Erik Herzog and Anders Törne. Information modelling for system specification representation and data exchange. In Frances Titsworth, editor, *Proceedings of the 8th IEEE International Conference and Workshop on the Engineering of Computer-based Systems*, pages 136 – 143. ©IEEE Computer Society, 2001.

Information Modelling for System Specification Representation and Data Exchange

Erik Herzog & Anders Törne*
RTSLAB, Department of Computer and Information Science, Linköpings Universitet, Sweden
{erica, andto}@ida.liu.se

Abstract

This paper presents the emerging STEP standard AP-233 with focus on the non-functional requirements that have guided the development process. The purpose of the paper is to present and motivate the modelling assumptions and approach selected for the AP-233 information model, and to present how the EXPRESS information modelling language have been used. Although the paper is focused on AP-233 and the constraints imposed by the STEP framework it is believed the structures and requirements presented are general and applicable to other systems engineering information-modelling projects.

1. Introduction

Over the last decades computer based specification and analysis tools have been introduced in the systems engineering process to facilitate the development of increasingly complex systems. This trend has led to new problems. Computer based tools typically cover specific aspects of the systems engineering process. Multiple tools may be required to specify all aspects of a system. Moving information between tools, in cases where there is an overlap in information coverage, is difficult due to the absence of accepted tool independent data formats. The resulting situation has been characterised as one of 'islands of automation' [4].

Furthermore, in co-operation projects it is often the case that partner organisations use different tools for accomplishing the same task. Automated information exchange is also in this case obstructed by the lack of tool support for data exchange. Information sent for further refinement or analysis in tools within or outside the originating organisation often have to be re-entered manually into the receiving organisation's tool-set, a tedious and error prone process. The lack of accepted information exchange mechanisms also complicate upgrades from legacy to modern tools. Unless tool interfaces are developed information entered in the legacy tool can not be made accessible to the modern tool.

The AP-233 activity within the STEP framework (STandard for the Exchange of Product model data — ISO 10303) is an attempt to reduce the cost for implementing tool data exchange capabilities through the definition of a standardised information model for the systems engineering domain. By combining the information model with other STEP elements it is possible to automate significant parts of the interface development process, thus reducing effort and cost for enabling tool data exchange capabilities. There have been a number of papers describing the scope and content of the AP-233 data model, e.g., [6] and [7], as well as papers describing progress in the standardisation process, e.g., [10], [12] and [11]. But no attempt has been made to motivate the requirements behind the selected information modelling style and architecture.

This paper is an attempt to collect all non-functional requirements guiding the development of AP-233 in one document and illustrate how they influence the architecture of the information model. The requirements and guidelines identified have guided the development of the information model collectively. It must be underlined that information modelling is not algorithmic. There are no universal criteria for distinguishing right from wrong in an information model. The decision on how individual concepts shall be represented have been taken after analysis of domain requirements against modelling guidelines.

The rest of this paper is outlined as follows. Sections 2 provides background information on the STEP framework and section 3 presents the scope and main elements of AP-233. Three non-functional aspects of the AP-233 information model are discussed in sections 4 to 6 in this paper. Section 4 presents our understanding of domain requirements for a systems engineering data exchange information model and discusses the consequences of the requirements. In sections 5 the use of the information modelling language EXPRESS

* Also with Carlstedt Research and Technology AB, Linköping, Sweden

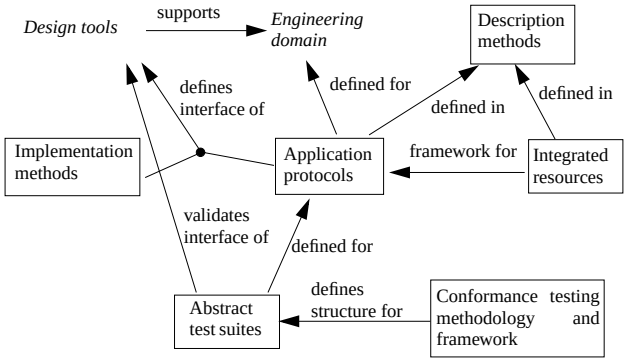
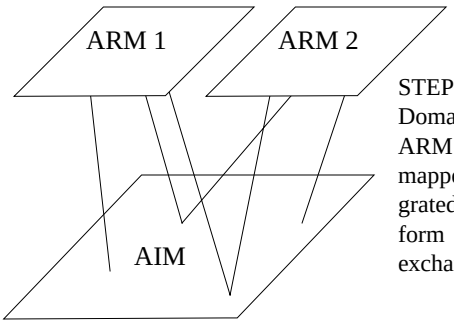


Figure 1. STEP classes and their relationships



STEP in theory: Domain specific ARM models are mapped onto integrated resources to form a unified data exchange format

Figure 2. STEP information model relationships

is illustrated followed by an outline of the selected for AP-233 approach to accomplishing interoperability and harmonisation with existing STEP parts in section 6. This is followed by a discussion on the factors which have shaped AP-233 in section 7 and a summary and outline of future work in section 8.

2. Framework

This section outlines the STEP standard framework and the benefits in using STEP components for information modelling and for achieving data exchange capabilities between tools. The overview is necessarily brief — in-depth information on STEP and its components are available in [17] and [13].

The objective of STEP is to provide the framework for the unambiguous representation and exchange of computer-interpretable product data throughout the life of a product. The framework is independent of any particular computer system. STEP is a modular standard. There is large number of parts in different levels of standardisation maturity. A part belong to one of the six classes presented in Figure 1. A brief description of each class is presented below.

An *Application protocol* (AP) is a definition of an information and process model for an engineering domain for the purpose of data exchange, e.g., AP-203 Configuration controlled design [1]. The structure of an application protocol information model is defined in terms of a *Description method*. The preferred description method used within STEP is the language EXPRESS [2]. More details in the EXPRESS language and how it is used in AP-233 is presented in Section 5.

Integrated resources provide a set of generic definition that is shared among application protocols to facilitate interoperability. The integrated resources are defined in EXPRESS. Data exchange is facilitated via *Implementation methods*. There are standard implementation methods defined for file based transfers and for specification of repository interfaces. *Conformance testing methodology and framework* define the validation process for STEP implementations and *Abstract test suites* provides a comprehensive set of test cases for an application protocol.

There is actually two information models defined for each application protocol. The Application Reference Model (ARM) defines the scope of the Application protocol using a terminology relevant to the engineering domain covered. This model is then mapped onto the set of integrated resources defined to create the Application Interpreted Model (AIM) for the Application protocol. The AIM defines the semantics of the data exchange format and, at least in theory, provides for Application protocol interoperability due to the reliance on common constructs. The relationship between the ARM and AIM is illustrated in figure Figure 2. The structure with two levels of information models has some impact on the information modelling style used within STEP. Implications of this approach for AP-233 is further outlined in Section 6.

The services offered by STEP allow for realisation of domain specific data exchange structures as well as domain specific repositories with uniform interfaces. As EXPRESS is a formal language, it is possible to transform EXPRESS models to representations in languages such as C or SQL, which in turn may be used to define the structure of repositories for model data.

Compared with the traditional method for data exchange, i.e., manually written interface software for reading/writing tool specific data formats, an information model driven approach have the following advantages:

- *2n* interfaces are required to integrate *n* tools if a common information model is used compared with *n(n-1)* interfaces if no standard model is employed.
- The existence of formally defined exchange formats allows for automation of large parts of the interface development process. Those parts not easily automated, i.e., the mapping from entities in the information model to the corresponding entities in a tool meta-model are supported by a formal definition — the EXPRESS

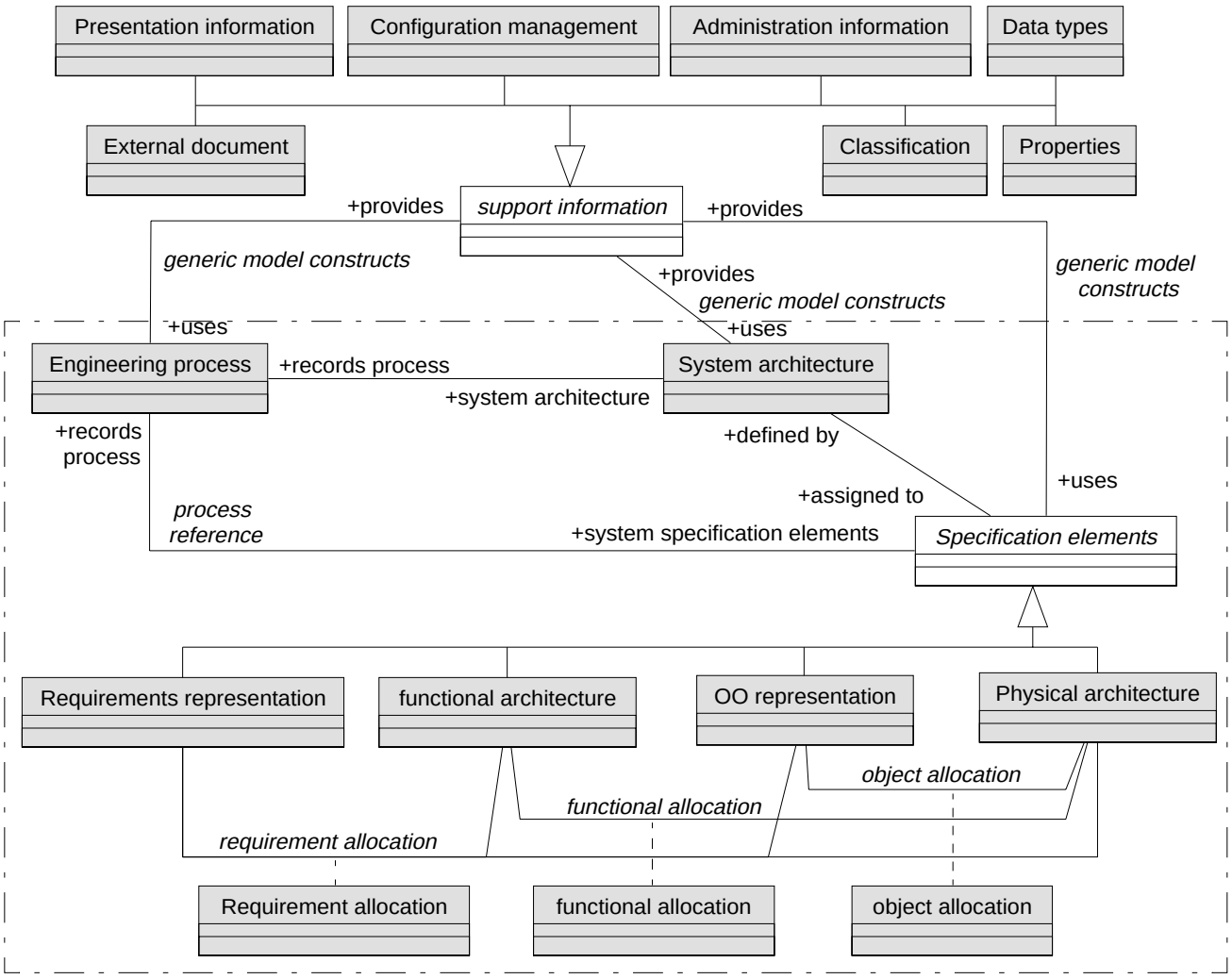


Figure 3. AP-233 conceptual view

model. This shall be compared with the manual process of mapping between two proprietary formats that has to be used if no common model is defined.

- An information model based approach allows domain experts to express their view on important domain information. In this sense the approach not only facilitates information exchange across tool boundaries — it also allow users to express requirements on future tools.

The selection of STEP as framework can be questioned in the light of the tremendous interest in the XML standard. However, the choice of framework shall not be over dramatised. An information model developed in express can easily be translated to XML. A standard mapping from EXPRESS to XML is defined in [18]. The end result, a standardised information model, regardless of language used, is the lasting value of the activity.

3. AP-233 scope and relationship to other STEP standards

Systems engineering is inherently heterogeneous. It spans over multiple engineering domains. The approach taken in AP-233 is to focus on areas that are technology independent — the representation of requirements in different formats and requirement traceability. Technology dependent aspects is covered in existing and future domain specific application protocols in the STEP framework. The consequence of this decision is discussed later in this section.

The scope of AP-233 is illustrated in Figure 3. using UML syntax. In the figure each UML class symbol represents a group of related entities. The classes enclosed in the dashed rectangle represent the core of AP-233 and those outside represent generic support concepts that in many cases are common with other STEP Application protocols. Opaque classes are generalisations inserted to

preserve clarity in the conceptual view. The main groups of the model are:

- System architecture, representing the building blocks for grouping all information valid for a system, a stakeholder or lifecycle view on a system or system interface. There is also support for representing the system breakdown structure.
- Specification elements, defining the basic building blocks for representing requirements, functional architecture and physical architecture. The functional architecture model contains detailed support for representing continuous and sampled system. Functional behaviour may be expressed in data flow, finite state machine or causal chain notations. There is also support for representing object-oriented specifications compliant with the UML 1.3 standard.
- Requirement, Functional and Object allocation, defining the mechanisms for maintaining traceability across different specification elements. There is support for tracing requirements to functions (including behaviour), objects and physical architecture elements and for allocating functional architecture or object oriented elements to physical architecture elements respectively. There is also support for representing verification activities and verification results.
- Engineering process, defining the building blocks for representing activities in the systems engineering process and associating specification information to the activities they originate/relate to.
- Support information, representing the building blocks for representing supplemental systems engineering information. This large group is an abstraction of groups for representing configuration management information, visual layout information, and mechanisms for referencing external documents, administrative information, data types and properties.

The relationships between the groups are outlined below. The elements of the *Engineering process* group can be used to capture the systems engineering process for a project and relate *Specification elements* and *System architecture* information to the process phase they were created/referenced in. *Specification elements* provide the structures to define requirements, functional and physical architecture of systems represented by elements of the *System architecture* group. The *Support information* group includes entities that provides information that is applicable to entities in the other groups.

The scope and structure of AP-233 is in accordance with those of published systems engineering information models, e.g., [16], [9] and [5].

The decision to exclude technology dependent information from AP-233 can only be justified if requirement traceability and allocation links can be established between AP-233 and existing production focused application protocols within STEP. The mechanisms for establishing application protocol interoperability is discussed further in section 6.

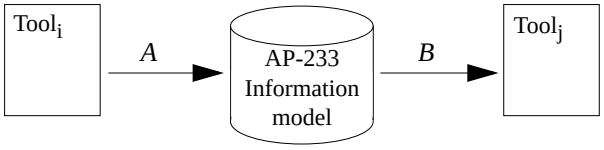


Figure 4. AP-233 usage

4. Information modelling requirements

This section presents seven basic non-functional requirements that have driven the development of AP-233 and the information modelling approach selected to meet these requirements. In this section the term ‘system specification’ is used for the collection of documents, models and other elements in scope of AP-233.

The purpose of AP-233 is to enable transient and permanent storage of system specifications as illustrated in Figure 4. Transient storage implies a point to point transfer from a tool A to a tool B and permanent storage implies an architecture where specifications are stored in a repository. From a data exchange perspective there are two scenarios for identifying non-functional requirement on AP-233. Data export from a tool which involves mapping tool specific data to the structures defined in AP-233 (corresponding to *A* in Figure 4.) and data import to a tool from AP-233 (Corresponding to *B* in Figure 4.).

The guiding principle behind the development of AP-233 has been to avoid encoding process, method or tool constraints in the information model.

Requirement: The information model shall be tool, method and process independent.

The rationale for this requirement is that AP-233 shall not dictate what a is a good systems engineering process, method or tool is. There shall neither be any process constraints enforced, nor shall there be any preference for tools or methods. In case multiple equivalent methods are identified as being within scope of AP-233 then the information model shall be fair in the support for all. A consequence of this approach is that no assumptions on what a ‘good’ specification structure is can be encoded into the information model. Classifications may only be encoded if there is unanimous acceptance within the systems engineering community. For instance, there exist no common view on requirement classification schemes [3]. In cases like this the

standard may propose a set of preferred classifications, but classifications must be open-ended and allow for additional used-defined elements. This is a strong indication that effective inter-organisation data exchange will require the definition of a common process. The definition of such a process is out of scope for a data exchange standard.

Just as the shall be no assumptions on what a good system specification structure encoded there shall be no constraints encoded that enforce a certain level of specification completeness before allowing a data exchange.

Requirement: No assumption on the level of completeness of a specification involved in a data exchange shall be encoded in the information model.

This requirement is justified by the fact that it is not meaningful to make an à priori prediction on the structure of a 'complete' specification. This requirement guides the structure of the information model such that dependency relationships between entities is avoided.

The systems engineering process typically generates a lot of alternate designs. Maintaining information on how individual specification elements evolve over time and the baselines individual versions are selected in is desirable.

Requirement: The information model shall be structured such that key specification elements can be used independently in the context of several system specifications.

The required capability can be used for answering queries like: "Which systems share particular functionality", "Which versions of a system share a specific requirement" or "How was a particular requirement handled in other systems". Similar requirements are expressed in [14].

System specifications shall be unambiguous and to the point, or there is substantial risk the specification is misinterpreted. If this is not detected there may be catastrophic consequence for a development project. Obviously, tool data exchange may introduce additional risk, as there is a risk that information is lost or modified in the transfer between a tool and AP-233. However, modifications to a specification imported from AP-233 format to a tool may always occur. This is inevitable when a specification is imported into a tool which can not handle some of the elements in the specification. This can be accepted as long as the data model provides the means for detecting modifications allowing them to be communicated to the operators in the import process.

Requirement: The information model shall be detailed enough to represent the semantics of a specification in scope of AP-233.

Requirement: The information model shall be detailed enough to allow for identification of specification fragments not correctly conveyed from a source to a destination tool.

Meeting these requirement is essential if the information model shall be an effective and reliable engineering tool a Consequently, the information model must be rich enough to support variants of important systems engineering concepts. At the same time it is desirable to keep a common core for similar concepts to keep the information model as small a possible. The objective is to avoid redundant representations of the same concept.

When exchanging specifications between tools it is not sufficient just to preserve the semantics of the specifications. For traceability and human interpretation purposes it is important to preserve the structure of the original specification after the transfer.

Requirement: The information model shall be constructed such that the structure of a specification mapped onto it shall be maintained.

Notation specific constructs are only accepted *iff* they are necessary to recreate the structure of the original specification. While it shall be possible to recreate an original structure from the information stored in the information model there are always cases where tool specific constraints force a change in specification structure. For instance, if a Mealy type finite state machine is exported onto the relevant structure in the information model, then it shall be possible to import it as a Mealy state machine. However, if the receiving tool is able to represent Moore finite state machines only then there shall be sufficient information preserved in the data exchange file to support the conversion.

Similarly the information model shall provide the means for representing the layout of a specification mapped onto the construct in the information model. Layout information shall not convey any semantics, only allow for recreation of presentation information retrieved from the original tool.

Requirement: The information model shall provide the capability to represent the layout of a specification.

Preservation of layout information is expected to be of high importance when complex specifications are transferred between organisations. If layout is not preserved in a data exchange it will be very difficult for engineers at different locations to discuss the content of a specification. At the same time no tool interface shall rely on the existence of layout information in a data exchange file.

5. EXPRESS MODELLING

In this section the basic information modelling constructs and the information modelling style selected for AP-233 is outlined. The selection of modelling language has a substantial impact on modelling style. AP-233 is implemented in the EXPRESS language [2]. EXPRESS supports the def-

inition of

- Entities: the basic object of the information model, i.e., a representation of an element within the scope of the information model.
- Inheritance relationships: The specialisation/ generalisation relationship between entities. Inheritance in EXPRESS can take one of the following forms: 'One of', an instantiation of a supertype is exactly one of the subtypes, 'And', an instantiation of a supertype is the union of all subtypes, or 'AndOr', an instantiation of a supertype is a variable subset of the union of all subtypes.
- Basic types: an elementary type that can not be further subdivided, e.g., an integer or a string.
- Properties: entities have properties. A property is a special aspect of an entity. From an information modelling view point a property may be expressed using combinations of the constructs below.
- Attributes: representing an aspect of an entity. Within AP-233 the term attribute is used to refer to aspects represented using basic types.
- Relationships: Defining associations between two constructs in an information model. Within AP-233 a relationship is always defined between entities.
- Cardinality constraints: a constraint on relationships and attributes defining the number of instances of one construct that can be associated by another construct. Constraints may be closed or open-ended.
- Textual constraints: EXPRESS supports the definition of formal constraints on entities, relationships, attributes and other modelling constructs. The expressive power is comparable to the combination of UML and OCL as used in [15].

The use of EXPRESS in the AP-233 information model is influenced by information modelling literature, e.g., [20] and [19], and by the use of EXPRESS in existing STEP application protocols, e.g., AP-214. The style is inherently influenced by the architecture of the STEP standard, i.e., the existence of two interrelated information models in the application protocol.

Information model entities are identified from concepts relevant to the systems engineering domain. There are three causes for defining entities:

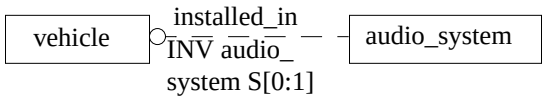
- The entity represents an established concept in the systems engineering domain or is used to further define the semantics of a concept in the systems engineering domain.
- The entity represents an abstraction (supertype) of other entities in the information model. The criteria for abstraction is that the sub-types are sharing a common attributes or relationships. All use of abstraction (inheritance in EXPRESS) in AP-233 specified using the one-of constraint to simplify implementation. This means that a supertype at any time can be used to represent the characteristics of exactly one of its sub-types. Furthermore, the use of abstraction in the information model is consciously minimised to avoid complicated expressions in the ARM to AIM mapping process.
- The entity represents a relationship between two or more other entities that in turn represent systems engineering concepts. The use of entities to capture relationship allows for recording attributes on the relationship. This approach is discussed more in detail later in this section.

EXPRESS offer two mechanisms for defining the semantics and ensuring integrity of an information model. As in any other information modelling language semantics may be defined explicitly be using specific entities with for each concept supported. The second approach is to define fewer entities and use formal rules to define valid attribute and relationship value combinations. In AP-233 the preferred modelling approach is to explicitly define entities for each concept supported and to minimise the use of rules for defining the semantics of the model. This approach result in more entities defined, but improves model transparency and readability. :

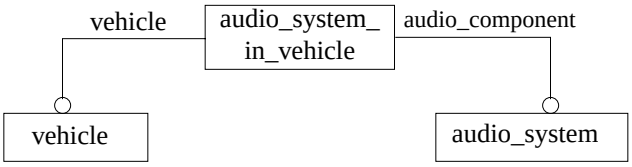
Table 1Entity relationship analysis

	INV [1:1]	**INV [0:1]**	**INV [1:?]**	**INV [0:?]**
[1:1]	Combine entities	[1:1] INV[0:1]	[1:1] INV[1:?]	[1:1] INV[0:?]
[0:1]	INV [0:1] [1:1]	Joint entity INV[0:1] INV[0:1]	INV [0:?] [1:?]	Joint entity INV[0:1] INV[0:?]
[1:?]	INV [1:?] [1:1]	INV[0:1] [1:?]	Joint entity INV[1:?] INV[1:?]	Joint entity INV[1:?] INV[0:?]
[0:?]	INV [0:?] [1:1]	Joint entity INV[0:?] INV[0:1]	Joint entity INV[0:?] INV[1:?]	Joint entity INV[0:?] INV[0:?]

Relationships between entities in the information model are identified using the guidelines illustrated in Table 1. The table is based on identified cardinality between an arbitrary pair of entities {*A*, *B*}. Each row in the table represents the cardinality between an object of type *A* to one of type *B*, and each column identifies the reverse relationship from *B* to *A*. The purpose of these guidelines is to improve the integrity and semantics of the information model by minimising entity interdependency, e.g., removing open ended sets,



a. Initial modelling style



b. Preferred modelling style

Figure 5. Information modelling style

optional attributes from the information model. The use of the table is illustrated by the following example.

Assume the following statement: *"A vehicle may be equipped with an audio system and a particular audio system may be installed in one vehicle only"*. Two types of artefacts are identified in the statement, vehicles and audio systems. The initial approach could be to model the relationship between car and audio system using an optional relationship as in Figure 5.a. The recommendation from table 1, for the case where the relationship is optional for both entities involved (row 2, column 2) is to use a joint entity as illustrated in Figure 5.b. There are several justifications for this recommendation. The artefacts in this example have life cycles that are independent of each other. A vehicle may not have an audio system installed and an audio system may be in storage for long periods of its operational life. The recommended approach also allow for capturing additional information on the relationship, e.g., the dates when a particular vehicle was equipped with a particular audio system. This improves the quality of information encoded in the data model.

6. Application protocol harmonisation

This section outlines the approach selected to harmonise the AP-233 information model with other STEP application protocols. As stated in section 2 the goal of STEP is to provide for product data exchange for all life cycles of a product and the mechanism is the two layered information model architecture defined in section 2.

The objective for interoperability implies that AP-233 must conform to the established modelling style within STEP in order to make the mapping to the AIM information model feasible. This is constraining factor for the structure the information model. The following paths to harmonisation has been followed:

- For each general concept identified as being within scope in AP-233 a comparison with existing application protocols has been performed.
- If concepts common to other application protocols are identified then this concept is copied in its entirety. For instance, AP-233 incorporates the same structure for representing system properties as AP-214.
- Overlap with existing application protocols is identified and hooks are provides. Redundant structures are avoided.

The traditional STEP approach with harmonisation in the AIM models has received a lot of criticism for being to elaborate and expensive to implement. A new architecture has been developed that requires harmonisation on ARM level [8]. The basic principle in this approach is to identify common basic information model fragments and implement them in reusable generic modules. The rationale for the new architecture is the expectation that the existence of core modules will decrease the time to develop an application protocol and improve interoperability. The most important activity in this new initiative is the development of a common schema for Product Data Management (PDM) information – The PDM schema. AP-233 will be harmonised with the new modules when they mature.

7. Discussion

Creating a systems engineering information model invariably lead to contradicting objectives. As with any information model it would be desirable with a model as small as possible. On the other hand, the information model shall provide a well defined semantics for the domain supported. In a heterogeneous domain like systems engineering this result in a substantial model. For the AP-233 information model we have given semantic rigour priority over desires to limit the size of the information model.

There is also a danger to be overspecific. If concepts which are not unanimously accepted are introduced in the information model they may constrain model usability for other parties. For this reason the use of closed classification criteria has been avoided, e.g., enumerate the set of possible classes using inheritance. The selected approach is to propose a classifications in the documentation instead of creating explicit entities. The result is believed to be an information model with a solid, well-defined, core and an extensible periphery.

8. Summary and future work

This paper has presented the information modelling approach selected for the development of AP-233. The main contribution is an outline of the basic modelling assumptions, how the information modelling language, EXPRESS, has been used in AP-233.

Maintaining specification semantics has been the high priority in information model construction. The current revision of the information model is extensive as it captures

the semantics as well as syntax of specifications. We believe this to be a crucial prerequisite for the successful standardisation and industrial acceptance of data exchange information models in general and for AP-233 in particular. At the time of writing the latest draft of AP-233 is being validated through the implementation of tool data exchange interfaces. The lessons learned from this exercise will be incorporated and the model will harmonised with the new modular structure in STEP.

Acknowledgments

The authors would like to thank all members of the SEDRES-2 and COHSY projects for their inspired work. The financial support from the European Commission and Swedish National Board for Industrial and Technical Development is also gratefully acknowledged.

References

[1] Anon. *Industrial automation systems and integration - Product data representation and exchange - Part 203: Application protocol: Configuration controlled design*. ISO 10303, 1st. edition, Dec. 1994.

[2] Anon. *Industrial automation systems and integration - Product data representation and exchange - Part 11: Description methods: The EXPRESS language reference manual*. ISO 10303, 1st. edition, Dec. 1994.

[3] Bahill, T. and Dean, F. Discovering system requirements. In Sage, A. and Rouse, W., editors, *Handbook of Systems Engineering and Management*, chapter 4, pages 175–220. John Wiley and Sons, 1999.

[4] Brown, A., Carney, D., Morris, E., Smith, D., and Zarrella, P. *Principles of CASE Tool Integration*. Oxford Press, 1994.

[5] Compatangelo, E. Towards an open framework for conceptual knowledge in ecbs domain and information modelling. In *Proceedings 1997 IEEE Conference and Workshop on Engineering of Computer Based Systems*, pages 211– 218. IEEE Press, 1997.

[6] Herzog, E. and Törne, A. Towards a standardised systems engineering information model. In Fairbairn, A., Jones, C., Kowlaski, C., and Robson, P., editors, *Proceeding sof the 9th annual international symposium of the international council of systems engineering*, volume 2, pages 909–916. INCOSE, INCOSE, 1999.

[7] Herzog, E. and Törne, A. Support for exchange of functional behaviour specifications in ap-233. In *Proceedings 7th IEEE International Conference and Workshop on the Engineering of Computer-Based Systems*, pages 351–358, 2000.

[8] Ishikawa, Y. and Vaughan, C. Sc4 industrial framework. Technical Report ISO TC184/SC4/WG10 N300, ISO TC184/SC4/WG10, 2000.

[9] Jackson, K. An information model for computer based systems. In Lawson, H. W., editor, *1994 Tutorial and Workshop on Systems Engineering of Computer-Based Systems*, pages 32–43. IEEE Computer Society Press, 1994.

[10] Johnson, J. The sedres project:producing a data exchange standard supporting integrated systems engineering. In *INCOSE*, 1998.

[11] Johnson, J., Herzog, E., Barbeau, S., and Giblin, M. The maturing systems engineering data exchange standard and your role. In *Proceedings of the 10th Annual International Symposium of the International Council on Systems Engineering*, pages 823–830, 2000.

[12] Johnson, J., Nilsson, B., Luise, F., Törne, A., Loeuillet, J.-L., Candy, L., Inderst, M., and Harris, D. The future systems engineering data exchange standard step ap-233: Sharing the result of the sedres project. In Fairbairn, A., Jones, C., Kowalski, C., and Robson, P., editors, *Proceedings of the 9th annual international symposium of the International Council on Systems Engineering*, volume 2, pages 923–931. INCOSE, 1999.

[13] Kemmerer, S. Step - the grand experience. Special publication 939, NIST, 1999.

[14] Mannion, M., Keepence, B., Kaindl, H., and Wheadon, J. Reusing single system requirements from application family requirements. In *Proceedings of the 1999 International Conference on Software Engineering*, pages 453–462. IEEE Press, 1999.

[15] Nordstrom, G., Sztepanovits, J., Karsai, G., and Ledeczi, A. Metamodeling- rapid design and evolutionof domain-specific modeling environments. In *Proceedings 1999 IEEE Conference and Workshop on Engineering of Computer-Based Systems*, pages 68–74. IEEE Press, 1999.

[16] Oliver, D., Kelliher, T., and Keegan, J. *Engineering Complex Systems with Models and Objects*. McGraw-Hill, 1st. edition, 1997.

[17] Owen, J. *STEP - An Introduction*. Information Geometers, 1 edition, 1993.

[18] Price, D., editor. *ISO/PDTS 10303-28 Product data representation and exchange: Implementation methods: XML representation of EXPRESS schemas and data*. ISO, 2000.

[19] Reingruber, M. and Gregory, W. *The Data Modeling Handbook - A Best Practice Approach to Building Quality Data Models*. John Wiley & sons, 1994.

[20] Schenk, D. and Wilson, P. *Information Modeling: The EXPRESS Way*. 1994.