SWAD-Europe Deliverable 6.2 The use of RDF in conjunction with other XML techniques and specifications.

Project name:
    Semantic Web Advanced Development for Europe (SWAD-Europe)
Project Number:
    IST-2001-34732
Workpackage name:
    6. XML and Semantic Web Integration Research Prototypes
Workpackage description:
    ☞http://www.w3.org/2001/sw/Europe/plan/workpackages/live/esw-wp-6.html
Deliverable title:
    6.2 The use of RDF in conjunction with other XML techniques and specifications.
Author:
    ☞Martin Pike, Stilo, UK.
Abstract:
    This document describes three use cases where RDF/OWL clearly has a role to play together with other XML syntaxes in encoding data for information capture, manipulation and dissemination.
Status:
    Completed report, 2004-06-06

Contents

The use of RDF in conjunction with other XML techniques and specifications.

Introduction

The ability of RDF and Semantic Web technologies to represent highly specific information of almost any kind in an open, machine-processable format opens up many opportunities for their use in the areas of information capture, manipulation and dissemination at a highly granular level. It also, by design, enables the meaningful and open association of formerly independent pieces of information which may be used to construct other information sources, such as individual documents or websites.

This paper outlines three different cases where RDF and Semantic Web technology can be used in conjunction with other XML data representations to create applications which would not necessarily be possible otherwise. Certainly one might assume that the use of proprietary data formats in these applications would have limited their acceptability and possibly destroyed their viability.

Use case 3 in this document is the use case upon which the WP6.2. code deliverable is based. However, the general infrastructure of the prototype for use case 3 would also be re-usable in cases 1 and 2, particularly the user interface and the XML interchange schema.

In WP5.2 research was undertaken into the extraction of semantic information from other XML structures. Although this is possible to a certain extent, in most cases assumptions have to be made as to why a construct was so built[1]. The information necessary to fully extract the semantics present in the structure is partly in the head of the author of the document and/or the author of the document schema or DTD. Therefore the availability of a semantic model, irrespective of the other XML encoded information, has value.

There is a general principle in the use of XML, including RDF - namely that the more well- defined and structured the information the more one may achieve with automated processes. However, the structure must contain some semantic information. For instance a document written with a standard word-processing system is said to be semi-structured. If the document is XML encoded with just the style attributes captured, little extra value can be gained from the XML representation. However, if the XML structure has a degree of semantic information then much more can be achieved, such as identifying and extracting particular fragments of information, or reordering document sections according to particular information attributes (e.g. a catalogue of parts - by part number).

The use of RDF takes this paradigm to the next level. The processing of RDF encoded data enables automated processing with a degree of sophistication greater than the automation that can be achieved with less semantically rich structuring.

The use cases stress the importance of the use of RDF data incorporated as information models as the **driver of applications** because of the high degree of definition and granularity such a model can offer.

References

[1]WP5.2 Extracting Semantics from XML Structure: S. Buswell
☞ In Search of the Semantics of Structure

Case 1. Enabling inter-application communication using content transformation.

Using semantics to automate the generation of data transformations; exploiting the difference between semantic meaning of data and its physical representation.

Overview - It is generally acknowledged that interchange of data between systems is made more understandable by the use of XML as a data format, despite some problems - such as bandwidth, that are cited against its use. However XML is only a syntax - vocabularies and grammars are designed to use XML encoded data for specific purposes. These are captured in DTDs and schemas. The problem lies in that, given a particular problem to solve, two people or groups are likely to create two differing solutions for it. The result is a plethora of both public and organisation schemas for capturing data and transferring it between applications and systems.

This is not a new problem. The same issues arose in Electronic Data Interchange in commerce (EDIFACT) for instance. But in that particular case, protocols were agreed on a point-to-point or an organisation-by-organisation basis. XML is seen as a way of avoiding this highly specific way of organising communications and enabling much more widespread, open communication between systems.

There have been a number of initiatives to avoid the point-to-point problem, involving whole industry sectors in the development of standards for data interchange. One particular example of this is UNIFACT ebXML. The problems with this type of solution is that they take a lot of organisation, a long time to formulate, result in a specification that is reached as a compromise between the parties involved, and are difficult to change; as are many standards developed in this way. Many organisations want the freedom to be able to move faster and adapt to change quicker than a standards organisation can manage. Standards may be used, but rarely will they be adhered to when this is the case.

One of the key principles of XML is that organisations, applications teams and individuals can use the XML Syntax and schema definition mechanim to capture and represent data in the way that they want it to. However, we then arrive back at the point-to-point solution for data transfer, but with a twist. The specification for the data protocol is in a standard, machine readable format - the DTD or schema. Also, the specification may be explicitly referenced in the data being transferred, as a <!DOCTYPE statement, or an xmlns attribute.

The issue lies in understanding what data represents semantically, not its physical representation. Then we can take advantage of its documented semantic representation to change it to a physical representation most suited for our system, or to send to a system where we know what representation is required. In short, the data will need to be transformed.

The problem - A global accountancy software organisation that has grown by acquisition, wants its many (50-60) accountancy packages to be able to communicate data between them. Each package performs standard accountancy tasks, but each has been tailored to the accountancy practises and tax laws of the location/country in which it is used. It is also maintained by developers dedicated to that particular package. Therefore, although the semantics of much of the data used by each package is similar, the format in which each can import and export data is different.

For example, the concept of *"Supplier Name"* might occur as *Vendor>Name* in a namespace *"English Purchase Order"* and also as *Fournisseur>Nom* in a namespace *"Bon de Commande Francais"*

Each package may import and export 10-12 particular data sets to perform its tasks. Therefore to ensure that any package can communicate its dataset to any other requires the creation (and maintenance) of $\sim 50^2$ *12 transformations. This is obviously not possible to accomplish manually. This is a manifestation of what is termed the n^2-n problem.

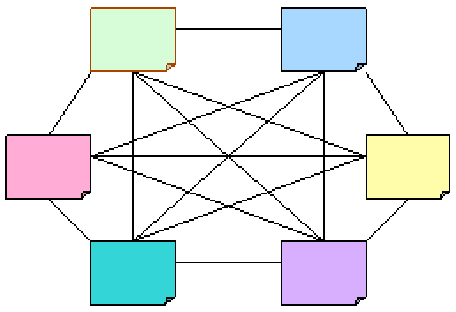

Fig 1: The n^2-n problem of transforming between data representations (Where n = 6. No. of connections (each-way) = 30)

The example of 'Supplier Name' given above may be seen as a relatively trivial case on its own, but the potential scale of problem, across many formats, increases the complexity of the problem.

In addition to the mapping data between different namespaces, other problems will exist:

- The layout of the data in the different formats will be different. For instance, the order in which the details of the supplier and customer appear may be different between purchase order formats.
- The breakdown of the data may be different. For instance a may be further divided into and , but may make no such division.
- The data itself may be represented differently. For instance, a date on a North American Purchase order may be represented as mm/dd/yy, whereas in Europe it would be dd/mm/yy. In another case, the customerID may be a unique numerical representation in one case and a company name in another.

The solution - It is obvious from the example above, that it is possible to describe the meaning of a piece of data independently to its physical representation.

The basis of a solution to this problem would be then to relate the physical representation of a piece of data to its semantic meaning, thereby enabling a connection between two pieces of data of different represenions to be established. It should be clear that it is possible to construct an 'ontology' that captures the domain of data used in accountancy and patterns used in its representation.

If this can be accomplished, then the problem of mapping data between many formats reduces from a exponential one to an (almost) linear one. ie. without the construction of an ontology, each data representation would have to be mapped to every other representation for transformation. With an ontology once a link has been made between the data representation and its semantic concept, then it is immediately linked with all other data representations with the same semantic meaning.

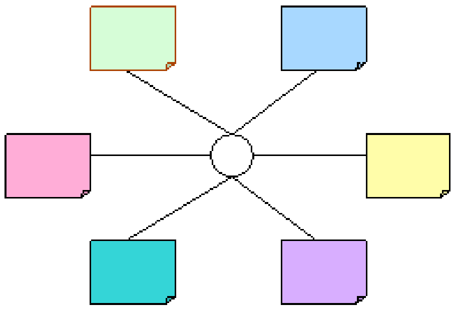

Fig 2: The n^2-n problem reduced to n by linking data representation to semantic meaning (Where No. of connections = 6, but interface-pair options still= 30)

However, although the number of links between data representations has been reduced, the transformation possibilities have not. Therefore a computational method for converting this 'knowledge' of data representations into instructions, to transform the data from one form into another, is required.

Additional rules will have to be captured within the ontology to handle the processes by which pure data transformations (such as the numerical customer ID/customer Name) will take place. These rules will have to be related to specific data transformations and therefore increase the complexity of the solution. They will in some cases, also have to relate to specific functionality. This cannot be automatically generated, but would have to be built manually. However, the automatic transformation mechanism should be able to include or invoke such functionality when it is required.

Given knowledge of the format of an input message and the format of the output message, information about the relationships between the data pieces, held in the ontology may be used to generate a set of rules that describe how each piece of data in the input can be placed in the output.

From those rules a transformation may be built. The transformations may be executed using any suitable technology. However, the nature of the problem does lend itself to implementation using an XQuery engine.

Conclusion - An application built using an RDF information model to capture the semantics of independent pieces of information and the relationships between them could be used as a transformation system to provide automatic conversion of information for transfer between systems. Such transformation need not be XML-to-XML but could be any format to XML and/or XML to any format.

Recently a discussion of this type of application has been aired, at the WWW2004 conference[1]. Tim Berners-Lee raised the idea of an 'RDF clipboard' where information from one application can be cut and pasted into another, with requisite transformation taking place according to the formats the applications can export and accept.

References

[1]WWW2004 Keynote: Tim Berners-Lee
☞http://www.w3.org/2004/Talks/0519-tbl-keynote/slide20-0.html

---

Case 2 - The development of Knowledge Models (ontologies) relating to interfaces for inter-application communication.

The use of ontologies to capture information about systems and the data that they need to import and export to automatically generate XML schema to specify their interfaces.

Overview - XML is being chosen as the 'lingua franca' for transporting data between and among systems of all types. Web Services are seen as a flexible, effective way of interacting with systems. Service-Oriented Architectures (SOAs), based upon Web Service architectures, are being heavily promoted as the next step in the way in which large organisational systems will interact with each other and are seen as the next step in the middleware evolution. They are heavily based on XML formats such as WSDL, SOAP and others, and also require well-defined interfaces in the form of XML schemas.

As those who have tried to do so can testify, designing schemas and DTDs for medium/large applications is a significant process. It has been compared to, intellectually, the design of a large database schema. Therefore it is a challenging and specialised task.

In some organisations and institutions the problems of developing such interfaces is becoming acute, such is the number and scope of the interface definitions (schemas) that are required.

The process of developing such interfaces involves:

- the creation of a working group of specialists that need to scope the problem area.
- develop the specification of what the interface is to achieve
- develop a schema that captures the specification.

This is a long and tedious process and may focus on only a small part of the problem, for political and timing reasons. This results in fragmented approach to interface development with little or no thought to re-use of the work carried out and consideration of the system overall.
In the past, organisations have attempted to address this problem by developing 'global data models'. These have suffered similar problems to those stated in the overview in Case 1, where the process is long, slow, inflexible, the result difficult to maintain and difficult to adhere to.

Some may say that the ability of many systems to be able to publish their interfaces automatically, in WSDL and schemas solves the problem of interoperability. It doesn't; it makes access to a particular service more open, it does not aid in the seamless transfer of information between systems with different interfaces.
This type of problem is very difficult to solve, but a step towards its resolution may lie in the level at which such a model is captured and the way in which XML technologies and Knowledge Acquisition techniques may be used to streamline and automate the development of such models and use them.

The problem - A large institution, with many diverse applications is trying to improve the flow of data between systems; to reduce paperwork and improve the quality of data in the systems.

The institution consists of many departments, each with limited communication with the others, by both personnel and machines. Knowledge of each of the systems is limited to those in a particular department. Knowledge of particular applications is often limited to those directly involved with them.
Many of the systems are quite old and have 'stovepipe' applications (where there has been little or no communication with any other system). Due to this there are a number of duplicate applications on these systems; for instance addressing applications.

The goal is to achieve better integration between systems, to deliver better service to the customers. This will be seen in a quicker response to queries, as data can be accessed and correlated more rapidly. Also the data delivered to customers should be more accurate, with duplicate systems being phased out, or at least coordinated. Other advantages to be gained are reduced costs, with a reduction in the number of systems and staff to maintain them and the ability to create new applications more readily and cheaply, as existing data should be more freely accessible and the type and quality of data available is documented.

A decision to use XML for data exchange has been made and a project initiated to define the data that is required to be passed between the different systems and the format in which they are to be passed. A number of schemas have been defined for some of the better known interfaces, but progress has been slow and the resulting schemas have not been well understood - what they represent in terms of the data used within the institution and how they are to be used, as the process of their creation was not well documented. Also although there is some re-use of components, the schemas are difficult to relate to each other.

The solution - The institution needs to speed up the process of schema definition, make the process more transparent and to relate elements of the schemas to each other, sharing the knowledge of significant parts, such that others may relate to them and potentially re-use them elsewhere. The ability to build a domain vocabulary and document relations between the items in the vocabulary with respect to the context of use is essential.

A project involves the resources of many groups of cooperative agents (human or otherwise). Each group makes its own contributions, and the overall success of the project depends on the degree of integration between those different groups throughout the development process. A key to effective integration is the accessibility of information, via rich ontologies, that characterise the domains addressed by each group.

Therefore, assisting each group to create a knowledge model that relates to the applications and data in their context, allowing that knowledge to be shared and extended by other groups is necessary. In this way a model of all cooperating systems and applications may be built up and documented. Working groups set up to develop and use the model will be able to gain a common understanding of the terminology used by other groups and the views of related issues, through access to collateral information that may be incorporated into the model. Note, that at this stage there is no development of schemas. The task is solely to develop a model of what is known about the environment. Information from schemas already constructed may be used as input to the model.

A prototype application for relating elements of XML to an ontology is being built as part of this work package[1]

The goal is not only to use the ontology for gathering, sharing and documenting information about the systems, but to enable the generation of the schemas needed to exchange information.

In creating the ontologies many abstract concepts may have been expressed and relations between them varied. However, to be successful at generating useful schemas from the lower level information in the ontology, a methodology needs to be instituted to capture that information at the lower level, such that schema generation tools will be able to relate to the structure.

The issue comes when generating schema. The model may capture the set of data items that relate to specific areas of functionality and that will be needed as input or output to the application. However there will be many possible combinations of schema structure that can be generated from the model, that will suffice as an interface. A task in building such an information modelling system is to devise a rule set that allows the consistent translation of modelling data into schemas, such that a pattern between the schemas is established and thus render the interfaces more comprehensible when viewed as a set.

References - [1]WP6.1 Extraction of semantic data from colloquial XML: I. Johnson and B. Matthews, CCLRC, UK
(6.1 xml_sw_prototype_schema)

Case 3 - Building knowledge objects from disparate, related resources

Using semantic web technologies to link resources, of disparate types in disparate locations, together to deliver to users coherent sets of information that are related and pertinent. In conjunction to establish a mechanism by which single-sourcing of information may be achieved

Overview - In many organisations electronic (and paper) information about processes, events, activities and items are held in various places. In many cases the information is very different in nature, stored in different formats and in different types of repositories. One of the issues within organisations today is access to all these different types of information concerning a subject and being able to aggregate it and present it to an inquirer. Connected with this is the requirement to be able to update information consistently. In many situations information has been taken from a source and copied somewhere else - put into a more accessible local folder or database, cut and pasted from one document into another for publication, etc. The result is that one piece of information resides in several locations. Should that information change it may be necessary to update that information in a number of locations.
The solution to this problem is known as single-sourcing; a system devised to enable information to be stored at a granular level, each piece of information being held as a unique resource, ie. held in one place and one place only. Each individual fragment of information may then be accessed and combined with other pieces of information, in a framework for publication.

The system that needs to be put in place needs to be both technical and cultural. A sufficiently capable system must be put in place to enable single-sourcing to be achieved comfortably. Staff and users of the system, must also understand the reasons for single-sourcing, accept it and be prepared to use the system. The cultural aspect often rears its head when ownership of information between departments or individuals becomes an issue. Ownership of information implies maintenance of that information. In the cut-and-paste culture when someone wants a piece of information updated, they do it and any consequences of that alteration are dealt with on an ad hoc basis. In a single sourcing culture a more rigorous mechanism for maintenance is required.

To enable this to happen in a large system each piece of information needs to be identified in such a way that it may be accessed in the correct context by publishing and editing applications. This requires that the information is self-describing and can be discovered using standard content search technology, or is associated with suitable metadata to allow its discovery. This use case will concentrate on how semantic web technology can be used to build such metadata frameworks and also provide templates for the creation of information objects from these multiple sources.

The problem - In large engineering organisations the manufacture and maintenance of products is a highly skilled activity, involving maybe thousands of staff, and copious amounts of information. This information may reside in Content Management Systems (CMS), Product Lifecycle Management Systems (PLM), other enterprise systems, ad hoc departmental databases, file systems, etc. Much information may also lie in the heads of experts who work in the organisation.

The information required to, say, build an aircraft or a motor vehicle is highly detailed, very specific and must be up-to-date and accurate. The challenge of maintaining these levels of information and streamlining and automating its production to optimise total costs of production is a constant one.

In these domains much information about particular parts, operations and processes may be held in different departments, using different technologies, for different purposes. For instance, a particular aircraft component has design documentation about its composition, its shape, its relation with other components, etc. Documentation may consist of engineers design notes, related papers, CAD optimisation code, diagrams, structural analysis data, etc. In a completely different department in the organisation, documentation may be produced for use in the field, concerning how to replace the component, torques to be applied to the bolts joining the component to others, warnings about space tolerances, etc. In yet another department reports may be received from customers of the aircraft about defects and other issues concerning maintenance and operation of the aircraft.

To coordinate and link related pieces of information across these disparate resources is a goal that a number of organisations would like to achieve. For instance, when a customer report concerning the difficulties involved in replacing a certain part is received it would be advantageous to be able to view that report in conjunction with its maintenance procedures and, potentially, elements of design documentation. If this could be achieved correlations between maintenance procedures and design criteria may be evaluated with a view to design alterations being made to improve the maintainability or life expectancy of the component.

The issue is not one of being able to make this happen; it can be done now, albeit with a considerable investment of effort in manual procedures. The issue lies in being able to build infrastructure and applications that can make the process more automated and cost-effective. If this can be achieved then existing processes may be improved, but it will also make it more viable to compare data and make decisions in more, perhaps smaller cases, where time and cost prohibited the attempt previously, but which will lead to greater cost savings in the production and maintenance of components.

The solution - To achieve the level of integration that is needed to bring together the different resources in an organisation of any size is obviously a very serious undertaking. Yet the savings that can be made in doing it are sufficiently large to warrant the investment. One of the issues to date has been the lack of open standards for metadata capture and information model construction that are necessary to provide the framework in which the systems integration may take place. The advent of RDF and other semantic web technologies has lifted this barrier.

Nevertheless RDF/OWL is only a syntax. There is still a significant problem in harnessing their capabilities and creating coherent, manageable models of the information objects, such that they may be accessed, combined and published/displayed automatically as part of an application.

Work Package 6.2 is a prototype application aimed at exploring this type of solution[1]

A number of methodologies have been constructed with the aim of providing documented processes by which a consistent set of procedures can be followed to capture and manage metadata. In engineering these include:

- MOKA -
  ☞http://www.kbe.coventry.ac.uk/moka/default.htm
- CommonKADS -
  ☞http://www.commonkads.uva.nl/frameset-commonkads.html

Other methodologies exist in other industries and domains. There are a number of areas where ontologies and the structured content capabilities of XML may be brought to bear in the problem.

References

[1]WP6.2 Prototype implementation of Building knowledge objects from disparate, related resources
☞http://www.w3.org/2001/sw/Europe/reports/WP6.3PrototypeDocumentation.htm