

Standard-Compliant, but Incompatible?!

Tineke M. Egyedi

Delft University of Technology, ICT Dept., Faculty of TPM, P.O.Box PO Box 5015,
2600 GA Delft, the Netherlands, +3115278-6344, T.M.Egyedi@tbm.tudelft.nl

Abstract. This paper addresses the question why standard-compliant IT products often do not interoperate. The findings are based on an institutional analysis, three case studies, and a debate among experts. The paper concludes that some dilemmas cannot be resolved easily. However, many causes *can* be addressed, in particular those in the area of standard development. Where interoperability is concerned, standard development and implementation issues cannot be meaningfully separated.

Keywords: standards, implementation, interoperability, incompatibility

1. Introduction

Standard-compliant products do not always interoperate, as most IT users will have experienced. It is a frustrating problem, and sometimes drastic measures are taken to circumvent it. For example, in a well-documented case study a university decided to adopt a standards-based single-vendor solution for its wide area network (IEEE 802.11b; Jakobs, 2005). This single-vendor ‘solution’, be it based on open or de facto standards, is a rather widespread defensive procurement strategy. It usually resolves incompatibility in the short run, but it undermines the basic notion that open standards allow us to combine the best of different vendors and protect us from vendor lock-in – a different, but equally frustrating problem. (As a warning, in the said case study the single vendor added a proprietary extension to the standard which later, as other vendors became involved, caused its own set of interoperability problems; Jakobs, 2005)

In this paper the problem of standard-compliant, but incompatible IT products and services is examined more closely. It addresses causes of incompatibility between standard-compliant software, and makes recommendations about how to deal with them.

The paper applies one important restriction. It does not address incompatibility which has a ‘malevolent’ background. That is, there are companies which introduce changes to standards to frustrate the development and adoption of competitive products, or to lock users into a proprietary technology. These companies, for example, elaborate standards by adding extra functionalities (embrace-and-extend strategy). Or

they implement only part of the standard (embrace-and-omit strategy); or they introduce local adaptations to the standard (embrace-and-adapt). In all three situations the integrity of a standard is at stake¹. Sometimes interoperability can be re-created, but this requires much extra effort. More commonly market fragmentation results.

What Sherif *et al.* (2005) argue with regard to standards quality, namely that there is no corrective market incentive to address lack of standards quality, also applies to the corrupt use of standards. “The diverse interests that affect standardization, the distributed nature of its management process and the time lag between a standard and its implementation in products and services mean that there is no clear accountability in terms of profit and loss responsibilities due to deficiencies in an ICT standard. In some cases, those who pay the cost of the lack of quality are not those who made the decisions. Thus, market mechanisms will rarely provide the driving incentive to carry out the intensive planning and coordination across organizational boundaries that are needed to produce a quality standard.” (Sherif *et al.*, 2005, p.230) Little can be done if harm is meant. This paper therefore does not deal with the intentional corruption of standards, or ‘malevolent standard deviations’. Instead, it focuses on incompatible implementations that come about unintentionally or for valid economic, functional or other reasons (i.e. *benevolent standard deviations*). For example, certain standards’ features may be unnecessary for the intended use and may therefore not be implemented. Although the consequences may be the same, namely incompatibility and in its wake uncertain exchangeability, loss of self-evident interoperability (increased transaction costs), and possible market fragmentation², the benevolent setting of these deviations offers more leeway for action.

2. Framework and Methodology

If standard implementations are not interoperable, despite the best of intentions, the cause may lie in one or more of the phases leading up to standard implementation. Schematically speaking, the average standards committee starts with an idea, adopts it as a work item and then takes it through the successive stages of standardization (i.e. the standards process). A document results, i.e.: the standard specification. The standard is then implemented in a product or a service. The implementation process results in a standard implementation. See Figure 1.³ It highlights the three main states of a standard: the conceptual idea, the specification, and the implementation; and the two translation processes between these states: the standard or standard maintenance

¹ Egyedi & Hudson (2005) refer to instances where (de facto) standards are adapted, extended or selectively implemented as problems of *standard integrity* - that is, as a specific subset of compatibility problems.

² The classic example of market fragmentation is UNIX, the multi-user operating system that became a *de facto* standard in the late 1970s. Different versions developed that fragmented the market.

³ Figure 1 aims to help identify and localize causes of implementation problems. I.e. it does not portray a life-cycle model for standards. For a discussion of life-cycles, see Söderström (2004).

process, and the implementation process. The examples below illustrate in what manner each phase can be the source of implementation problems:

- The *idea* that underlies a standard may not be implementable (e.g. too comprehensive).
- The ideal of consensus decision-making may affect the *standards process* (e.g. lead to too many options) and, indirectly, the implementability of standards.
- Different use of terminology in a *standard specification* may lead to problems of interpretation, implementation and interoperability.
- Modest user requirements and cost-constraints in the *implementation process* may lead to partial standard compliance and incompatible implementations.

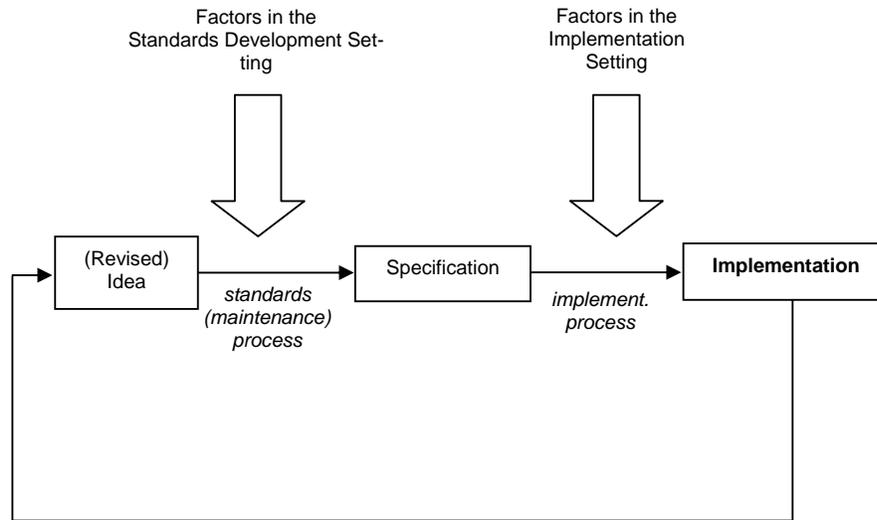


Figure 1: Schematic representation of the phases leading up to the standard implementation.

As figure 1 indicates, the immediate cause of incompatibility may primarily lie in the standard implementation setting, but the underlying causes may lie in factors that affect standard development.

Three complementary methodologies have been followed to gather data about possible causes of incompatibility. Firstly, to investigate whether certain institutional characteristics of standardisation can lead to problems of interpretation and implementation, formal standards policy documents and literature on standardisation have been analysed. Secondly, to gain insight in what happens in practice with standards from formal and other standards bodies, three case studies of ICT standardisation areas have been done, i.e., Standard Generalized Markup Language (SGML)/ Extensible Markup Language (XML), Open Systems Interconnection (OSI) standards, and

Unified Modelling Language (UML). From earlier studies these cases were known to highlight different implementation problems.

The third source was an expert panel discussion. The panel members, who were standardisation experts from formal standards bodies, standards consortia and industry⁴, were asked to discuss implementation problems from their personal experience and illustrate them.

The structure of the paper follows the methodologies used. The institutional analysis, which focuses on possible causes in the standard development setting, is discussed in section 3. The findings from the case studies are discussed in section 4. The findings from the panel of experts are presented in section 5. Section 6 summarizes the causes of incompatibility problems and discusses ways to solve them. The paper ends with conclusions (section 7).

3. Institutional Dilemmas

In the past, the policy of the formal standards bodies such as ISO was to focus on standards development, and on their role in supporting the democratic, voluntary consensus process. Formally, the question of implementation of standards lay outside their framework (Schmidt & Werle, 1998, p. 304).⁵ A policy shift took place in the mid 1990s which coincided with the rise of standards consortia and other ‘grey’ standards fora. Overall, these consortia prioritised the usability of standards, and treated standards development and implementation as co-evolutionary issues. See Table 1. For example, Internet standardisation, which at the time had an exemplary status, included demonstrated implementability in its standards process (IETF/ RFC 2026). More policy recognition of the importance of standards implementability and use by the formal standards bodies was discussed as one of the means to strengthen their position. However, the ideological rationale behind the formal policy made such a change difficult. At stake were two prominent ideals: the ideal of developing standards in a democratic, consensus-oriented manner, and the ideal of developing implementation-independent standards. Both ideals were and still are highly relevant, and are possible institutional causes of incompatible implementations.

⁴ Panel discussion, 22 October 2003, 17.00-18.30 hours, 3rd IEEE Conference on ‘Standardization and Innovation in IT’, October 22-24, 2003, Delft Univ. of Technology, the Netherlands. Panel Discussion ‘Problems of Standards Implementation’ with Jim Carlo, President-Elect IEEE-SA (Moderator), Oliver Smoot, ISO President (Commentator), and the panel members Patrick Droz, IBM, Manager Networking Software; John Hill, Chairman JTC1/SC22 Programming Languages, Sun Microsystems; Erik Huizer, IETF trustee of Internet Society, Univ. of Twente; Steven Pemberton, Chair W3C HTML and Forms working groups, CWI/W3C; Anthony Wiles, ETSI Protocol and Testing Competence Centre. In addition, some remarks made by Jim Isaak (IEEE board of directors) and Mostafa Hashem Sherif (AT&T) during the discussion are included in the text.

⁵ However, on the practical level conformance testing already took place in the 1980s (e.g. in ISO/ CCITT on X.25). (Linn & Uyar, 1994)

3.1 Consensus decisions

Regarding the first ideal, the emphasis on democratic, consensus-oriented decision making in committee standardisation sooner solicits political compromises. These, in turn, may result in standards with several options, or in vague formulations behind which opposing parties can nonetheless rally. Or, in more positive wordings, “(...) politics helps to achieve an agreement precisely because it does not consider technical details. (...) Terminating a conflict through the adoption of incompatible options (...) keeps the organization viable.” (Schmidt & Werle, 1998, pp. 303, 271, 270)

3.2 Implementation independence

The second ideal which sets formal standardisation apart from consortium and de facto standardisation is its concern for implementation-independent standards (see Table 1). The formal standards bodies strive for impartial standards solutions, that is, for solutions which do not favour certain companies, technologies or markets. Keeping these options open sooner leads to problems of implementability. Standards that need to cater to different application environments are inherently more complex (e.g. STEP; see Wapakabulo *et al.*, 2005) than standards that cater to a specific implementation environment (e.g. Internet). More options are needed to address the sometimes irreconcilable standards requirements. The generic applicability aimed for conflicts with the specificity needed to implement a standard in an unambiguous and consistent manner.

<i>Approach</i>	<i>Formal standardisation</i>	<i>Consortium standardisation</i>	<i>De facto standardisation</i>
<i>Standards process & implementations</i>	Successive occurrences/ Implementation outside scope	Parallel occurrences	Standards follow implementations/ Standardisation is by-product
<i>Implementation independence</i>	High	Medium	Low

Table 1: Typification of three standardisation approaches in the mid-1990s. (Source: adapted from Egyedi, 1996)

Although by and large the differences between the three standardisation approaches are currently still recognisable (e.g., the ideals of the formal standards bodies have remained the same), the procedures of the formal bodies and the most important consortia are converging. Conformance testing and - in the case of ETSI also - standards implementation have the attention of the formal bodies (e.g. ISO, 2001; ETSI, 2005a), while impartiality and consensus ideals have become part of consortium procedures (Egyedi, 2006).

4. Three Case Studies

The following case studies throw light on some additional causes of incompatibility. They address three clusters of standards, i.e.: SGML and XML, standards belonging to the family of the OSI reference model, and UML, a modelling standard used for system development.

4.1 SGML & XML

As the reader may know, the initial idea behind XML (W3C, 1998) was to bring the 'structured data exchange' functionalities of SGML (ISO 8879: 1988) to the web (Egyedi & Loeffen, 2002). In the web environment, XML was to succeed SGML. The aim was that XML would remain compatible with SGML. However, this was only partly achieved because XML documents could not be processed by SGML (1988) tools. The implementation problems addressed here are set against this background.

To address incompatibility between XML and SGML, two initiatives took place. First, an ISO/IEC (International Standardization Organization/ International Electrotechnical Commission) SGML working group drew up Technical Corrigendum 2 (Cor 2: 1999) that "(...) remedies defects revealed by the multiple adaptations of SGML for the World Wide Web (WWW), intranets, and extranets. The annex corrects errors, resolves ambiguities for which there is a clear resolution that does not cause existing conforming documents to become non-conforming, and provides a choice of alternative resolutions for other ambiguities. Although motivated by the World Wide Web, applicability of this annex extends to all uses of SGML." (SGML, 1999; annex K) Full implementation of the technical corrigendum was to make an SGML system XML compatible. However, in practice new software providers and standards implementers had no involvement with SGML. They immediately turned to XML rather than implement an elaborated SGML.

Second, the XML working group included non-binding recommendations in the standard. Implementation thereof was to allow XML documents to be processed by SGML (1988) software. However, the standard would not guarantee compatibility (i.e. implementation thereof only "increases chances" of interworking). In any case, many XML system designers ignored these guidelines.

The emphasis in the SGML standard has always been on its ubiquitous applicability. XML emphasises simplicity and implementability. Although the SGML standard has been successful in many ways and for a very long time in IT terms, the current wide adoption of XML suggests that in a web-based environment wide standards implementation requires simplicity.

4.2 OSI model

The Open System Interconnection (OSI) model is a standard reference framework. It was initiated to rationalise and integrate standards activities in the merging fields of

IT and telecommunications in the 1980s. It identifies Information and Communication Technology (ICT) services as consisting of a set of functions that are mapped onto seven layers (i.e. physical, data link, network, transport, session, presentation and application layer). Within these layers generic building blocks are specified, called base standards. Base standards can contain options. Problems arise if two service implementations are based on different options in base standards. This causes interoperability problems. To avoid this problem, the formal standards bodies (ISO/IEC JTC1 (Joint Technical Committee 1) and International Telecommunication Union's (ITU) CCITT) also standardised sets of specified base standards with fixed options for certain application areas (e.g. banking). These are called profiles or functional standards. A functional standard is a '... document which identifies a base standard or group of base standards, together with options and parameters, necessary to accomplish a function or a set of functions'. (ECITC, 1993)

Taking a closer look at the options in the base standards, for the transport layer protocol a compromise of five different protocol classes was defined. This complicated interworking. To alleviate interworking problems, means were developed to allow a certain amount of negotiation between protocol classes. In addition, profiles were developed that defined a fixed OSI protocol stack for specific applications, including the necessary transport protocol class. For example, the classes of TP0 and TP1 were prescribed for CCITT's message handling recommendation X.400. (Egyedi, 1997)

For the session layer, functional units were defined with overlapping functionalities. According to participants in the standards process, this was a political compromise. There was no viable technical reason for the overlap. The consequence of the overlap was that implementers of the session protocol usually implemented one or the other combination of functional units, and not both. That is, the session layer, too, gave rise to different OSI stacks (i.e. to fragmentation) - and to interworking problems.

In sum, OSI's objective of implementation- and field-independent standards was ambitious and came at a cost. According to some critics, the costs of implementation were too high. In their opinion OSI standards comprised much overhead, too many options, and complex answers to specific and simple needs. To cut down costs, OSI implementers sometimes omitted functionality's that were part of the standard. Nominally, OSI-compatible products resulted. In reality, only partial compliance existed. Partial implementations damaged OSI's reputation.

4.3 UML

The Object Management Group (OMG) unanimously adopted the Unified Modelling Language (UML) as a standard in November 1997 (OMG, 1998). The standard aimed to simplify and consolidate the large number of Object Oriented (OO) software developing methods that had emerged (e.g. Shlaer & Mellor, 1988; Coad & Yourdon, 1991; Booch, 1991); to reduce gratuitous divergence among tools; to encourage widespread use of OO modelling among developers; and to facilitate the development of a

robust market of support tools and training “now that neither user nor vendor have to guess which approaches to use and support”. (UML reference manual, 1998)

However, there are different types of inconsistencies in UML modelling. In modelling approaches consistent naming is a well-known requirement for avoiding impedance mismatches. Impedance problems arise when a unified naming convention is lacking within and across modelling techniques. UML comprises several complementary and substitutive modelling techniques (e.g. class diagrams, state transition models, activity models and functional models). These modelling techniques use different terminology for similar things, which (a) leads to misunderstandings between the parties involved in system development (e.g. developers, testers and users); (b) aggravates the problem of integrating information from one model into another during the system design stage – even apart from the problem which different terminology poses during system implementation; and (c) leads to difficulties in the traceability and re-use of components.

Related to the latter point, UML is not intended to be a complete development method. That is, it does not include a step-by-step development process. Originally, a companion book for UML-based system development, the Rational Unified Process (RUP) (Jacobson *et al.*, 1999) was proposed. However, according to experts, this work lacks the necessary vigour and freedom of modelling, and has been challenged by UML-based methodologies such as Select Perspective (Allen & Frost, 1998), Catalysis (D’ Souza & Wills, 1999), UNIFACE (2000), KobrA (Atkinson *et al.*, 2000) and CBD/e (Castek, 2000). Again, these methodologies usually produce similar functional solutions. However, they often cannot be exchanged or integrated. Also, they may differ completely in how they implement the system.

UML is more complicated than some of its antecedents because it aims to be more comprehensive. It incorporates several kinds of models. Normally, one does not need all UML modelling techniques for each project. Although experts know how to combine parts of UML, newcomers do not. Therefore, UML profiles are needed that indicate which combinations are useful in certain situations.

Lastly, lack of consistency in and interoperability of UML-based systems also plays at the level of the data model. An example best illustrates what is at stake. Let us suppose that ‘Student’ is a Class in the UML class diagram. Its real representation in the application area is an instance object with a name (e.g. S. Mohamed) and a number of other instance attributes. In the generic model, the Class ‘Student’ represents any student. But students may come from different countries. For example, while normally a year has 12 months, the Ethiopian year has 13 months. This poses a problem for representing the date attribute of Class ‘Student’. That is, if at instance level the data model is inconsistent, this will obstruct system interoperability. (Stojanovic *et al.*, 2001)

5. Panel of Experts

The third important source of information was a panel discussion among experts. Drawing on their personal experience, the experts and discussants confirm some of

the causes of incompatibility identified in the case studies, but also note several additional causes.

Complexity of comprehensive, ambitious standards. Some standards include too many details and unnecessary niceties ('bells and whistles'). The level of detail should take account of the maturity of the technology under consideration.

This is elaborated in standardisation literature. If a standard "(...) includes too many details at an early stage of the lifecycle [of the technology], the wrong aspects may be included. If standardisation takes place late in the life cycle but does not respond to specific end-user demands, it may be irrelevant." (Sherif *et al.*, 2005, p.225)

Ill-structured standards. Writing a standard specification is not trivial. Standards need to be complete, unambiguous, readable, well-structured, etc. It is not easy for people coming from non-standards environments to write standards. Perhaps they need help with what one of the panellists calls 'standards engineering': support for editors and rapporteurs in the application of good standards engineering practices based on experience.

The encryption algorithm for the DECT standard (Digital Enhanced Cordless Telecommunications) illustrates what can be at stake. DECT was a voluminous standard. It comprised at least eight different parts. The encryption algorithm itself was well defined, but the information about whether you enter the bit stream into this algorithm the least significant bit first or the most significant bit first was not easy to find. The answer was there in the standard, but not where one would have expected it to be. European manufacturers misunderstood the standard and implemented it the wrong way in - a lot of - hardware. Whereas Asian implementers perhaps spent more time reading the standard and interpreted it correctly. There were intense battles between the two camps, battles which could have been avoided by restructuring the standard and presenting the relevant information together.

Ambiguity of natural language. To cope with the ambiguity of natural language, complex protocol engineering uses formal specification languages (e.g. ASN.1: Abstract Syntax Notation One). (Additional advantages are that their use eases the testing and evaluation of these protocols, and that they add value where one can build models that can be validated and simulated.) Many ETSI (European Telecommunications Standards Institute) standards use ASN.1 successfully (e.g. GSM (Global System for Mobile Communications)). However, it is not always helpful (e.g. SMTP (Simple Mail Transfer Protocol) standardization). Sometimes the formal technique seems to acquire more emphasis than the specification itself (e.g. with SDL (Specification and Description Language)). Formal specification techniques should therefore be used with care, where needed, and with moderation.

Uncertainty about how to handle options. Standards with a narrow range and few options are more likely to be implemented correctly and interoperate. But they leave less room for product ingenuity. Standards with a broad range and many options leave more room for product ingenuity, but the chance of non-interoperability increases as a result of picking and mixing options. The products then conform to the standard but do not necessarily interoperate.

Options can be of a different kind. There are options with similar functionalities (i.e. there are three ways to do it - specify which way), and options with orthogonal functionalities (-if you are going to do 'A', do it this way). Lack of clarity about which type of option is at stake, leads to confusion.

Moreover, options are often not fully specified. Although standards usually indicate what to do if one implements an option, they seldom indicate what to do if an option is not implemented. Engineers then make their own decisions.

Complexity: standards overload. Apart from the complexity which stems from too comprehensive standards, lack of interoperability can also result from an overload of standards. For example, the large number of Internet standards (Request for Comments, RFCs, >100) for Internet Protocol Version 6 (IPv6) makes it difficult for engineers to pick out the RFCs necessary for their products. As ETSI interoperability tests show, IPv6 protocol stacks that have been implemented according to the RFCs still do not interoperate. Although the standards themselves are thought to be of good quality, there are too many of them.

'Bugward compatibility'. Sometimes standards are implemented incorrectly in order to preserve backward compatibility with a bug-ridden prior version of a product. One of the experts coins this phenomenon with the term 'bugward compatibility'. For example, Microsoft implemented CSS1 (Cascading Style Sheets 1) incorrectly in its early browsers, but insisted on downward compatibility in later versions so that the content produced for the early browsers would still work.

Missing details, monopoly on tacit knowledge. In DECT the wrong value was given to an extension bit in one of the information elements. However, engineers in field knew what the correct value should be, and widely implemented the standard based on that common understanding - all except for one large manufacturer who followed the standard. This caused interoperability problems, a mistake that was discovered during performance testing of the products. See also Moseley, Randall, & Wiles (2003).

There is always a need to rely on the implicit, common understanding of the implementation community. Because of time constraints, not every piece of information can be put into a standard. On the other hand, one also cannot always assume that everyone shares this understanding. Errors should be fed back into the standards body and corrected in the standard.

A slightly different situation arose with regard to a number of router protocols. Here, only a handful of people knew the details that were missing in the specifications. They had a monopoly on knowledge of the standard. The companies that were in the routing business had to hire one of them to get the protocol running.

Interference between standards. New standards sometimes hinder existing standards, standards that were implemented and were interoperable before.

Table 2: Causes of incompatibility categorised and where the cause originates.

CAUSES OF INCOMPATIBILITY	LOCUS C= conceptual idea SP= standard process S= standard spec IP=implem. process
Errors, ambiguities, inconsistencies	SP/S
Ambiguity of natural language	SP/S
Missing details, monopoly on tacit knowledge	S/IP
Ill-structured standards	S
Unclear how to handle options	S
Uncertain compatibility of non-binding recommendations	S
Complexity of comprehensive, ambitious standards	C
Too many options and parameters	SP/S/IP
'Bugward compatibility'	C
Unclear official status of standard's companion book	S
Single company pushing for standard; weak specs	SP
Overload of standards	C/IP
Deviation from and partial implementation of a standard	IP
Interference between standards	C/IP

6. Problems and Suggested Solutions

Table 2 summarises the causes of incompatibility mentioned in the previous sections and assigns them, where possible, to a specific state or process. Most problems are closely tied up with what the standard specification (S) looks like. Some are caused by a flaw in the design (conceptual idea; C) of the standard. Some result from a flaw in the process of drafting the standard (standards engineering process; SP). Although there are constraints of time and cost⁶, the standards setting can to a large degree determine whether the above problems are likely to occur. Indeed, some standards bodies - and standards committees – already place more emphasis on standards quality than others.⁷

⁶ The consequences which the measures suggested in the following have in terms of costs and delay in standard's delivery are difficult to estimate. They require further study.

⁷ The term 'quality' is used here pragmatically, namely in the sense of addressing the above-listed problems.) Whether quality or opening up a market should be the first priority is a matter for discussion. Is speed is more important than standards quality (i.e. maturity of a standard)? E.g., a qualitatively more mature Hyperlan standard (ETSI) lost its market to the WLAN standards of IEEE 802.11. Indeed, in the case of Parlay, e.g., the strategy was to quickly develop a standard in order to secure an early market share. At this stage speed was deemed more important than quality. Implementation feedback was used to improve the standard. (ETSI, 2005b)

6.1 Deliberation: Systematic or Random Causes

What can be done to reduce incompatibility between standard-based products? Some implementation problems would seem to be of a more structural kind (e.g. parallel options and ambiguities that result from political compromises); while others would seem to be more random and temporary (errors, accidental ambiguities, etc.). For example, the first version of a complex standard might be expected to include some errors. Such errors will be random in the sense that they will differ from those in a different standard and do not contain a systematic bias. However, random errors may also have a structural background. They may be random in where and how they are expressed, but systematic in *that* they occur. The search here is to find structural solutions for both random and structural problems.

Some structural problems, however, are rooted in fundamental dilemmas and are therefore very difficult to resolve. Box 1 summarises the main ones. The difficulty of solving them adequately needs to be recognized.

Box 1: Dilemmas regarding the Implementability of Standards

Some incompatibilities derive from a set of fundamental dilemmas, choices regarding the standard's design and use that can be summed up in four questions.

Standard's design:

- Comprehensive or simple standards?
- Implementation-independent or implementable standards?
- Consensus on a compromise or implementable standards?

Standard's use:

- Adapt a standard to ones simpler needs or aim for interoperability with other standard-compliant products?

These deliberations have to be faced by standardisers and implementers, respectively. The choice they make affects the implementability of standards.

6.2 Recommendations

Different standards bodies use different means to address causes of incompatibility. For example, ETSI uses a wide range of techniques to improve the standard development process, and validate and test standards (ETSI, 2005). The procedures followed by the World Wide Web Consortium (W3C) involves wide scrutiny (public comment via Internet), two implementations of every standard feature and a demonstration of their interoperability. Internet's IETF requires two and four independent implementations in the phases of proposed standard and draft standard, respectively. Standardizing bodies both (1) try to *prevent* incompatibility with, for example, interoperability tests (e.g. ETSI), reference implementations (e.g. IETF), and/or by making public the rationale that underlies the decisions of a technical committee (e.g. ISO); and, (2) address the causes of incompatibility retrospectively by means of defect reports, technical corrigenda, etc. during standards maintenance.

Table 3: Recommendations

Institutional measures towards reducing standard-based interoperability problems
<p>Drafting of standards</p> <ul style="list-style-type: none"> • provide institutional support for editors and rapporteurs on standards engineering • involve technical editors (i.e. not standard developers) • focused use of pseudo-code or formal languages (side-effect: this also eases the generation of test suites) • adopt a unified naming convention • clarify what type of options is involved (optional/mandatory, orthogonal/equivalent) • specify how to deal with options (e.g. profiles) • specify consequences of (not) implementing options (i.e. of a pick and mix in real products) • make explicit the rationale behind the specification • add a reference guide as part of the standard • organise wider scrutiny of standard • translate the standard (this uncovers ambiguities) • coordinate the standard initiatives of different standards bodies (for compatibility among interrelated standards)
<p>Pre-implementation</p> <ul style="list-style-type: none"> • validate standards before they are implemented in products (“walk throughs”) • develop a reference implementation / pre-implementations • develop a reference environment • include testing (standard conformance and interoperability testing) • organise interoperability events with products from different vendors (e.g. plug tests) • organise opportunities for dialogue between standard developers and implementers
<p>Post-implementation</p> <ul style="list-style-type: none"> • improve consistent standard’s use and standards’ integrity with help of e.g. compliance and interoperability conformance statements, compatibility logos, certification programs • supply test suites
<p>Standards policy</p> <ul style="list-style-type: none"> • prioritise implementability as a standard’s requirement • reconsider desired level of consensus across all areas

Table 3 summarises the solutions discussed in the standardization literature, and proposed by the panel experts. Four categories of recommendations are distinguished: Drafting of standards, Pre-implementation, Post-implementation, and Standards policy. See Table 3. The most significant category of recommendations concerns the Drafting of standards. There are many ways to improve the drafting of standards. For

example, to deal with the ambiguity of natural language formal specification languages can be used. But also: Translating international standards into other languages often discloses ambiguities. In these cases making public the rationale which underlies the decisions of a standards committee and reference implementations help implementers to resolve ambiguities.

Regarding standards options, the experts recommend that

- standards be very explicit about which options are mandatory and which are optional;
- optional requirements be fully specified;
- implementers acquire instruments to handle options systematically (e.g. profiles);
- the implications of a pick and mix in real products be made clear.

For the readability of the standard it might be better to install a technical editor, rather than a person who has been involved with the development of the standard. Or, alternatively, to offer more support for the ‘involved’ editor. To ascertain compatibility among interrelated standards, coordination between the different standards committees is needed.

Lastly, standards policy might want to

- reconsider procedures such as the use of consensus decision-making irrespective of the standards scope (i.e. focused majority voting for some areas, which would reduce the need for political compromises).
- prioritise the implementability and validation of standards. This would require the systematic inclusion of standards conformance and interoperability testing in the standards process, and a complementary market-oriented testing and certification program (including e.g. the availability of test suites and plug test events).

7. Conclusion

What causes lack of interoperability? Although the immediate problem usually lies in the way standards are implemented, the underlying causes are usually flaws in the scope of standardization, in the standards process or in the specification itself. Therefore, most recommendations made in the previous section focus on improving the drafting of standards and their validation prior to implementation. To a lesser degree the proposed measures intervene in the implementation process or post-implementation phase.

To conclude, many standardizing organisations neglect standard implementation issues because this is argued to be a matter best left to the market. However, standards development and implementation are intertwined in their impact. In this respect, both areas of activity cannot be meaningfully separated. Standardizing organisations are therefore recommended to shift their emphasis from standard development to a more systematic inclusion of implementation concerns, both at the technical level of standard committees and at the policy level of standard organizations.

Acknowledgements

I am very much indebted to Sun Microsystems for funding this strain of research, to the NO-REST EU project partners for feeding me with new insights and providing a context in which to pursue writing on the subject, to Ajantha Dahanayake for her knowledgeable input on UML (Egyedi & Dahanayake, 2003), and last but not least to the SIIT 2003 panellists and discussants Jim Carlo, Oliver Smoot, Patrick Droz, John Hill, Erik Huizer, Steven Pemberton, Anthony Wiles, Jim Isaak and Mostafa Hashem Sherif for sharing their valuable experience on the subject.

References

- Allen, P., & S. Frost (1998). *Component-Based Development for Enterprise Systems: Applying the Select Perspective*. Cambridge: Cambridge University Press.
- Atkinson, C., Bayer, J., Laitenberger, O., & J. Zettel (2000). 'Component-based software engineering: The Kobra approach.' The 3rd int. Workshop on component-based software engineering. Limerick, Ireland.
- Booch, G. (1991). *Object oriented Analysis and Design with Applications*. Redwood City, Calif.: Benjamin/Cummings, 1st ed.
- Castek (2000). *Component-based development: the concepts, technology and methodology*. Castek company's white paper. <http://www.castek.com>.
- Coad, P. & E. Yourdon (1991). *Object Oriented Analysis*. Englewood Cliffs, N.J.: Yourdon Press, 2nd ed.
- ECITC (1993). *ECITC Guide to IT&T testing and certification*. Brussels, Belgium.
- Egyedi, T.M. (1996). *Shaping Standardization: A study of standards processes and standards policies in the field of telematic services*. Dissertation. Delft, the Netherlands: Delft University Press.
- Egyedi, T.M. (1997). Examining the relevance of paradigms to base OSI standardisation. *Computer Standards & Interfaces*, 18, pp. 431-450.
- Egyedi, T.M. (2006). 'Beyond Consortia, Beyond Standardization? Redefining the Consortium Problem', in K. Jakobs (Ed.), *Advanced Topics in Information Technology Standards and Standardization Research*, Vol. 1, pp.86-104.
- Egyedi, T.M., & A. Dahanayake (2003). 'Difficulties Implementing Standards'. In: Egyedi, T.M., Krechmer, K., & K. Jakobs (Eds.), *Proceedings of the 3rd IEEE Conference on Standardization and Innovation in Information Technology, SIIT 2003, October 22-24 2003, Delft, the Netherlands*, pp.75-84.
- Egyedi, T.M. & J. Hudson (2005). 'A Standard's Integrity: Can it be Safeguarded?', *IEEE Communications Magazine*, 43/2, 151-155.
- Egyedi, T.M., and Loeffen, A.G.A.J. "Succession in standardization: grafting XML onto SGML," *Computer Standards & Interfaces* (24:4) 2002, pp 279-290.
- ETSI (2005a). Making better standards. <http://portal.etsi.org/mbs/>
- ETSI (2005b). Lectures by Jørgen Friis (ETSI) & Zygmunt A Lozinski (Parlay) "Impact of Standards?! - New Insights", NO-REST conf., 27 May 2005, ETSI, Sophia Antipolis, France.
- ISO (2001). *ISO Strategies 2002-2004: Raising Standards for the World*. ISO/Gen 15:2001. <http://www.iso.org/iso/en/aboutiso/strategies/isostrategies2002-E.pdf>

- Jacobson, I., Booch, G., & J. Rumbaugh (1999). *The Unified Software Development Process*. Reading, Mass.: Addison-Wesley.
- Jakobs, K. (2005). Installation of an IEEE 802.11 WLAN in a Large University Setting. Proceedings of the 4th International Conference on Standardization and Innovation in Information Technology, September 21-23, 2005, ITU, Geneva, Switzerland. Aachen, Germany: Wissenschaftsverlag Mainz, pp. 171-184.
- Linn, R.J., & M.U. Uyar (1994, Eds.). *Conformance Testing Methodologies and Architectures for OSI Protocols*. IEEE: Computer Society Press.
- Moseley, S., Randall, S., & A. Wiles. (2003). 'Experience within ETSI of the Combined Roles of Conformance Testing and Interoperability Testing.'. In: Egyedi, T.M., Krechmer, K., & K. Jakobs (Eds.), Proceedings of the 3rd IEEE Conference on Standardization and Innovation in IT, SIIT 2003, October 22-24 2003, Delft, the Netherlands, pp.177-190.
- OMG (1998). Unified Modelling Language Specification. Framingham, Mass.: Object Management Group, Internet: www.omg.org.
- SGML (1999). ISO 8879:1986/Cor 2:1999. Geneva, Switzerland: ISO.
- Sherif, M.H, T.M. Egyedi & K. Jakobs (2005). 'Standards of Quality and Quality of Standards for Telecommunications and Information Technologies', in: T.M. Egyedi & M.H. Sherif (Eds.). Proceedings of the 4th International Conference on Standardization and Innovation in Information Technology, September 21-23, 2005, ITU, Geneva, Switzerland. Aachen, Germany: Wissenschaftsverlag Mainz, pp.221-230.
- Shlaer, S., & J.M. Stephen (1988). *Object-Oriented Systems Analysis: Modelling the world in data*. Englewood Cliffs, N.J.: Yourdon Press.
- Schmidt, S.K., and Werle, R. *Co-ordinating technology: Studies in the international standardization of telecommunication* MIT Press, Cambridge, 1998.
- Souza, D.F. D', & A.C. Wills (1999). *Objects, Components and Frameworks with UML: The Catalysis Approach*. Reading, Mass.: Addison-Wesley.
- Söderström, E. (2004). *B2B Standards Implementation: Issues and Solutions*, Dissertation, Department of Computer and System Sciences, Stockholm University, Sweden. Edsbruk: Akademityck AB.
- Stojanovic, Z. & Dahanayake, A. & Sol, H. (2001), *A Methodology Framework for Component-Based Systems Development Support*. Proceedings of the 6th CAISE/IFIP8.1 International Workshop on Evaluation of Modelling Methods in Systems Analysis and Design.
- Uniface (2000). Compuware Corp. Uniface products, <http://www.compuware.com/products/uniface/>.
- W3C (1998). Extensible Markup Language (XML) 1.0. W3C Recommendation 10 February 1998. Editors: T. Bray, J. Paoli & M. Sperberg-McQueen.
- Wapakabulo, J., Dawson, R., Proberts, S., & T. King (2005). A STEP towards the adoption of Data-Exchange Standards: a UK Defense Community Case. in: T.M. Egyedi & M.H. Sherif (Eds.). Proceedings of the 4th International Conference on Standardization and Innovation in Information Technology, September 21-23, 2005, ITU, Geneva, Switzerland. Aachen, Germany: Wissenschaftsverlag Mainz, pp.255-266.

Abbreviations

ASN.1	Abstract Syntax Notation One
CCITT	Comité Consultatif International Télégraphique et Téléphonique (ITU)
CSS	Cascading Style Sheets
DECT	Digital Enhanced Cordless Telecommunications
ETSI	European Telecommunications Standards Institute

GSM	Global System for Mobile Communications
ICT	Information and Communication Technology
IEC	International Electrotechnical Commission
IEEE	Institute of Electrical and Electronics Engineers
IETF	Internet Engineering Task Force
IPv6	Internet Protocol Version 6
ISO	International Standardization Organization
ITU	International Telecommunication Union
JTC1	Joint Technical Committee 1 of ISO/IEC
OSI	Open Systems Interconnection
OMG	Object Management Group
RFC	Request for Comments (Internet)
SDL	Specification and Description Language
SGML	Standard Generalized Markup Language
SMTP	Simple Mail Transfer Protocol
UML	Unified Modelling Language
WLAN	Wireless Local Area Network
WWW	World Wide Web
W3C	World Wide Web Consortium
XML	Extensible Markup Language