TECHNICAL REPORT

ISO/TR 24529

First edition
2008-04-15

Intelligent transport systems — Systems architecture — Use of unified modelling language (UML) in ITS International Standards and deliverables

Systèmes intelligents de transport — Architecture de systèmes — Emploi du langage de modélisation unifié (UML) dans les Normes internationales ITS et produits livrables

Reference number
ISO/TR 24529:2008(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) is a worldwide federation of national standards bodies (ISO member bodies). The work of preparing International Standards is normally carried out through ISO technical committees. Each member body interested in a subject for which a technical committee has been established has the right to be represented on that committee. International organizations, governmental and non-governmental, in liaison with ISO, also take part in the work. ISO collaborates closely with the International Electrotechnical Commission (IEC) on all matters of electrotechnical standardization.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of technical committees is to prepare International Standards. Draft International Standards adopted by the technical committees are circulated to the member bodies for voting. Publication as an International Standard requires approval by at least 75 % of the member bodies casting a vote.

In exceptional circumstances, when a technical committee has collected data of a different kind from that which is normally published as an International Standard ("state of the art", for example), it may decide by a simple majority vote of its participating members to publish a Technical Report. A Technical Report is entirely informative in nature and does not have to be reviewed until the data it provides are considered to be no longer valid or useful.

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. ISO shall not be held responsible for identifying any or all such patent rights.

ISO/TR 24529 was prepared by Technical Committee ISO/TC 204, *Intelligent transport systems*.

Introduction

The objective of this Technical Report is to provide guidance on the use of the “Unified Modeling Language” [UML] in the development of standards for “Intelligent Transport Systems” [ITS].

The advantages of applying UML to the development of ITS include the following:

— UML provides an Internationally Standardized form of system model that should be readily interpreted anywhere world-wide;

— UML enables cohesive description from multiple user views;

— There is available extensive training and tool support for UML;

— UML is capable of manipulation by a metadata registry for ITS;

— UML tools enable conversion directly to computer coding;

— UML is very widely used in the architecture, design and development of software-intensive systems.

The disadvantages of using UML include the following:

— UML is not understood by many stakeholders who are not also software developers;

— UML uses a larger amount of unfamiliar language and jargon which, while it may be necessary for precision, is daunting and off-putting to the non specialist and lay reader;

— UML is not yet developed enough to support the full scope of systems engineering;

— UML is still under active development and therefore the compatibility of UML models may be an issue.

There are therefore some risks in using UML but nevertheless the benefits are widely judged as exceeding the disadvantages. This document is intended to provide guidance to stakeholders who are considering the use of UML for ITS.

Intelligent transport systems — Systems architecture — Use of unified modelling language (UML) in ITS International Standards and deliverables

1 Scope

The scope of this Technical Report is the use of UML within International Standards Technical Specifications and Technical Reports and related documents.

This Technical Report discusses the application of the “Unified Modelling Language” [UML] to the development of standards within the context of “Intelligent Transport Systems” [ITS].

2 Normative references

ISO 14813 (all parts), *Transport information and control systems — Reference model architecture(s) for the TICS sector (Parts 1 to 6)*

ISO 14817, *Transport information and control systems — Requirements for an ITS/TICS central Data Registry and ITS/TICS Data Dictionaries*

ISO/TR 17452, *Intelligent transport systems — Using UML for defining and documenting ITS/TICS interfaces*

ISO/IEC 19501, *Information technology — Open Distributed Processing — Unified Modeling Language (UML) Version 1.4.2*

ISO/TR 25102, *Intelligent transport systems — System architecture — ‘Use Case’ pro-forma template*

3 Terms and definitions

For the purposes of this document, the following terms and definitions apply.

3.1
actor
coherent set of roles that users of an entity can play when interacting with the entity.

NOTE An actor may be considered to play a separate role with regard to each use case with which it communicates.

In the metamodel, ‘Actor’ is a subclass of ‘Classifier’. An ‘Actor’ has a ‘Name’ and may communicate with a set of ‘UseCases’, and, at realization level, with ‘Classifiers’ taking part in the realization of these ‘UseCases’. An ‘Actor’ may also have a set of ‘Interfaces’, each describing how other elements may communicate with the ‘Actor’.

An ‘Actor’ may have generalization relationships to other ‘Actors’. This means that the child Actor will be able to play the same roles as the parent ‘Actor’, that is, communicate with the same set of ‘UseCases’, as the parent ‘Actor’.

3.2
architecture
⟨ITS⟩ set of concepts and rules for an intelligent transport system describing the inter-relationship between entities in the entire system, independent of the hardware and software environment

NOTE Architecture is described through a series of viewpoints that may be at varying levels of generality/specificity, abstraction/concretion, totality/component and so on. See also communications architecture, logical architecture, organizational architecture, physical architecture, reference architecture and system architecture definitions below.

3.3
communications architecture
framework that tells designers how elements of hardware and software are to operate in harmony using common protocols and air interface techniques (where applicable)

3.4
logical architecture
definition of the processes (the activities and functions) that are required to provide the required 'User Services'

3.5
model
representation of an entity from which the important elements have been abstracted by removing unimportant detail while at the same time retaining the interrelationship between the key elements of the whole

NOTE A model may be made more or less abstract by the successive suppression of detail such that the concepts and relationships come into enhanced focus and become more readily understood. However the process can be taken too far when the simplification has exceeded the threshold where a necessary understanding may be achieved. Thus the process of modelling is one of going only far enough to achieve the optimum understanding and insight — and no further.

NOTE A model is a way of representing something, other than in its natural state. (See 'Models of ITS' documents at http://www.frame-online.net/library.htm).

3.6
organizational architecture
framework into which business processes are deployed and ensures that the organization's core qualities are realised across the business processes deployed within the organization

NOTE In this way organizations aim to consistently realise their core qualities across the services they offer to their clients.

3.7
physical architecture
high-level structure around the processes and data flows in the logical architecture

NOTE The physical architecture defines the physical entities (subsystems and terminators) that make up a system.

3.8
reference architecture
list of functions and some indication of their interfaces (or APIs) and interactions with each other and with functions located outside of the scope of the reference architecture

3.9
relation
relationship
nature of how two entities affect each other including dependencies

3.10
requirement
statement of user need, typically expressed in a single-sentence form to assist with later verification of compliance

3.11
scenario
sequence of steps that are taken to change the state from that before the scenario to that immediately after the scenario

3.12
system architecture
system architecture is a framework for ITS deployments; it is a single, high-level description of the major elements or objects and the interconnections among them

NOTE System architecture provides the framework around which the interfaces, specifications and detailed system designs can be defined. An architecture is not a product design, nor a detailed specification for physical deployment. [Mil4]

3.13
template
framework that may be used repeatedly to meet requirements that are similar to some extent

3.14
use case
representation of a series of interactions between an outside entity and the system, which ends by providing business value

NOTE A use case is a sequence of actions that an actor (usually a person, but perhaps an external entity, such as another system) performs within a system to achieve a particular goal [Rosenberg].

4 Abbreviated terms

ITS
intelligent transport system

KAREN
keystone architecture required for European networks

MDD
model driven developments

OMG
object management group

POM
process oriented methodology

SEI
software engineering institute

TICS
transport information and control systems

TR
technical report

UML
unified modelling language

5 Background

5.1 TC 204 working group 1 (WG 1)

This Technical Report arose from work by WG 1 on the elaboration of ITS architecture in the ISO 14813 series of International Standards and Technical Reports. It had become apparent that there was concerted opposition from some sources to the uniform use of UML for ITS architecture. While WG 1 has never mandated the sole use of UML above other architecture methodologies, and the 14813 series focuses on consistency and interoperability rather than preferring any one modelling technique, UML is seen as an increasingly useful tool in a developing and changing sector because of its ability to change and adapt without abandoning previous work and having to start again. Additionally, the ability to present a cohesive model from different perspectives (views) is seen by many to be useful in many situations. Finally, UML is an ISO International Standard and therefore one of the modelling techniques that should be supported. WG 1 therefore agreed the need for a Technical Report [TR] to discuss the applicability, strengths and weaknesses and recommended best practice for the use of UML in ITS standards.

5.2 UML as a standard

UML is very widely used as a specification and modelling language for software-intensive systems. This wide use of UML is recognised in the publication of the ISO/IEC standard 19501 that addresses UML.

The development of UML is continuing and has now reached version 2.0 with added features and the prospect of increasing support for model-based development (also known as model-driven development or MDD). This progress towards greater automation, particularly with the development of web services, is important for ITS standards.

5.3 Modelling for ITS architecture

The need for multiple viewpoints in architecture models has been widely recognised The similar need for a layered architecture reflecting differing levels of abstraction and encapsulation is also widely agreed. What is not agreed yet is the preferred manner in which these complementary model artefacts should be expressed.

The heterogeneous result is a proliferation of ITS architectural models expressed in a mixture of notations and formats. This has been considered acceptable for human comprehension — up to the point where complexity, confusion and misinterpretation can arise. However this approach is acceptable when only manual modelling and interpretation is involved. In the future this work will be of such complexity that it will be desirable to employ automation in various forms.

At that time the existence of multiple formats and inconsistently applied rules in manual approaches will not support automation, and hence standardization is needed.

These needs are recognised by ISO/TC 204 in their Technical Report/Draft International Standard ISO, by the ISO 14813 series of International Standards and Technical Reports determining the domains, service groups and services of the ITS sector, providing example elaborations and reference models and tutorials and specifying standardized data definition to enhance interoperability, and by the International Standard ISO 14817, *Transport information and control systems — Requirements for an ITS/TICS central Data Registry and ITS/TICS Data Dictionaries*.

UML is not the only available solution on offer. Other computerised architecture modelling techniques exist and are recognised to also have advantages and disadvantages compared with UML. Generally, while most computerised architecture modelling systems provide advantage over manual techniques, the disadvantage of most computer driven modelling schemes is their inflexibility in the event of change. The ITS sector is still young and is evolving and developing, often in unforeseen directions, at a rapid pace. UML provides flexibility and is well adapted to this environment. (Proponents of other methodologies will quite rightly claim their differences and, in some circumstances, advantages over UML, and it is not the objective of this Technical Report to claim preference for UML, simply to say that it is a suitable technique and to consider the ways of using it most effectively.)

UML is likely to provide the basis for much architecture based standardization because it is already so widely used and supported and is codified as the International Standard ISO/IEC 19501. At the end of the day ITS system design finishes up to a large extent as computer coding, and the closer an architecture model can get to the coding level, the easier and quicker the implementation, and the greater the probability that the final system will reflect the system conception and design. As UML is increasingly taught to new generations of systems designers and developers, any alternative approach that does not approach or reach the coding level, and have an ability to adapt and change, will face increasing difficulties once implementation phases arrive. To adopt an idiosyncratic approach for the single domain of ITS, the choice of some, ignores the fact that ITS is a field that is still very small by comparison with many other domains, particularly IT focussed domains, and sector idiosyncratic approaches increasingly seem improbable, and systems are most likely to be designed either using UML or the traditional engineering based process oriented decomposition model.

The need for interoperability between ITS systems, particularly at the base communication and data exchange levels has been recognised in the adoption by ISO/TC 204 of ASN.1 for interoperable machine-readable specification of data concepts in its Technical Report/Draft International Standard ISO/TR 14813-6 (Use of ASN.1 in ITS Standards). A similar need will arise for the many other aspects of rich architectural models if they are to be amenable to meta-modelling and manipulation. Thus there is benefit in thinking ahead to when ITS architecture specifications and standards will be used in a semi-automated fashion, perhaps through use of a data registry as is currently being piloted by the Highways Agency in the UK.

This does not preclude use of other notation — such as entity-relationship diagrams, data-flow diagrams, Markov chains, state-transition diagrams or Petri nets — but this proliferation will only make automation more difficult later.

6 Discussion

6.1 Scope of the discussion

“Intelligent Transport Systems” (ITS) generally comprise a distinguished form of a large-scale, distributed information system deployed in some form of tangible or virtual network.

NOTE Those forms of ITS which are not monitored, controlled or managed in a networked fashion are not considered further in this Technical Report. They are generally special-purpose devices that are used in an autonomous role rather than systems in the broad sense. However, discrete ITS will generally be subject to increasing levels of integration over time.

This Technical Report addresses the specific needs for systems architecture of heterogeneous, distributed, networked systems that are designed, developed and deployed in ITS applications and which may interface to a wide range of existing systems — both ITS and non-ITS used in an ITS context (for example cellular telephony, Bluetooth, etc.). For this development process to be undertaken effectively, it is customary to define requirements, undertake analysis and then to develop an architectural design, before moving into detailed design and implementation.

6.2 What is systems architecture?

Systems architecture in its most fundamental form is the description of an intended complex system that includes the major features and interfaces together with sufficient detail for the interfaces between sub-systems and to other external entities. The architectural entities and interfaces must be described or specified sufficiently for their intended behaviour to be understood sufficiently by stakeholders to support their approval to proceed with implementation.

Systems architecture identifies the major actors, interfaces and components and provide a basis to understand all their inter-relationships and interactions

Systems architecture for ITS has frequently been described as comprising several major viewpoints, such as:

— Reference architecture;

— Logical (sometimes called functional) architecture;

— Physical architecture;

— Communications architecture;

— Organizational architecture.

This Technical Report asserts that in general practice there may be several other viewpoints needed to fully comprehend an architectural model and the three forms (see Clause 3) of architecture specified for ITS is an unnecessary, and possibly unhelpful, restriction. What matters most is that the composite description satisfies all user and interface requirements, all non functional requirements and provides a rigorous basis not only for the initial design, but also for the ongoing development of the system as it evolves and as it interacts in new ways with its environment.

> ‘Software architecture involves the descriptions of elements from which systems are built, interactions among those elements, patterns that guide their composition, and constraints on those patterns.’ (Shaw 1996).

'The software architecture of a program or computing system is the structure or structures of the system, which comprise software elements, the externally visible properties of those elements, and relationships among them.' (Bass 2003).

Architectures may be described and defined in many different ways, for example the ODP-RM (ISO/IEC 10746) of (Putman 2001), (Hofmeister 2000) or (Clements 2003) of the SEI. Differing descriptive formats and notations may be used in these descriptions but the notation that is being adopted most rapidly and widely is the Unified Modelling Language [UML] from the OMG.

NOTE Although much of the descriptive work on systems architecture addresses software, the same approach applies to hardware and other forms of tangible and intangible structure and processing in ITS. The reason for the rapid expansion of systems architecture practice is the urgent need to deal with the complexity involved in systems evolution.

6.3 Why is architecture relevant?

The development of an architecture has been recognised as being a critical stage in the development of any complex system such as a networked ITS. The key issues that are addressed by an architecture include:

— The partitioning into logical or functional entities such as sub-systems, modules or components;

— The allocation of responsibilities for system behaviour within constraints (stated or implied) to the available entities and/or to external entities;

— The identification of functional interfaces and other relationships among the logical entities;

— The system modes of operation including degraded and alternative modes;

— Performance constraints and enablers and how these may be characterised;

— Levels of service that will be supportable;

— Reliability, maintainability, availability and safety of the system as a whole;

— The basis for evolution through extension, integration or substitution of entities and/or interfaces.

The overall levels of interconnectivity and interoperability that can be achieved and what risks and constraints may apply to these.

This Technical Report asserts that the relevance of a systems architecture is demonstrated by inspection of the list above with respect to any practical ITS in use or in prospect.

6.4 How is interoperability defined and realised?

An often stated goal of using an ITS architecture is the enhancement of interoperability.

Interoperability is a much-admired attribute of distributed systems but it is also subject to much interpretation. For example take the following definitions of interoperability:

— An ITS view from TC 204: 'Interoperability: The ability of systems to provide services to and accept services from other systems and to use the services so exchanged to enable them to operate effectively together.' [TC 204 document N271 quoted in (McQueen 1999) and taken from the then ISO/TR 14812 Glossary of ITS terms (deleted project)].

— Contrast this with a definition from the "Australian Logistics Council" [ALC]: 'Interoperability: The ability for partners to coordinate information and processes, especially across an electronic network.' (ALC 2002)

— The KAREN project agreed on the following definition: 'Two or more systems are interoperable if they can pass data between each other to their mutual benefit, i.e. to provide harmonious and/or complementary functionality; interoperability includes the technical, operational and organizational aspects.'

Clearly there is potential for misunderstanding with such variation in expression of such a key concept. It must be carefully specified to be of practical benefit; otherwise it remains a vague expression of good intent lacking any means of application or even measurement.

The enhancement of interoperability is the critical motivation for a number of companion TRs including ISO/TR 17452 which discusses the use of UML for ITS interfaces. It should be noted that interfaces are often addressed as being the most visible aspects of ITS in its context or environment and also the point of greatest clarity and control of behaviour. Nevertheless it is critical to recognise that a system cannot be characterised nor its behaviour specified solely by the interface protocols, but also requires interoperability and reuseability of data concepts/data concept values.

6.5 What are the desired levels of interaction with ITS architecture

Interaction of stakeholders with ITS architecture may be considered at four levels as shown in Table 1:

Table 1 — The Levels of stakeholder involvement with ITS architecture

Level	Comprises	Description
1	Community, societal and governmental policy level	At this level the issues are about what are the fundamental user needs in terms of the community, government, and general public — both collectively and individually.
2	Business and organizational management level	At this level the issues derive from consideration of resources, priorities, risks and the implications of new and changing requirements from Level 1. This level also includes the business analysis level responsible to provide clear specifications of business requirements. This level may also seek extensive insight into the system domain through research and academic investigations into land-use, transport and other related planning issues.
3	Professional management; architectural specification and acceptance	This level is dealing directly with the developers and integrators whether they be in-house or under contract. At this level the business requirements will be interpreted and extended into extensive statement of technical requirements — both functional and non-functional.
4	Practitioners and professional developers, integrators and maintainers	Generally these will be expert systems and software development, test and deployment professionals with extensive specialist knowledge of available components, design patterns and applicable standards.

This Technical Report examines the implications of providing architectural design information in an appropriate form for the groups of people associated with each level.

6.6 What are use cases and why apply them?

At Levels 2 and 3, a current means to describe the business requirements in a concise and intuitive manner is to apply so-called ‘Use Cases’ (Kulak 2004) (Rosenberg 1999). A ‘UseCase’ is defined by Kulak as simply a 'representation of a series of interactions between an outside entity and the system, which ends by providing business value’. A use case will typically include the following information regarding a scenario for the required needs to be met through action to be undertaken as described in the example use case taken from ISO/TR 25102, for example:

USE CASE TEMPLATE / OUTLINE TABLE OF CONTENTS	
Use Case Name	
“Use Case” Description	
“Use Case” Scope	
Target System Release	
Generality/Abstraction level	
“Use Case” Author	Name: Phone: Email: @
“Use Case” Stakeholders	
“Use Case” Goal	
“Use Case” Requirements Reference	
“Use Case” Assumptions	
“Use Case” Technology Restrictions	
Relationships to other “Use Case(s)”	
Actors for this “Use Case”	
“Use Case” Triggers	
“Use Case” Pre-conditions	
“Use Case” Scenario #Scenario.	REPEAT AS NECESSARY TO COVER ALL SCENARIOS
“Use Case” Step #Scenario.#Step	REPEAT AS NECESSARY TO COVER ALL SCENARIOS
“Use Case” Expected Outcomes	
“Use Case” Acceptance Criteria	
“Use Case” Exceptions	
Post-conditions	
“Use Case” Extensions	
“Use Case” Inclusions	
User Interfaces for “Use Case(s)”	
“Use Case” Business Rules	
“Use Case” Verification Approach	
“Use Case” Test Cases	
“Use Case” Version	
“Use Case” Modification History	
Template Version	
Template Modification history	
Approvals	
Application Notes:	
Open Issues	

Figure 1 — Example of a use case template (ISO/TR 25102)

A companion deliverable, ISO/TR 25102, provides more detailed explanation of a use case pro-forma template for ITS.

6.7 What is the "Unified Modelling Language" (UML) and why does it seem so daunting?

The UML is being used increasingly worldwide to describe the static and dynamic logical design and behaviour of complex systems — see (Holt 2001) for example. This has come about because of the wide recognition of its advantages and de facto universality in describing software intensive systems in a manner that is understandable across cultures, companies and customs.

UML does provide for user requirements, when encapsulated in use cases, to be related to other requirements in other use cases and also to other UML artefacts (when suitable computer-aided software engineering [CASE] tools are employed). However it is not necessary for use cases to be expressed in UML for them to be meaningful and unambiguous as is also discussed in ISO/TR 25102.

Unfortunately many other aspects of UML may seem arcane, full of unusual words and jargon and daunting to most people in the Level 2 grouping and even to some members of Level 3 (of Table 1 above), enough for them to turn away from it. On the other hand Level 4 developers and implementers may well *expect* that UML will be used unless their client specifically requires otherwise, since UML is becoming the de facto standard for this purpose at software design level. This gives cause for concern.

The primary reason for the difficulty that many people have with UML is its deceptively simple notation compared with the rich interpretations of the significance of the artefacts when produced.

Of special note is the use of 'abstraction' as an approach to design at every level, but especially at the architectural design level. Abstraction has been described as follows:

> 'Abstraction is perhaps the most powerful tool available to a software engineer. Abstraction aims at reducing detail, making the thing that has been subjected to abstraction simpler to handle… Abstractions usually build on lower-level abstractions, leading to a layered, hierarchical design'. (Szyperski 1998)

However, to a task oriented client project manager, whose pressure is to get the job done and commissioned on time and to budget, 'abstraction' may seem an esoteric luxury rather than a key tool to best enable him to achieve his objectives.

6.8 Where do International Standards fit into the equation?

The topic of International Standards is often mentioned in connection with system architecture. The significance of standards is generally agreed but their relationship to architecture is not readily apparent. Standards have been formulated for use at many levels of a concern — this may align with the levels of involvement described earlier, or the levels of concern may relate to the levels of abstraction or the layers of an architecture that are in use.

For example, in a typical layered architecture, the most abstract layer (usually shown at the top of the 'stack' of layers) is generally the service or application that is of interest at interaction Levels 1 and 2 (and possibly 3) of Figure 1 above. At the other extreme is the so-called physical layer that deals with tangible media such as copper, optical fibre or wireless links carrying electronic and radio-frequency communication. A multitude of standards, both International Standards and regional or national standards have been defined and agreed and are in wide use for each of the layers and for many related aspects of their use.

However the layering of the architecture and the associated standards are necessary but not sufficient to define the system in sufficient detail to approve its design and to implement it with confidence. Other viewpoints must also be considered, and not all of them have associated standards… at least not yet. In other words there is always scope to define new standards to deal with new relationships, interfaces and systems or to describe new views of those same entities.

A further confusion with architecture standards is the well understood and popular desire to design a one-size-fits-all solution. This may be appropriate for a national or regional architecture within a given socio-political framework, but is not an appropriate subject for a standard. This then causes frustration: 'why isn't there a Standard' ITS architecture' as if failure to produce one is a failure of the architecture process, rather than a recognition of the fact that socio-political frameworks differ between countries, and in large countries may differ within regions, pan-continental or global architectures can only at best exist at the highest level, and

provide tools for developers and implementers to use architecture to develop their solutions, and where appropriate list elements of the required solution from other implementations.

Thus the role of the standards developer in the field of architecture is not to try to find the elusive one-size-fits-all architecture, but to provide standards, specifications and technical reports to show how to use appropriate tools (UML,POM, ASN.1, XML, web services etc.) consistently within a sector in order to maximise interoperability, and the mobility of subset solutions, and to provide cohesiveness, consistency and mobility in the skill base as it develops to enhance the successful expansion and exploitation of the ITS sector.

6.9 ITS Data registries and UML

There has recently been wide-spread interest in the development of data registries for ITS in a similar manner to their use in other related fields for example geographic (spatial) information. A current example is the ITS data registry developed for the UK Highways Agency [X] to the standard ISO 14817 (14817 2004). In this approach the artefacts are captured in forms of UML model fragments that are suitable for harmonisation and synthesis to create more extensive models. ISO 14817 *Requirements for an ITS/TICS central Data Registry and ITS/TICS Data Dictionaries*, provides a consistent framework and set of rules for the operation of data registries and dictionaries in the ITS sector and is consistent with internationally standardized and accepted data registration techniques.

The draft International Standard ISO 25106 Procedures and format for ITS Glossaries, at a simpler but consistent level shows how these techniques can be used to operate an on-line data registry to maximise common understanding and use of ITS terms in the sector.

A data registry supports the development of the elements previously defined in formal standards documents by incremental definition and publication. This process is intended to provide faster and more responsive progress to consensus on standards applicable to the intended domain.

6.10 User acceptance of the logical architecture (example)

NOTE Much of the content of this clause is sourced from the results of a national workshop on ITS Architecture in Canberra, Australia in March 2003 and the resultant 'Clark Report' [Clark 2004]. Other studies have found many similar results.

Investigations into the user acceptance of a logical architecture tend to conclude that:

a) Technical UML and use case descriptions need to be supported by descriptions readable by non-technical people.

b) A traceable structure should be developed across related documents allowing for the refinement of detail or treatment of perspectives suitable to the intended audience.

c) Document size and complexity should be reduced.

d) A choice of navigable or printable documents should be provided.

e) Should use language that is less technical and less verbose.

f) UML should be contained at a technical level with sufficient linkages and amplifications to allow the interested reader to delve more deeply where required.

g) An inventory of ITS Services should be built that could be addressed by an ITS architecture.

h) A formal set of requirements for an ITS logical architecture (and other documents) should be identified and used to control and guide further development.

i) Cross-linking and cross-references should be incorporated to allow easy access to dependent and related information.

j) A target audience should be identified for each document.

k) Language and content of documents should be tailored to the identified target audience.

l) Suitable tools need to be identified to facilitate the development of logical architecture complete with any required document automation features prior to any significant investment in content development.

m) A set of defined success factors should be identified, documented and measured.

n) Formal support from both industry and government is required.

o) A viable business case needs to be identified, and explained in layman's terms to obtain support for further development.

p) An appropriately documented and agreed process needs to be established to facilitate further development.

q) Avenues should be explored to obtain suitable commitment from appropriate participants in the process.

r) The needs of the various stakeholder groups need to be determined as a matter of urgency as to what it is that an ITS logical architecture should provide to them.

s) Those using this tool must always be aware that those not using the tool will find it difficult to follow and accept their results unless they relate to their client in terms that the client will understand rather than in the jargon of UML.

7 Implications (of user acceptance) for the use of UML

7.1 General comments

Returning then to the use of UML in standards, it should be noted that the subject of ITS architecture is itself under some scrutiny and thus adding further complication by use of UML when this is inappropriate may be 'adding fuel to the fire'.

Having said all of that, there are significant benefits in adopting a standardised approach that is unambiguous, comprehensible, definitive and expandable or flexible to accept inevitable change, and UML provides one useful and internationally accepted means to achieve this in the context of systems architecture.

7.2 Adoption of profiles

As with all standards it is always possible to select a subset of normative features and requirements as a 'profile' and to specify this profile for use in stated circumstances. This can be done for ITS standards. At the very least this would reduce the scope creep that would arise from always using the latest version of UML, and it would reduce the likelihood of backward incompatibility whereby later models cannot be interfaced to earlier models because some of the later features have no counterpart in the earlier model.

7.3 Modelling tools

There are several modelling tools for UML and there are also useful comparative evaluations for benefits, features and costs. What is more important is the use of a common format for model artefacts so that model parts can be exported from one tool for further development in another tool. The XML based standard XMI is not yet perfect but is improving.

7.4 Sufficiency of UML

There seems to be ample evidence that UML cannot cover all requirements for definitive modelling of complex, networked, distributed ICT systems.

UML provides perhaps the most comprehensive and flexible tool available. In some circumstances it can be used down to or near computer coding level. Its approach to supporting multiple 'views' within the same model is attractive. However those using this tool must always be aware that those not using the tool will find it difficult to follow and accept their results unless they relate to their client in terms that the client will understand rather than in the jargon of UML.

It must also be recognised that, capable though it is, and recognising that its capabilities are expanding, UML will not be appropriate in every and all situations. Therefore there must always be consideration of other forms of system modelling and its interpretation from the perspective of UML, and it is inappropriate to make claims that UML can cope with any possible requirement for architectural system design and implementation.

Bibliography

1471 2000 IEEE Std 1471-2000. IEEE Recommended Practice for Architectural Description. IEEE. 2000

14817 2004 ISO 14817-2002. *Transport information and control systems — Requirements for an ITS/TICS central Data Registry and ITS/TICS Data Dictionaries*. Standards Australia. 30 June 2004

ALC 2002 An eBusiness Interoperability Framework (Draft) Australian Logistics Council. 23 Dec 2002

AusLink 2004 AusLink White Paper. Building our National Transport Future. Department of Transport & Regional Services, Australian Government. June 2004

Bass 2003 BASS, LEN, CLEMENTS, PAUL, KAZMAN, RICK. Software Engineering in Practice (2nd Ed). Addison Wesley 2003

Bossom 2004 BOSSOM, RICHARD. 'Two architectures. One goal.' Traffic Technology International. Apr/May 2004 pp 65-69

Clark 2004 CLARK, JOHN. Report on The ITS Logical Architecture Review Version 1.0 26 May 2004. Prepared for ITS Australia

Clements 2003 CLEMENTS, PAUL et al. Documenting Software Architectures. Views and Beyond. Addison Wesley. 2003

e-Transport 1999 e-transport. The National Strategy for Intelligent Transport Systems. Austroads 1999

FRAME The European Framework ITS Architecture. European Union www.frame-online.net

Hofmeister 2000 HOFMEISTER, CHRISTINE, NORD, ROBERT & SONI, DILIP. Applied Software Architecture. Addison Wesley. 2000

Holt 2001 HOLT, JON. UML for systems engineering. IEE. 2001

ITSJ 1999 System Architecture for ITS in Japan. ITS Japan, November 1999 www.its-jp.org/english/arch_e/index.htm

Kulak 2004 KULAK, DARYL & GUINEY, EAMONN. Use Cases. Requirements in Context. (2nd Ed) Addison Wesley. 2004

McQueen 1999 MCQUEEN, BOB & MCQUEEN, JUDY. Intelligent Transport Systems Architectures. Artech House. 1999

NITSA 2003 The National ITS Architecture Version 5.0. US Dept. of Transportation, 2003 www.iteris.com/itsarch/

NRA 1999 A National Reference Architecture for Intelligent Transport Systems (ITS). Final Report. PPK for ITS Australia. Report. May 1999

Putman 2001 PUTMAN, JANIS R. Architecting with RM-ODP. Prentice Hall. 2001

Rosenberg 1999 ROSENBERG, DOUG. & SCOTT, KENDALL. Use Case Driven Object Modeling with UML. Addison Wesley. 1999

Shaw 1996 SHAW, M. & GARLAN, D. Software Architecture. Perspectives of an Emerging Discipline. Prentice Hall, 1996

Smith 2004 SMITH, DR JOHN. Email to NIAWG and IT-023 dated 18 April 2004

Szyperski 1998 SZYPERSKI, CLEMENS. Component Software beyond Object-Oriented Programming. Addison Wesley. 1998

ICS 03.220.20; 35.240.60

Price based on 14 pages