Architectural Meaning in the Existing Architectural Notations

The Technologies for Interoperable Architectural Data Management

SZEWCZYK Jaroslaw

Bialystok University of Technology, Faculty of Architecture
ul. Krakowska 9, 15-875 Bialystok, POLAND
cad.cam@interia.pl

The paper presents results of investigations upon possible representations of architectural objects containing 'architectural meaning'. Five groups of such existing representations are analyzed briefly: (1) structural EXPRESS–based definitions, (2) object EXPRESS- and IFC- based notations, (3) XML-based notations, (4) experimental ones, and (5) binary notations. The working taxonomy of the notations is presented and their shortcomings are mentioned.

The potential of XML notations for CAD data has been recognized by software vendors and standard organizations all over the world. Many corporations and standardization bodies are developing XML-based notations of CAD data, focusing on interoperability problems. Some of these notations are becoming standards (aecXML, i-drop), influencing the development of all CAAD industry. There are some main groups of problems with the existing commercial standards: Lack of 'open' data in its 'open' context, lack of architectural meaning in commercial notations, problems with simpler intuitive standards for notation of conceptual design data in early design stages, too complex data semantics, too atomized data, and richness of the data structures. The problems are taken under consideration in order to discuss the present state of architectural standards.

Nowadays architects are forced to work with semi-architectural notations, lacking their essence, i.e. lacking methods to describe elements of cultural heritage connected to geometry forms. Instead of language of architecture they deal with virtual cyber-slang.

***Keywords**: Architectural databases; architectural notations; interoperability, standards, XML*

The Problem

The problem in CAAD is a great variety of semi-interoperable data notations suffering from a poor architectural meaning. CAAD vendors and organizations have made many efforts, trying to solve the problem of interoperable A/E/C CAD data: ISO has developed first IGES, then Express-based STEP; International Alliance for Interoperability – Express-based IFC definitions (www.iai-na.org), and recently GDL language has been well known, as a result of Graphisoft's efforts followed by the activities of GDL Alliance and GDL Technology (www.gdlalliance.org, www.gdltechnology.com). Some standards were built upon XML

basis. W3C announced in February 1999 that it has started the development process of XML-based notation for VRML language (X3D). For last three years many vendors have begun to develop XML schemas as a panacea for problems of interoperable architectural data management.

The basic limitation of these technologies is defining new data sets unforeseen in these standards, and processing complex data in its flexible, open context. This implies problems with creating architectural databases. The existing, fixed data sets are too complex, difficult to deal with, they are often inconsistent and in result they are not really oriented towards architecture (Szewczyk, 2002a).

The results of investigations upon the existing representations of architectural objects are presented in the paper. The analyses led to the conclusion that the architectural notations lack the architectural meaning.

The Existing Notations of Architectural Data

Structural Express–based Definitions: STEP and STEPML

STEP is not an architectural standard, but - due to its popularity – there were several attempts to construct STEP-based models of architectural data, for example Popova (1998), Watson and Wambugu (2000) and Poyet, Zarli, Besse and Monceyron (1996). STEP is defined using a formal specification language, EXPRESS, and it is organized in a series of parts, grouped as follows: (1) description methods, (2) integrated resources, (3) application protocols, (4) abstract test suites, (5) implementation methods, and (6) conformance testing. STEP uses application protocols (APs) to specify the data representation for different applications. Only a few of APs can be applicable in AEC: AP230 (Structure), AP232 (PDM), AP201 and AP202 (Draughting), AP225 (Building and Construction), AP228 (HVAC), AP212 (Electrotechnical). It is expected that hundreds of new APs may support new industrial applications soon. Now STEP can't really compete with other notations in AEC application fields. None of architectural applications uses it natively.

STEPML is a library of XML Data Type Definitions, based on the STEP. The ISO 10303-11 EXPRESS data modeling language PDM Schema version 1.2 (upon which the DTDs are based) consists of 2100 code lines, and provides the STEP semantics to be used within the web. It has all the STEP-specific limitations and generally is not dedicated for architects, but it can be used for exchanging general CAD data rather.

These are not the only STEP-and-XML-based notations. For example XMLPlant/ STEP Schema has been created for plant industry. Other, like a PDES (Product Data Exchange Standard) are on the margins of architects' interests.

Object Express- and IFC- based Notations: IFC, IFCXML and BLIS-XML

IFC specifications are defined using the EXPRESS language. IFC consists of a great variety of class definitions, covering many applications, such as architecture, building services, structural engineering, project management, cost estimating, planning, maintenance and space management.

IFC seems to serve well cost estimating and architectural software (ADT, ArchiCAD, ALLPLAN, VISIO 2000, ARCHITREND), but it hasn't been implemented commonly. It has its strong limitations, too: (1) Only a part of the IFC classes is dedicated strictly to architects, so IFCs lack consistency in defining architectural databases; (2) IFC structure is 'software-oriented'; and (3) it is very 'atomized' (it consists of the four layers and a great number of classes).

IFCXML defines a subset of IFC classes using XML syntax. 70 percent of definitions do not describe spatial data, and 50 percent include data used rather by non-architectural applications, such as cost estimators. The IFCXML syntax consists of 2269 code lines, so it is rather a big, 'software' notation not only for architects.

BLIS-XML is a notation based on Express and XDR (XML Data Reduced), developed by Building Lifecycle Interoperable Software - a small group of members of IAI (www.iai-na.org). BLIS-XML was

defined to provide efficient IFC import/export using XML syntax, and to enable the use of IFC model data in Web browsers. It has a reduced, but very effective syntax, so potentially it can be effectively implemented in scientific architectural .

These are not the only attempts to translate IFC into XML language. An aecXML version of the IFC has also been generated.

XML-based Notations for AEC

AecXML, the most popular and the biggest XML notation for AEC industry (developed since 1999 by IAI) includes some stand-alone standards (such as LandXML). Now about 700 organizations, companies and individuals are involved in aecXML improvement. Its syntax is worked out by 7 working groups: Catalogs Working Group; Design/Specification/Scheduling/Costing W.G.; Facility Management Operations & Maintenance W.G.; Procurement W.G.; Project Management W.G.; Project News W.G. and Plant W.G. They pay attention to data interoperability between many application fields through product lifecycle, but they do not focus on architecture itself neither on strictly architectural notations. Educational, experimental and scientific usages of aecXML are on the margins of developers' interests.

GbXML (the GreenBuildingXML schema), published in 2000 by GeoPraxis, describes construction of buildings for the purposes of energy and resource analyses. It provides data interoperability between CAAD and building analysis software (such as DOE-2.2). It contains 4091 code lines, defining 261 elements and 85 data types.

DesignXML, on the opposite, is a small and simple set of XML definitions describing geometry in 3D space. It is rather geometry-based and not sophistically oriented towards architecture, but it defines a common vocabulary for 3D geometry and graphics representation (it can represent DWG data in XML syntax, replacing DXF format). DesignXML seems to be interesting due to its simplicity and in the future it can stimulate the development of more advanced architectural notations. It has been optimized for working with other existing XML formats such as SVG. Its syntax allows for association XML data with undefined or external data using 'Link' elements, which can be used to associate DesignXML elements with other data, including records in external databases.

BcXML is the building-construction XML dialect developed by the European organization eConstruct (Electronic Business in the Building and Construction Industry) for supporting electronic business in AEC domain, and for integration of engineering AEC applications.

AdpML is an Autodesk solution for developing electronic design catalogs that supports hierarchical part classification and allows self-publishing to a central searchable directory. It is useful in A/E/C, but rather not as real architectural notation.

RedlineXML enables to exchange digital design revision markups relative to design documents. It has been incorporated into some existing CAD/PDM software, for example Autodesk's Volo View.

LandXML schema facilitates the exchange of Land Planning, Civil Engineering and Land Survey data. LandXML has been embedded in aecXML. There are the attempts for converting LandXML coordinate geometry to GML (Geography Markup Language) in order to being used by GIS databases. Now the LandXML Design Data Model consists of 121 elements and its code contains 2878 code lines.

I-drop is an Autodesk XML-based technology which enables to create web content with the ability to drag and drop design content from a web page directly into design. I-drop works with the software, which have installed i-drop ActiveX controls (idrop.ocx). The i-drop ActiveX control is data-ignorant, because it's only responsible for transferring bytes and it doesn't interpret the data. Therefore i-drop can't be regarded as a stand-alone technology for data description in architectural databases.

SVG is an XML grammar for describing two-dimensional graphics. It includes elements for vector graphic shapes, raster images, animation, and text. There are successful attempts for applying SVG notation for describing GIS and GIS-relative information, including maps, site plans etc.

X3D has been developed by W3C since February 1999. X3D is going to be an XML-based notation for VRML language, which was used both as architectural data notation and as communication/interfacing technology.

Experimental Notations

There are many experimental notations of architectural data. Most of them were developed, but not implemented in commercial software. A few notations were based on commercial standards, i.e. on STEP (Popova, 1998; Watson and Wambugu, 2000; Poyet, Zarli, Besse and Monceyron, 1996) and on XML (**Multidimensional XML, XML for VR-DIS**). Most of the experimental notations were developed in order to solve some data-specific problems or to analyze individual aspects of architectural databases and in fact they can't be working commercial standards.

Binary Notations

DWG and **DGN** are the most popular binary formats (*de facto* standards) not optimized for AEC data structure. Object version of DWG (created by Architectural Desktop) is AEC-oriented, but not fully compatible with popular CAD software (AutoCAD needs *Object Enabler* to read it). DGN is not an architectural standard, so that Micro-Station architectural add-ons (TriForma, Speedikon) must write data in their own databases.

GDL (*Geometric Description Language*) is a scriptable language similar to BASIC, creating an open system for building parametric CAD objects. It can store 3D data, 2D data and property data, so it is used for creating parametric libraries of architectural objects (but, comparing the concurrent *i*-drop technology, GDL libraries contain parametric objects and they are much smaller – 50-100 times!). The existing GDL technology enables data interoperability between GDL and IFC. That is why a couple: GDL and IFC can potentially a good base for notation of architectural data (IFC for full A/E/C data notation, GDL for compressing elements in an A/E/C library).

O2c (*objects-to-see*) is a technology for creating parametric 3D objects, competing to VRML standard. *O2c* objects are even 3-5 times smaller than VRML ones. There are many existing libraries of *o2c* architectural objects shows, and this format is implemented in CAD software (in Speedikon, ArCon, DataCAD, SPIRIT 10 and – *via o2c-Player for MicroStation* – in MicroStation). O2c files can contain definitions of geometry, lights, materials (including transparency) and other attributes.

VRML and **Cult3D** are other standards for 3D data description, sometimes implemented in CAAD software.

The Taxonomy

The working taxonomy was developed in order to organize common AEC data notations. The taxonomy is presented below (Table 1).

Looking for the Architectural Meaning

The existing data notations in A/E/C are too atomized, 'software-oriented' and they lack consistency in defining architectural content. The developers of the largest notations (like aecXML and IFC) have not focused on architecture itself neither on strictly architectural notations and they have paid most attention to data interoperability between application fields through product lifecycle. That is why these standards have neither simple nor intuitive object definitions, they have poor 'architectural semantics', they contain a great number of fixed "technical" information and they do not carry many important information that can be called 'conceptual': aesthetic information, architectural meaning, etc.

The notation of intellectual property value, classifications of spatial forms and architectural object types are the examples of data with architectural meaning. The architecture itself is not only the geometry of shapes – it has its roots in cultural heritage, which is not represented in CAD data notations. These aspects are wasted in the commercial notations are they are partially worked out in the experimental notations.

Nowadays architects are forced to work with semi-architectural notations, lacking their essence, i.e. lacking methods to describe elements of cultural

THE EXISTING ARCHITECTURAL NOTATIONS IN CAD		Native notations	Notations optimised...	
			...through XML syntax	...other
***EXPRESS*-based**	Structural	**STEP**	**STEPML** XMLPlant/STEP*	PDES*
	Object-oriented	**IFC**	**IFCXML** **BLIS-XML**	**BLIS IFC**
***XML*-based**	AecXML dialects (Bentley initiative)		**aecXML** dialects (**LandXML,** etc.)	
	DWG-optimised (Autodesk initiative)		**DesignXML** **RedlineXML** adpML* ***i*-drop**	
	Independent		**gbXML** **bcXML**	
Binary	Basic (CAD)	**DGN** **DWG**	**SVG**	
	Object-oriented	**GDL**		**DWG** (ADT)
	3D/VR	**VRML** **o2c**	**X3D** **o2c**	**VRML 2.0**
Experimental	Based on STEP standard	*a few*	*a few*	
	Based on XML syntax		**XML for VR-DIS**	
**- notations not oriented towards architecture and on the margins of architects' interests*				

Table 1. The existing architectural notations

heritage connected to geometry forms. Instead of language of architecture they deal with virtual cost-working slang.

The Data for Conceptual Design

In fact there are no standards for notation and for interoperable management of conceptual design data in early design stages! Some software notations (GDL, o2c, VRML) tend to be such standards, but software implementing them (except of few VRML interfaces) is not optimized for conceptual architectural modeling.

Only a few notations have really interesting grammar (DesignXML and some experimental ones) or they present inspiring solutions, which can make them a good base for further development of real architectural notations for conceptual modeling stages.

Architectural Notations for Scientific Purposes

There are no standards for scientific architectural notation (except of educational databases)! Architectural databases for scientific investigations must have very specific data layout. They have to include a very large amount of heterogeneous information, which is needed to be structured in many ways. Besides, it is often impossible to fix the structure of information, and all the pieces if information can belong to many data structures and can be accessed for many different purposes. This implies problems with defining 'open' (not fixed), big structures of architectural data, optimized for scientist's needs. Some aspects of these problems were mentioned by Korolczuk and Szewczyk (2002) and Szewczyk (2002).

In fact notations of 'open' architectural scientific databases were not developed until now (expect of few very specific ones, for example educational).

Defining Richer Architectural Information Structures

An architectural analysis is expressed through a number of abstractions (textual, drawings, models, images, etc.), reflecting specific aspects of the objects (such as function, structure). Abstractions and the relationships between them define the information structure and they present its different aspects. A richer information structure can be achieved by expanding the abstraction structure, replacing abstraction entities by detailed component substructures, and by augmenting the structure's relatedness with content information (Tunçer and Stouffs, 2000).

The existing "richer" A/E/C notations (especially IFC- and XML-based) suffer from data redundancy (data are redundant in order to achieve compatibility with application-specific data layouts or as a result of a poor synchronicity of developers' efforts). There are some proposals for developing not-redundant richer information structures: for example Tunçer and Stouffs (2000) suggest the adoption of an XML syntax for representing the data abstractions, enabling them to be interpreted and broken up into components, and the relationships to be added. The result is a richer information structure of components and relationships, providing views which are new and not inherent to the original structure of abstractions, and offering new interpretations of data model.

XML: The Direction in Developing Architectural Notations

The analysis of the existing standards shows the increasing popularity of notations based on XML syntax. Most of A/E/C notations are XML dialects, some other has been optimized by adding XML syntax. The possible influence of XML on A/E/C notations are presented below (Table 2).

Conclusions

Existing standard CAD notations (enabling application interoperability and management of heterogeneous information) haven't been optimized well for architectural data storage yet, they are oriented towards cost-working applications, too complex, too atomized and often not well implemented in the existing software;

Commercial standards imply problems with architectural meaning (intellectual property value, aesthetic information, cultural context, etc.);

The existing notations are not optimized towards managing 'open' data structures;

There are no standards for notation and for interoperable management of conceptual design data in early design stages;

Defining standards for open, 'richer' semantically architectural data sets is required to build architectural databases for scientific analyses and conceptual design;

The popularity of notations based on XML increases rapidly, but XML syntax has many shortcomings limiting its usage as a stand-alone language for A/E/C

Merging XML notation and CAD software-specific binary notations (managing it with CAD viewers and Web browsers) seems to be rather effective way to handle architectural data stored in CAD libraries; but there are no stand-alone notations which could effectively replace such merged heterogeneous technologies;

Detailed investigations of scientific architectural notation are needed to work out software technologies and methods for scientific, semantic-insensitively communication and intuitive CAD data handling.

Acknowledgements

This work is supported by rector's grant at Bialystok University of Technology (Faculty of Architecture), Grant No. W/WA/4/00, funded by Polish Research Council.

References

Badard T. and Richard D.: 2001, Using XML for the exchange of updating information between geographical information systems, Computers, Environment and Urban Systems 25(2001) pp. 17-31

The possible advantages of XML usage	The limitations of XML usage
1 XML can help with storage, integration, access and fast digitization of very large tree-like data structures containing heterogeneous information used in scientific investigation.	1 The technology lacks its internal integrity and is still under development (W3C Consortium haven't standardized many XML solutions yet).
2 STANDARDS: XML is a widely adopted standard, and is popular especially in IT world – it has became the key technology in some PDM applications.	2 XML-based concurrent notations are not interoperable enough and haven't been developed definitely (aecXML schemas, DesignXML).
3 TOOLS: There are many XML parsers which can validate XML documents. XML schemata and parsers can be easily used for checking database consistency.	3 The existing technologies (BLIS-XML technology, adpML, RedlineXML, IFCXML) have limited usage until now.
4 IMPLEMENTATIONS: XML is being implemented in CAD, Web and PDM software and (through interfaces) in commercial databases.	4 Some XML-based notations of architectural data are in fact quite large (aecXML, IFCXML, DesignXML) and they are difficult to implement in modeling software;
5 DATA NOTATION: XML allows meaningful, content-oriented data notation, providing means for building many hierarchies for the data.	5 CAD data needs conversion to the existing XML dialects.
6 DATA PROCESSING: Its syntax can be used to describe large hybrid data sets in a relatively simple manner; it allows efficient processing of semi-structural data.	6 The basic limitation of these technologies is defining new data sets unforeseen in XML Schemas. This implies problems with creating open, changeable, XML-based data sets; and the key problem is how to manage complex architectural data in its open, not-fixed XML-based context.
7 METADATA CONCEPT: Its metadata concept makes it possible to display a great number of various data efficiently, even if using popular web browsers.	7 Besides, the XML-based CAD data notations suffer from rough semantics.
8 SIMPLICITY: Its easy to learn and to implement syntax consists of simple rules.	8 Semantics of XML-notated 'architectural' data are in many cases oriented towards needs of cost-working applications (the development of many XML-based technologies has been inspired by vendors of cost-working applications, and not by architects, so these technologies haven't been optimized for managing intellectual property value and they lack the architectural meaning.
9 DATA STANDARDS: XML gives methods and tools for development and sharing of meaningful data standards.	9 Fisher, Burry and Woodbury (2000) noticed, that XML itself was purely syntactical with no means for semantic definitions, so that data might easily be misinterpreted, and XML do not represent well design tasks, which consist of an unpredictable number of elements with an unpredictable number of attributes. These researchers write: "Interoperable standards require the opposite: experiences made in well-known problem spaces upon which a finite number of data elements and attributes can be collected and become specified as a problem-oriented communications protocol."
10 TEXT FILES: Both data structure and data itself can be written in ASCII text files or linked to ASCII files (every text editor can be used to modify these files).	10 Text files (and especially XML ones) are too big: a DXF text file is 2,5 times larger than an identical DWG one, and the same DesingXML one is about 10 times larger than DWG!
11 UNICODE: Data notation based on Unicode makes it language-independent	
12 SOFTWARE INTERFACE: It can provide a consistent interface, mediating among the various retrieval systems.	
13 USER INTERFACE: The XML usage can add value to user interface, for example the XML document can have many independent presentation styles.	
14 FLEXIBILITY: Data syntax, data itself, and display methods can be independently defined, exchanged and stored in different documents.	
15 COMPATIBILITY: XML notation is platform-independent and it increases data compatibility with other data sets and with external software.	

Table 2. The advantages and the disadvantages of XML usage for architectural data description

Fisher, T., Burry, M. and Woodbury, R.: 2000, Object-Oriented Modelling Using XML in Computer-Aided Architectural and Educational CAD: The Problem of Interoperability exemplified in two Case Studies, in CAADRIA 2000: Proceedings of The Fifth Conference on Computer Aided Architectural Design Research in Asia, Singapore, pp.145-155.

Houlding S. W.: 2001, XML - an opportunity for <meaningful> data standards in the geosciences, Computers & Geosciences 27(2001) pp. 839–849

Korolczuk D., Szewczyk J., 2002: XML Schema for Investigations of Polish Traditional Rural Architecture, in Proceedings of the 7th International Conference on Computer Aided Architectural Design Research in Asia : CAADRIA'2002, 18-20 April 2002 Cyberjaya, Malaysia, Prentice Hall, pp.65-72

Lan J. and Jeng T.: 2001, Enhancing shared understanding in collaborative design communication - An XML approach, in Gero J. S., Chase S. and Rosenman M (eds), CAADRIA 2001, Proceedings of the Sixth Conference on Computer Aided Architectural Design Research in Asia, Sydney 19-21 April 2001, Key Centre of Design Computing and Cognition, University of Sydney, 2001, pp. 285-289

Popova, M.: 1998, Model of Design Parts and its Use to The Design Team, in T. Sasada, S. Yamaguchi, M. Morozumi, A. Kaga and R. Homma, (eds.), CAADRIA'98: Proceedings of The Third Conference on Computer Aided Architectural Design Research in Asia, Osaka, Japan, pp. 233-242

Poyet P., Zarli A, Besse G. and Monceyron J.L., 1996: KBS and STEP Architectural CAD Systems. The integration of STEP compliant KBS and CAD systems in Building Design. URL: http:/-/cic.cstb.fr/ILC/publicat/Poyet.pdf: May 2002

Shanmugasundaram J., Shekita E., Barr R., Carey M., Lindsay B., Pirahesh H. and Reinwald B.: 2001, Efficiently publishing relational data as XML documents, The VLDB Journal (2001/10) pp.133 –154

Shaw N. and Kimber W.E.: 1999, STEP and SGML/XML: what it means, how it works, in XML Europe '99 Conference Proceedings, Graphic Communication Association, 1999, pp.267-70.

Stouffs R. and Krishnamur R.: 2000, Alternative Computational Design Representations, in SIGraDi'2000 - Construindo o espacio digital (Constructing the Digital Space), 4th SIGRADI Conference Proceedings, Rio de Janeiro (Brazil), 25-28 September 2000, pp. 200-202

Szewczyk J., 2001: Engineering Portals: Networked Architectural Information Management, in Penttilä H. (ed.), 19th eCAADe Conference on Architectural Information Management, 29-31.08.2001 Helsinki, Finnland, Helsinki University of Technology, pp. 150-155

Szewczyk J., 2002: The Limitations of Architectural XML-powered Databases: Open Standards Versus XML Glue, in Proceedings of the 7th International Conference on Computer Aided Architectural Design Research in Asia: CAADRIA'02, 18-20 April 2002 Cyberjaya, Malaysia, Prentice Hall, pp.49-56

Tunçer B. and R. Stouffs: 1999, Computational Richness in the Representation of Architectural Languages, in Architectural Computing: from Turing to 2000, Brown A., Knight M. and Berridge P. (eds.), Liverpool, eCAADe and The University of Liverpool., pp. 603-610

Tunçer B. and Stouffs R.: 2000, A Representational Framework for Architectural Analysis, in SIGraDi'2000 - Construindo o espacio digital (Constructing the Digital Space), 4th SIGRADI Conference Proceedings, Rio de Janeiro (Brazil), 25-28 September 2000, pp. 206-208

Van Leeuven J. P. and Jessurun A. J.: 2001, XML for Flexibility and Extensibility of Design Information Models, in Gero J. S., Chase S. and Rosenman M. (eds), CAADRIA 2001, Proceedings of the Sixth Conference on Computer Aided Architectural Design Research in Asia, Sydney 19-21 April 2001, Key Centre of Design Computing and Cognition, University of Sydney, 2001, pp. 491-501.

Van Leeuven, J.P and Jessurun A. J.: 2001, Added Value of XML for CAAD, in Proceedings of AVO-CAAD 2001, Brussels, Belgium, April 5-7, 2001

Watson A. and Wambugu W.: 2000, A Product Model Based Architecture for Engineering Applications Software, in www.cae.civil.le-eds.ac.uk/information/papers/CIT2000.htm.