

*XML schema, UML,
condition monitoring*

Richard LEWIS¹
Clive ROBERTS²

A GENERIC METHODOLOGY FOR THE DEVELOPMENT OF RAILWAY CONDITION MONITORING XML SCHEMAS

This paper introduces the concept of using the Unified Modelling Language (UML) to model a domain for eXtensible Mark-up Language (XML) schemas, for use within a railway condition monitoring application.

OGÓLNA METODOLOGIA ROZWOJU STRUKTUR MONITOROWANIA STANU KOLEI

Artykuł wprowadza koncepcję stosowania Jednolitego Języka Modelowania (UML) do modelowania domen dla schematów języka eXtensible Mark-up Language (XML), znajdującego zastosowanie w aplikacjach monitorowania uwarunkowań kolejowych.

1. INTRODUCTION

For many years remote condition monitoring systems have transmitted data as simple text files on a client server basis [1]. The server has a detailed knowledge of the structure of the text transmitted and this allows the end user to process the data.

Research has shown that only a small amount of transducer data is necessary to achieve a level of modelling that would enable fault detection and allow preliminary diagnosis to be performed when raw data is distributed as text based files [2]. Initial work using point machine data, where raw data was distributed as text, revealed compatibility issues [3]. This has led to centralised systems requiring a full version of the source condition monitoring software to process and display the data. A requirement therefore exists for systems to use common software to process data limits and sharing of data between disparate systems.

In the UK, work has already commenced in this area in the form of Network Rail's Engineers Workbench (EWB), developed by Atkins Rail who are committed to developing a cross platform distributed maintenance system.

¹ Electronic, Electrical and Computer Engineering, School of Engineering, The University of Birmingham Edgbaston, Birmingham B15 2TT, UK, lewisr@eee-fs7.bham.ac.uk

² Electronic, Electrical and Computer Engineering, School of Engineering, The University of Birmingham Edgbaston, Birmingham B15 2TT, UK, C.Roberts.20@bham.ac.uk

While the concept of XML instance documents and XML schemas go some way towards meeting this requirement, some problems were highlighted by the EWB.

The flexibility of XML schemas to model data presents a problem of constraining parameters and a specific method is required to ensure consistency throughout the model. A schema can quickly grow to an unmanageable size so a modelling method would help to manage this. Multiple schemas may help to manage the problems of scale, but meta-models are required to then manage the hierarchy of multiple schemas used by numerous users. The user's ability to produce valid XML documents can be hindered by a lack of understanding of the schema provided.

In a previous investigation, the application of XML to represent railway asset condition monitoring data was considered [4]. In this paper a generic methodology for the development of XML schemas for railway condition monitoring data is described. Particular attention has been paid to a method of extending conceptual UML models to constrain the data elements represented as XML schema profiles in the model. These profiles describe the combination of administrative data with raw asset data and the constraint of the parameters within them. A logical level diagram is presented that can be used as a basis for communicating the meaning of the schema to developers and users. Also presented is a method for defining multiple schemas based on a meta-model for multiple schema version control.

1.1. DOCUMENT MODELLING

Hyper Text Mark-up Language (HTML) is a language that enables the transfer of web pages, inclusive of page display descriptor, over the World Wide Web. HTML however, is limited to a fixed tag set, for displaying the information.

XML has, in recent years, emerged as an important development in the transfer of heterogeneous data on the World Wide Web. XML is a mark up language for documents containing structured information. Structured information consists of both the content and description of the role that the data plays [5]. A document model describes documents in XML. There are two ways of modelling a document:

- 1) Document Type Definitions (DTDs);
- 2) XML Schemas.

1.1.1. DOCUMENT TYPE DEFINITIONS

DTDs have been used extensively in web data management and database applications. DTDs describe a document's structure with declarative rules and provide information about the content of documents by describing what elements are, and are not, valid.

However, DTDs are limiting for XML since they do not provide information on the cardinality of elements and have a different syntax to the XML document they describe. The conversion of DTDs and other schemas into XML schemas enables the integration and querying of data [6, 7].

1.1.2. XML SCHEMA

An XML schema describes a document’s structure by using element templates. The XML schema is formed in the same syntax as the XML document it is describing, which is desirable for ease of understanding. Schemas also allow for elements to be declared as named data types and enable the constraint of attribute values within elements.

The W3C consortium has recommended a Schema Definition Language that describes and constrains the content of XML documents [8]. Industry standard XML Schemas enhance software interoperability by creating flexible file formats. There are a number of other XML schema languages available (Schematron, RELAX NG), though, it is considered that the W3C’s will become the industry standard. An example of the use of the W3C XML Schema Language is given in this paper.

1.2. GENERIC METHODOLOGY

The generic process for development of a schema is displayed in Fig.1. The initial step is to apply a Use Case Analysis to the remote condition monitoring system. A Use Case Analysis, forming part of the Unified Modelling Language (UML), models the use of a system, from the perspective of external systems and users, to define the requirements of that system. The data elements for the domain defined in this process are represented within the schema. If some, or all, of the assets to be monitored are new or unknown, then the process may include a Failure Mode Effect and Criticality Analysis (FMECA). This activity is based on Network Rail standards for maintenance and user requirements for Remote Condition Monitoring (RCM).

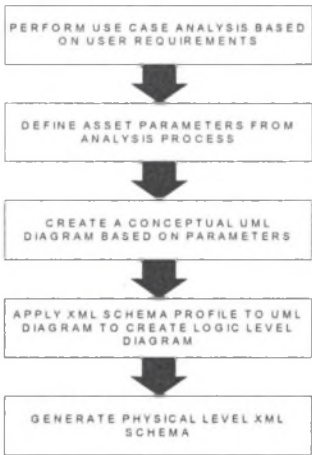


Fig.1. Generic XML Schema development process

Within the development process is a step to develop a conceptual level UML model from an analysis of the domain. The subsequent steps outline the process of analysing the

conceptual level diagram in order to create a logical level diagram, from which a schema can be created.

A method of mapping XML schemas from UML using a database design approach is presented by Routeledge [9]. This method is intended for mapping structured relational data from an existing data system using successive steps. In this method, nesting of domain elements is established through application of a rule set in order to achieve minimum redundancy. This method does not lend itself to the application presented here because the data to be modelled is relatively unstructured and the nesting can, to some degree, be arbitrarily defined.

Previously, a method of schema mapping has been presented that defines schemas designed to support the interfaces of the functional hierarchical layers of software component architecture [10].

1.3. CONCEPTUAL LEVEL DIAGRAM

After the initial analysis is complete and a list of the data elements essential to maintenance has been created, a conceptual level diagram that contains all of those elements is defined. At this stage none of the relationships between elements are indicated, this information will be added in the generic process to follow. This allows focus to be maintained on the elements and their content, without the distraction of their external relationships.

The content of the UML classes can now be defined. This need not be comprehensive since the process can be repeated when the domain becomes more familiar. The diagram shown in Fig.2 is the result of this process. It can be seen that the class elements have prefixes, which indicate to which schema they belong. These prefixes can be added at any stage in the process, depending on knowledge of the domain and the application.

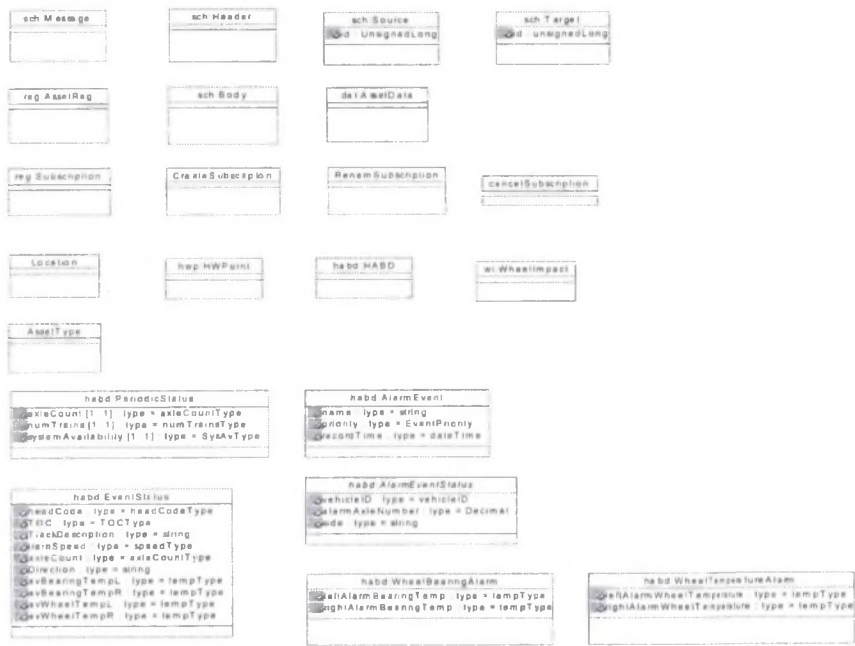


Fig.2. Conceptual Level UML diagram

Now that a set of classes relating to the domain exists, the details of the associations and cardinality can be added to the conceptual diagram. Note that each UML class maps to a complex type in XML unless explicitly defined as a user defined type. User defined types are definitions of UML extensions, that enable the representation of simple XML element types, which are difficult to model in UML, such as codes, IP addresses and parameters with specific constraints.

1.4. UML EXTENSIONS

The extensibility mechanisms of UML are called constraints, tagged values and stereotypes. Extensions are intended to provide additional model characteristics for particular application domains and programming environments.

A constraint is a semantic restriction represented as a text expression, which may be interpreted as a number of formal notations such as mathematical notation, pseudocode, program code, etc. If the language used to define the constraint is informal then its interpretation is also informal and must be done manually.

A tagged value is a pair of strings that contain some information about an element. This could be information about stereotyped model elements. A stereotype is a type of model element, defined in the model itself, which is used to tailor the modelling language to a particular application. A stereotype may have a list of constraints that apply to its usage, and

may use tagged values to store additional properties that are not supported by the base element.

The extension mechanisms of constraints, tagged values and stereotypes make it possible to tailor a UML profile for the domain under consideration. In this instance the additional modelling characteristics will be unique to the XML schema design.

Previously, it has been shown how stereotypes are assigned to one or more constructs that are modified by the profile extension [11].

In the model presented in this paper a stereotype called `anonymousType` is used to represent a restriction on a class that can only exist as an unnamed complex type. This stereotype allows only a sequence XML compositor (model group), which can only contain composition relationships with other elements. The associations with this class will then be numbered with sequenced integer position values. This is shown in Fig.3.

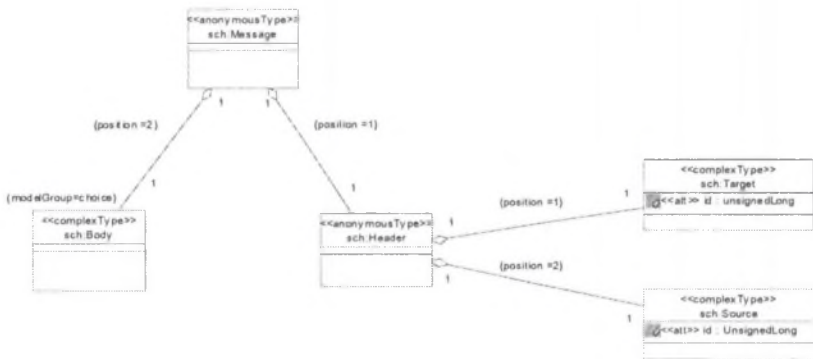


Fig.3. AnonymousType in conceptual model

The `complexType` stereotype uses the UpperCamelCase, (i.e. RegisterAssetType as opposed to register AssetType) element naming convention. The XML compositor type (model group) is indicated in brackets, {}, and the attribute and element mappings are defined as follows:

- Complex type element instantiations are shown as association end elements. These elements will:
 - Have position integers represented in brackets and will be an anonymous type;
 - Be false by definition;
 - Have name mapping of lowerCamelCase.

UML content elements will also be represented in lowerCamelCase name mapping. This is illustrated in Fig.4.

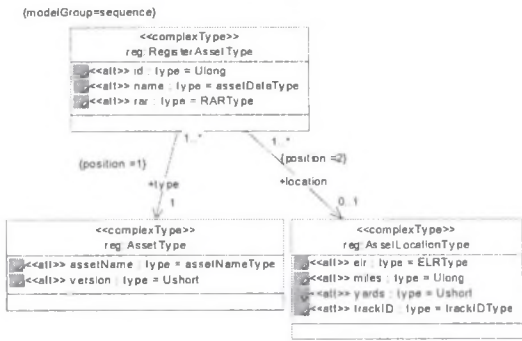


Fig.4. Complex type associations

Extending a UML element with the `<<att>>` stereotype enables the representation of XML attributes. These can be a user-defined type that restricts a standard XML Schema Document (XSD) type or standard XSD type. This is shown in Fig.5, where `assetName` is of user-defined type `AssetNameType`. A user-defined type inherits the properties of a simple standard XSD type, and then uses a stereotype `<<XSDfacet>>` to restrict that type. In Figure 5, the stereotype restricts a string to represent a list of enumerated types.

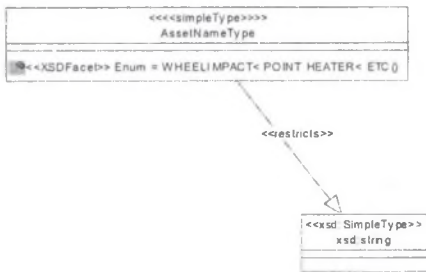


Fig.5. User defined type

Extending a UML element with the `<<elt>>` stereotype enables the representation of XML simple type elements. Again these can be user-defined types extending standard XML types or just standard XML types.

Another XML facet, that is non-standard in UML, is the uniqueness constraint. It may be a requirement for a schema to represent a parameter that is unique, such as a telephone number or, more appropriately, an asset identification number.

In previous work an example is presented to illustrate how a uniqueness constraint for a telephone number is represented in XML [12]. This is shown as follows

```
<xs:unique name="phoneNums">
  <xs:selector xpath="phone"/>
  <xs:field xpath="@addr:number"/>
</xs:unique>
```

Now if a given contact's element contains two phone elements with the same value for their number attribute, the schema processor will generate an error.

In UML a simple method for representing uniqueness constraints to be mapped to the XML schema is required. Previously a method for representing key fields for relational data has been considered [13]. Using the same principle to represent uniqueness constraints, an '@' symbol is placed in front of the elements or attribute to be constrained. This is illustrated in Fig.6.

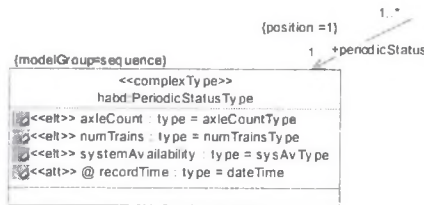


Fig.6. Uniqueness constraint representation

1.5. LOGICAL LEVEL DIAGRAM

Creating a logical level diagram from the original requirements, based on the conceptual diagram, involves the use of standard UML notations. These are briefly described as follows.

1.5.1. ASSOCIATION

The association relationship type is used to represent relationships between classes within the domain. The instantiation of the class is indicated by the role name given to the association.

1.5.2. INHERITANCE

This is based on concepts for creating object oriented complex type definitions [14]. This allows the subscription class to be abstract such that the class itself shall not be instantiated, only inherited by other classes. This means that subscribe instantiation can become a choice of create subscription, renew subscription or cancel subscription

In general, elements and attributes within the conceptual domain can map directly to elements and attributes in the logical level. However, the method described in this paper constrains this flexibility by using stereotypes to force the type definition at the conceptual level.

The diagram in Fig.7 shows the UML notations and conceptual elements described previously, combined into one logic level diagram for the domain. This diagram shows the representation of all the elements, parameters and composition information required to create a schema for the railway asset condition monitoring data discussed. This would enable the user of a schema to understand how the elements are related in the schema layout. Note that the user-defined types are not displayed in this diagram for clarity.

In this diagram it can be seen how information relating to both asset elements, and administrative processes, for remote systems are placed in the same schema document. The nesting of each data element can be observed from the relationships between them.

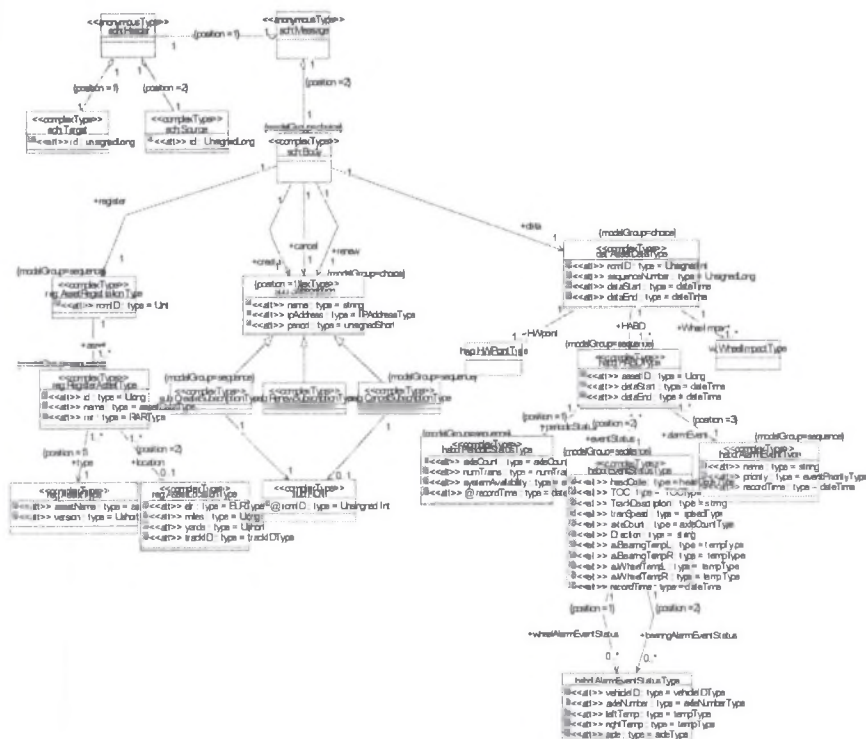


Fig.7. Logical level diagram

1.7. MULTIPLE SCHEMAS

In the process of schema development and use, a requirement may exist to issue a new version of a schema to represent an amendment to an element, or include an entirely new component schema to represent a new asset. This proved to be a significant problem to the EWB since numerous parties may use a version of a particular schema. Release of a new version to one party may lead to the re-issue of a new schema to all users.

In previous work, a method is presented that enables a top-level schema that contains the component schemas in the namespace [15]. Creating a new release of that namespace allows the extension of a set of component schema defined in one namespace. This creates change control since all documents valid for a particular release of a namespace will also be valid for the next release. Note that in this method, the year of issue is included in the namespace name to limit the number of releases, and a prefix is included indicating the origin of the schema.

A top-level schema, `ast_schema2003_r1.xsd`, contains three component schemas, some of which contain their own component schemas. Extending some of the complex elements of `ast_RegistrationSchema_v1` to create `ast_RegistrationSchema_v2` creates a requirement to re-release the top-level schema. Then `ast_RegistrationSchema_v2` will be included in the new release, but `ast_RegistrationSchema_v1` will not.

This is feasible since `ast_RegistrationSchema_v2` still contains all of the elements of `ast_RegistrationSchema_v1`, but uses XML inheritance mechanisms to extend the components requiring amendment. This is illustrated in Fig.8.

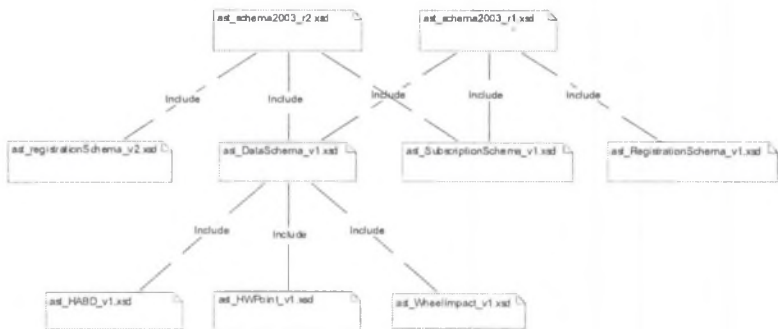


Fig.8. Extended schema view

1.8. CONCLUSION

In this paper a generic methodology is presented that aids in the development of models for XML schemas. Included in this is a brief description of document modelling.

The example model given is based on railway administrative and condition monitoring data and presents the use of the Unified Modelling Language (UML) to represent data elements and their relationships. A method of extending standard UML elements to represent XML profiles is illustrated.

A method for defining the domain elements in a conceptual diagram is presented and the process of developing a logical level model diagram is detailed. It is shown how the logical level diagram forms a basis for communication of the schemas meaning to users.

A concept for the use of multiple schemas is presented with an example based on a meta-model for multiple schema development from Network Rail Standards.

This work goes some way to addressing the problems experienced by EWB in creating XML schemas. It is shown that UML provides a solution to the problem of modelling XML schema profiles but also highlighted is the arbitrary nature of the representation of these elements. The work presented deals with the issue of setting a standard generic methodology for representation of railway condition monitoring data.

ACKNOWLEDGEMENT

The authors would like to thank Network Rail and Atkins Rail for their assistance and continued support during this project.

BIBLIOGRAPHY

- [1] DASSANAYAKE, H.P.B, ROBERTS, C., GOODMAN, C.J. (2000). An architecture for system-wide fault detection and isolation. Proceedings of the IMechE Part I Journal of Systems and Control in Engineering, vol. 215, no. 1, pp 37-46.
- [2] ROBERTS, C., DASSANAYAKE, H.P.B., LEHRASAB, N., GOODMAN, C.J. (2001). Distributed quantitative and qualitative fault diagnosis: railway junction case study. IFAC Control Engineering Practice, vol. 10, no. 4, pp 419-429.
- [3] ZHOU, F., et al. (2002). Remote condition monitoring and validation of railway points. IEE Journal of Computing and Control Engineering, pp 221-230.
- [4] LEWIS, R., ROBERTS, C. (2003). Using XML to support distributed maintenance management: a railway industry case study. 16th International Conference on COMADEM, pp199-208.
- [5] VAN der VLIST, E. (1998). What is XML? www.XML.com.
- [6] COLLINS, S., NAVATHE, S., MARK, L. (2002). XML schema mappings for heterogeneous database access. Information and Software Technology, vol. 44, pp 251-257.
- [7] MELLO, R., COSTANO, S., HEUSER, C. (2002). A method for the unification of XML schemata. Information and Software Technology, vol. 44, pp 241-249.
- [8] FALLSIDE, D. (2001). XML Schema Part 0: Primer, <http://www.w3.org/TR/xmlschema-0/>.
- [9] ROUTELEDGE, R., BIRD, L., GOODCHILD, A. (2001). UML and XML schema, 13th Australasian Database Conference.
- [10] THURSTON, M. G. (2001). An open standard for web-based condition-based maintenance systems, Proceedings of IEEE Systems Readiness Technology Conference, pp401-415.
- [11] CARLSON D. (2001). Modelling XML Vocabularies with UML: Part III. www.XML.com.
- [12] HAROLD, E., MEANS, W. (2002). XML in a nutshell. O'Reilly & Associates, Inc.
- [13] PROVOST, W. (2002). UML for W3C XML Schema Design. www.XML.com.
- [14] KIM, L. (2003). The official XML spy handbook. Wiley Publishing, Inc.
- [15] DORÉ, J., Railtrack Information Systems - XML standards and guidelines.

Reviewer: Prof. Andrzej Lewiński