighth IFIP WG 6.1 Symposium on Protocol Specification, Testing, and Verification*, North-Holland, Amsterdam, 1988

Bolognesi, T. and Brinksma, H.: *Introduction to the ISO Specification Language LOTOS*, Computer Networks and ISDN Systems, *14* (1), pp. 25–29, North-Holland, Amsterdam, 1988

Budkowski, S. and Dembinski, P.: *An Introduction to Estelle: A Specification Language for Distributed Systems*, Computer Networks and ISDN Systems, *14* (1), pp. 3–24, North-Holland, Amsterdam, 1988

CCITT: *Specification and Description Language*, Z.100, International Consultative Committee for Telephony and Telegraphy, Geneva, January 1988

Cohen, B., Harwood, W. T. and Jackson, M. I.: *The Specification of Complex Systems*, Addison-Wesley, Reading, 1986

Diaz, M. *et al.*: *The Formal Description Technique Estelle*, North-Holland, Amsterdam, 1989

van Eijk, P. H. J., Vissers, C. A., Diaz, M.: *The Formal Description Technique LOTOS*, North-Holland, Amsterdam, 1989

ISO: *Information Processing Systems - Open Systems Interconnection - Estelle, A Formal Description Technique based on an Extended State Transition Model*, ISO 9074, International Organisation for Standardisation, Geneva, November 1988

ISO: *Information Processing Systems - Open Systems Interconnection - LOTOS, A Formal Description Technique based on the Temporal Ordering of Observational Behaviour*, ISO 8807, International Organisation for Standardisation, Geneva, August 1988

ISO: *Guidelines for the Application of Estelle, LOTOS, and SDL*, TR 10167, International Organisation for Standardisation, Geneva, October 1990

Logrippo, L. M. (it ed.): Proc. of the *Tenth IFIP WG 6.1 Symposium on Protocol Specification, Testing, and Verification*, North-Holland, Amsterdam, 1990

Quemada, J. (it ed.): Proc. of the *FORTE 90 Conference*, North-Holland, Amsterdam, 1991 (*to appear*)

Rockström, A.: *An Introduction to the CCITT SDL*, Televerket, Stockholm, 1985

Saracco, R. and Tilanus, P. A. J.: *CCITT SDL: Overview of the Language and its Applications*, Computer Networks and ISDN Systems, *13* (2), pp. 65–74, North-Holland, Amsterdam, 1987

Turner, K. J.: *LOTOS - A Practical Formal Description Technique for OSI*, Proc. International Conf. on *Open Systems*, pp. 265-280, London, March 1987

Turner, K. J. (it ed.): Proc. of the *FORTE 88 Conference*, North-Holland, Amsterdam, 1988

Turner, K. J. (it ed.): *Using Formal Description Techniques*, John Wiley, 1991 (*to appear*)

Turner, K. J. (it ed.): *An Introduction to Formal Methods using LOTOS*, Macmillan, 1991 (*to appear*)

Vissers, C. A., Brinksma, E., and Scollo, G. (it eds.): Proc. of the *Ninth IFIP WG 6.1 Symposium on Protocol Specification, Testing, and Verification*, North-Holland, Amsterdam, 1989

Vissers, C. A. and Scollo, G.: *Formal Specification in OSI*, Lecture Notes in Computer Science, *248*, pp. 338–359, Springer-Verlag, Berlin, 1987.