A SURVEY OF INTERNET-ORIENTED TECHNOLOGIES FOR DOCUMENT-DRIVEN APPLICATIONS IN CONSTRUCTION OPEN DYNAMIC VIRTUAL ENVIRONMENTS

Alain Zarli[1], **Yacine Rezgui**[2]
[1]Centre Scientifique et Technique du Bâtiment
290, route des lucioles, B.P. 209, 06904 Sophia Antipolis, France
zarli@cstb.fr

[2]Information Systems Institute
University of Salford, The Crescent, M5 4WT, Salford/Manchester, UK
Y.Rezgui@salford.ac.uk

ABSTRACT: Recent years have seen the emergence and development of a plethora of standards, tools, techniques, and methods for wide enterprise concurrent engineering. These developments have enabled and promoted, in most industries, new forms of collaboration between actors / companies, within and across geographically dispersed locations. While the concept of a Virtual Enterprise (VE) is gaining an increasing popularity and acceptance, the construction industry has adopted for decades its modus operandi, without taking much advantage of the opportunities offered by current Information and Communication Technologies (ICTs). This paper, based on research conducted within the frame of the European OSMOS (IST-1999-10491) project, proposes a survey and first evaluation of open Internet-oriented technology and standards for use in open dynamic virtual environments, analysing the potential usefulness of technologies like STEP, XML, CORBA, and MOM. The benefits of using XML, as a technology for flexible and dynamic representation of complex objects (including documents) and their stream-based interchange, are highlighted along with distributed object frameworks such as CORBA for remote access to data. The scope of this evaluation addresses the specific needs and peculiarities of the construction industry.

KEYWORDS: Construction open virtual environments, information exchange and sharing, application integration, middleware technologies, Internet technologies.

1. INTRODUCTION

From standards for product / process modelling and data exchange, to integration through the use of message-oriented or object-oriented technologies, a lot of developments for enterprise applications integration have been experimented and used in order to highlight the benefits of information and communication technologies in the construction domain. Indeed, integration and seamless communication between (proprietary and commercial) software applications are increasingly becoming fundamental issues tackled by a variety of emerging technologies. However, all these latest IT developments did not have so far the expected impact on the construction industry, due to several factors, including poor investment in construction IT by SMEs, and mismatch between IT innovations and construction industry needs. In fact, behind the global issue of application integration within the (real or virtual) enterprise, appear a lot of architectural problems (along with the "right" selection of tools, toolboxes and infrastructures) that take a critical dimension in the case of open systems issues.

The building sector is essentially characterised by its fragmentation, with a high proportion of SMEs involved in the design and build process of Construction projects. In fact, from a

macroscopic point of view, Extranets and Virtual enterprises should be supported and deployed easily in order to provide smooth co-operation between actors for all stages involved in the design and delivery of the building product. Consequently, the problematic is not only to ensure internal (*intra-company*) communication between systems as in a single, even large, company, but to provide the required capacity to exchange information and collaborate outside the boundaries of the information systems constituting the IT infrastructure of a company, including means to organise the flow of information and the co-ordination of tasks in the context of *inter-companies* communication. The main key points that identify the differences between these two situations can be summarised as follows:

- While within a single company, the various applications can be strongly connected using proprietary technologies (that can be possibly based on *corporate* standards), it becomes mandatory to rely on industry standards within the VE. This is due to the most of the time divergent environments and capabilities of the legacy systems that exist in the different companies forming the VE and that have to be connected.
- Systems in a single company can be fully integrated, and made in some way deeply inter-operable between each other. Within the VE, these systems are more or less easily integrated, depending on whether they provide just some rough gateways, or that they publish a full API (e.g. through IDL interfaces). Moreover, the VE often requires the development of common repositories and datawarehouses in order to deal with information that must be shared by the various information systems.
- The heterogeneous and unknown nature of the various systems to be inter-connected in a VE makes it very difficult, if not impossible, to agree *a priori* on the level of information and communication technologies to deal with, each time a new project starts. For instance, in the case of data publication on the Internet, it may become necessary to deal with clients desktops potentially having different data management capacities (e.g. a simple Web Browser, or a browser powered with XML tools, or a CORBA-compliant client, or a mobile phone, etc.).
- Projects in the construction industry are one-of-a-kind projects involving short-term partnering between non co-located actors. This necessitates agile and flexible deployment of ICT infrastructures to support the VE. These infrastructures should be set-up in a matter of days or a couple of weeks, as opposed to several months (as it is the case today).

To address the issues of communication and interoperability, several routes have been explored and are still under development. Moreover, after an initial phase related to bulk exchange of models and documents, middleware has become the focus, with investigations around CORBA or DCOM (for object-oriented middleware-oriented application integration), and MOM (for message-oriented middleware and the routing and formatting "on-the-fly" of messages). More recently, new architectures have been suggested, especially the now well known 3-Tier based architecture and application servers (Client desktop – middle-tier Web or application server – DBMS for persistent storage). Eventually, technologies dedicated to the Web have emerged, particularly the XML technology and Java-oriented technologies (e.g. servlets, JSP and the EJB).

This paper introduces to the initial architectural works that have just started within the frame of the OSMOS[1] project. Driven by the identification of intra- and inter-company business processes and information / process requirements of the Construction domain, and also on

[1] OSMOS is a European RTD project within framework V: IST-1999-10491, *Open System for inter-enterprise information Management in dynamic virtual envirOnmentS*. The consortium includes construction IT service providers: *DERBi*, *JM Byggands*, *Olof Granlund*, and European leading research centres and academic: *CSTB*, *Information Systems Institute* of University of Salford, *VTT*.

case studies and previous experiments conducted in former European projects, including VEGA, GENIAL and CONDOR, the main objective of OSMOS is to develop and promote optimised solutions to support effective information sharing and smooth co-operation between non co-located teams. This paper proposes a survey and first evaluation of open Internet-oriented technology and standards for use in open dynamic virtual environments, and analyses the potential usefulness of some current technologies, including STEP, XML, CORBA, and MOM. It also highlights the benefits of using XML, as a technology for flexible and dynamic representation of complex objects (including documents) and their stream-based interchange, along with distributed object frameworks such as CORBA for remote access to data. The proposed evaluation mainly concentrates on needs related to the design of an infrastructure for application integration and tasks collaboration, and elaborates on how recent ICTs fit into architectures required for the specific needs and peculiarities of the construction industry.

2. STANDARDS AND TECHNOLOGIES

Recent and continuous investigations on the use of advanced computer-based technologies, especially in ESPRIT funded European projects like VEGA (Stephens J. et al. 1999), CONDOR (Rezgui Y. et al. 1999) and GENIAL (Radeke E. et al. 1999) among others, have already shown promising results. However, the major issue, indeed not only within the construction sector, but also within other industries, is to combine the foundation technologies to realise the enterprise applications integration in Intranets, Extranets or even over the Internet. As regards application integration within the VE, there are needs for:

- communication technologies, and especially middleware services, e.g. RPC, ORB, MOM, etc., along with issues to consider for effective Client/Server computing like platform and language interoperability, communication protocols, object management, security, openness and availability of standards, etc.;
- co-operation technologies, with the use of standardised shared information repositories, containing meta-information on documents and objects, as well as common structured information that are useful to all the applications (based on data filtering, transformation, aggregation, etc.);
- co-ordination technologies, including access control, events notification and tasks collaboration and synchronisation;
- documentation technologies, including document routing and version control.

The paper does not get into the details of co-ordination and documentation issues. It rather concentrates on the two first aforementioned items, providing with a first step towards an approach to solving communication and co-operation questions in open virtual environments, by establishing a common foundation to better take advantage of the benefits of distribution and Internet-oriented technologies.

2.1 Information modelling, exchange and sharing for improved co-operation

Systems that are to be provided first must be based on standards for data modelling and exchange. Besides the media used for communication between applications, those applications need to have a common understanding of information (e.g. its semantics). Such a common understanding will be more and more developed for each vertical industrial domain: standardised vertical models will ease the interactions between systems in a business domain. Two major types have to be envisaged: engineering product data, and metadata (i.e. data about information or data).

As regards product data, it is today possible to rely on standardised works that have reached a reasonable level of maturity. In the Construction industry, the current major works so far are:

- STEP (ISOa 1994), the ISO 10303 standard for the uniform representation and exchange of product data, including the EXPRESS language (ISOb 1994), a format for STEP physical files (ISOc 1994) and an API for common access and sharing of product databases (ISOd 1995) via data and application independent mechanisms.
- The IFC (Industry Foundation Classes) general model, developed by the IAI (International Alliance for Interoperability) as a universal model for integration and collaborative work in the AEC/FM industry (IAI 1997).

On the other hand, metadata are of primarily importance when dealing with common repositories and/or datawarehouses. After defining a set of potential repositories for the VE, workgroup can be realised on the basis of those repositories. Indeed, it is clearly acknowledged that today, users of workgroup-based tools are finding difficult to co-ordinate their software effort across the enterprise, especially due to the heterogeneity of enterprise repositories. Thus, new solutions must be provided in order to simplify heterogeneous, enterprise-wide interoperability among tools, object repositories and datawarehouses. Two current ongoing works seem today very promising and are worth mentioning:

- The Resource Description Framework (RDF), developed in the context of the W3C, and currently released as a W3C recommendation (RDF 1999).
- The Meta Object Facility (MOF), a specification coming from the OMG (MOF 1999).

Most information systems define their own combination of metadata and own facilities for storing and managing them, most of the time with no facility for sharing or interchanging metadata across systems. RDF is an attempt to deal with metadata and repositories that can be exchanged and accessed over the Web, i.e. in a fully open context and independently of the applications using them. Currently under development within the W3C, RDF provides with:

- A standard schema for general meta-information about documents, named the "*Dublin core*". This schema contains 15 properties such as title, creator, subject, contributor, date, etc., that aim at characterising any document that can be published on the Web.
- A common model for describing medatata and representing knowledge in terms of relationships between an object (any resource described by a URI - Uniform Resource Identifier, e.g. a URL on the Web, a specific section within a document, etc.), a PropertyType that identifies a resource (i.e. some piece of information) that has a name and can be used as a property, and the link (called a property) between the object and the PropertyType through a given value identifying an instance of the PropertyType. The underlying algebra allows for manipulating data structures as directed labelled graphs (which extends the basic tree-based structure of XML), and a specific XML grammar gives ways to exchange such metadata via XML documents.

The MOF provides with a common basis for meta-models (i.e. models of meta-information) so that applications dealing with different meta-models that are MOF compliant, can exchange meta-information among them and can manage or access common repositories. The set of core structures that form the MOF are independent of any specific standard model: for instance, the MOF is independent of the Java meta-model, and so independent of the Java language, it is independent of CORBA IDL as well, i.e. it is not linked to the specific CORBA middleware meta-model. It is also independent of the UML meta-model (UML is the OMG's object-oriented design language), but the UML meta-model is MOF compliant, meaning that UML tools have the potential to share integrated repositories.

An interesting associated development is XMI, that is a new OMG open industry standard that combine XML's ability to define, store and share documents formats on the Web with the features of UML. For instance, XMI allows to interchange UML conceptual models between modelling tools or repositories, the representation of an UML model in a given tool being translated and kept in an XML Ascii file (with a specific UML DTD). Then, any other XMI-compliant modelling tool can decode and recover the initial UML model. This enable development teams, users and customers to use the more adequate tools (i.e. delivering the more adequate features) from a variety of vendors, and developers will streamline their collaborative application development efforts.

2.2 Communication services

With some common data semantics and shared repositories, applications then must inter-operate despite the heterogeneity of systems. However, a major previously identified constraint is that systems have more and more to be adapted to interactions in a context of Extranets and the Internet as well, making the task fairly complex. The various technologies that are candidates to support the future VE applications will need to have the ability to easily integrate with other technologies. Moreover, they should be either, generic enough (and thus probably "simple" enough) or they should provide adequate gateways, in order to allow the selection of "best-of-breed" modules supplying the best solution to end-users needs.

To address the issues of communication and interoperability, various developments have emerged, with a particular focus that has been put on middleware technologies. The latter include CORBA (today, the new 3.0 specification) and its family of various commercial or free implementations, the Microsoft's COM+/Active X model (with DCOM), Java/RMI, and MOM (a technology supplying asynchronous mechanisms that can be found within tools like IBM's MQ-Series or Microsoft's MSMQ). The aim of the paper is not to detail these technologies, as they have been extensively described in the literature. In a parallel track, the Web has had in recent years a tremendous impact on industry, in terms of new approaches for communication. As regards Web clients, a plethora of tools and technologies has emerged: stream-oriented languages like HTML, DHTML, VRML and now the XML family, scripting languages like Perl, Javascript, etc., and server-side programming like CGI or ASP, and of course Java. Initially a language developed so as to be light enough for small programs on embedded systems, Java has grown to a set of programming techniques and tools that have invaded the Web. The main Java-based technologies today are servlets, JSP, EJB, and JTS:

- Java Servlets are programs running on the server side, acting in similar way than CGI scripts and using the HTTP communication as well, but getting benefits of Java technology (including multi-threading).
- JSP, the Java Server Pages, is built on top of servlets. It can be viewed as a standard on the server side for dynamic generation of HTML code, with Web pages embedding Java code compiled in reusable servlets. Especially, JSP allow developers to separate functions for creating content from those for presentation.
- EJB is the components model developed by SUN, and that can be viewed as a generalisation of the Java Beans, but on the server side, which means with all the underlying infrastructure to deal with transactions access to data bases, etc.. EJB aims at supporting business components on the server.
- JTS is the Java Transaction Service, enabling transactions in Java-based environments.

Likewise, it is worth mentioning that the Java platform deals with its own CORBA implementation, and ships with a lot of various APISs, including JMS, the Java Messaging Service that provides asynchronous messaging in Java (i.e. a Java MOM).

3. A GENERAL ARCHITECTURE FOR CONSTRUCTION OPEN DYNAMIC VIRTUAL ENVIRONMENTS

An attempt for a general architecture for construction VEs is suggested in Figure 1. Indeed, in order to deal with any type of client, especially on the Internet, a loose coupling should be provided that allows the enterprise business logic to be independent from any presentation and client desktop issues. Note that the "Business Components" layer only manages what refers to the business of the enterprise (e.g. a banking transaction, a digital simulation of the behaviour of construction components based on a mathematical model, a notification module in a document management system, etc.). The specific ways information is requested and results are further visualised have not to burden the business process itself and to influence the information resulting from it. A similar model of loose coupling can be envisaged between the intermediate layer and corporate information warehouses, once again in order not to mix the application logic with the specific rules to access a given database.

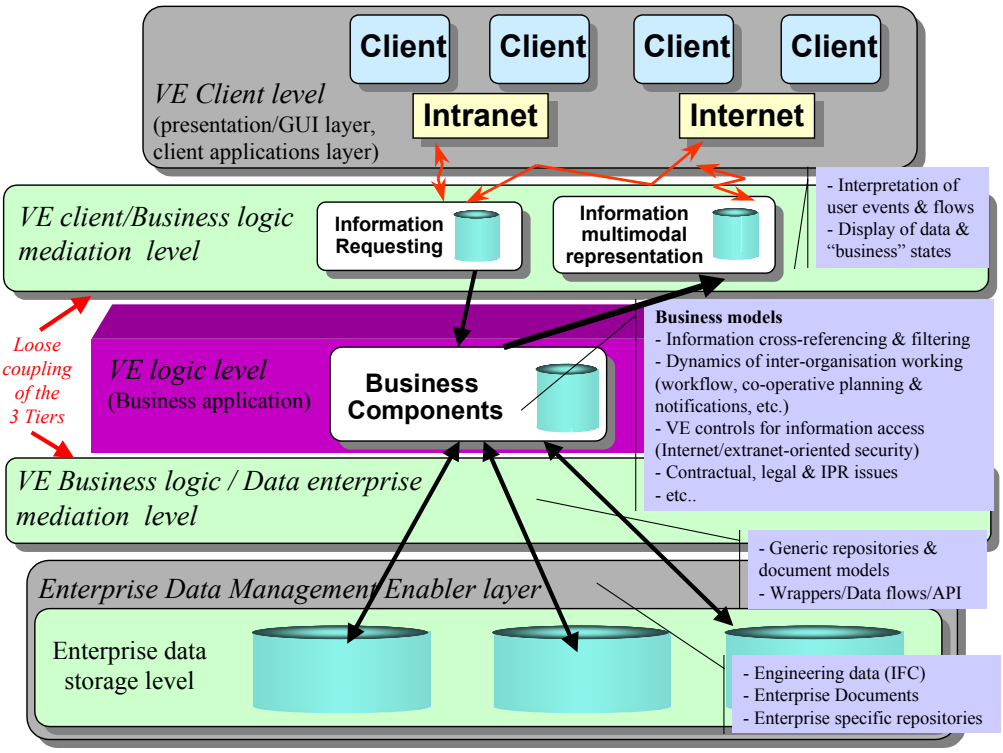

Figure 1. An architecture for construction open dynamic VEs.

As previously mentioned, a fundamental issue is information access and transfer between the different layers of the proposed architecture, and between all the components that form each of these layers. There are several blueprints that can be used in order to transfer data (and in general information) from one system (in that context a server) to another system (a client):

- Model-based transfer: in that case, when the client wants to get some information, the whole model (e.g. a full STEP model) is transferred from the server to the client. This is a total transfer, typically deployed with the STEP SPF technology.
- Object-based transfer: whenever the client needs an information, and that information is not within the client yet, only the referred object (or set of objects) is transferred. This leads to a minimal transfer of information, which can be implemented using XML.

- Method-based access: whenever the client needs some data, it calls the right method on a remote object (the method is declared within an interface for this object). This approach is the classical one as used in CORBA.
- Operations-based transfer: this last specific design consists in not transferring the data themselves, but the sequence of operations that allow to rebuild the whole set of data on the client side. It reveals useful in order to avoid transferring very huge sets of data.

According to the VE context, the expressed end-users requirements, and technical constraints, each of these blueprints can have their own supremacy. Some of the criteria for the selection of the appropriate method (and its counterpart in terms of technology) can be exhibited, e.g.:

- Management of messages;
- Protocol issues;
- Shipping of large set of data *versus* small set of data;
- Data duplication and control of the coherency (need for check-in/check-out mechanisms);
- Synchronous *versus* asynchronous mechanisms.

In the remainder of the paper, we first briefly introduce XML, and then try a first comparison of the respective benefits and shortcomings of STEP, CORBA, MOM and XML taking into account the aforementioned criteria.

4. XML IN THE CONTEXT OF A VIRTUAL ENTERPRISE

4.1 Main concepts of XML

SGML (SGML 1986), the Standard Generalised Markup Language (ISO 8879) has for a long time been a major reference to deal with structured documents, as a language for describing and inserting in a neutral way tags within any type of documents. As the Web appeared these last years as the major upcoming media in order to broadcast SGML-like document repositories, XML has been developed by the W3C originally for exchanging structured documents within Intranets or on the Internet, in a simpler way than when using SGML. XML is text-based, self-describing, and, above all, gaining acceptance as a global standard. The XML language is indeed a meta-language, allowing to create any (XML-based) new language, and especially what can be called “data presentation language” (for displaying, exchanging, storing data, etc.): this enables the definition of new file formats that can be instantly parsed by any XML-compliant application. Thus, for a particular domain, it is quite possible to create a new presentation (or exchange) semantics, which is however different of the semantics of the data themselves (i.e. the content of the XML message).

XML is a universal, easy to comprehend and very adaptable file format, well adapted to on-line catalogues and product configuration information. It manifests quite interesting features to support Web-based information servers, as a technology dedicated to large-scale Web content providers for industry, media-independent publishing, vendor-neutral data exchange, workflow management in collaborative (authoring) environments, and processing of Web documents by intelligent agents. By providing a standardised way of structuring data within Web-based information systems (when HTML only enables to mark-up the information for presentation), it allows to rely on this structuring for queries, thus leading to much more value-added results. XML also has been designed for quick client-side processing consistent with its primary purpose as an electronic publishing and data interchange format. Indeed, XML is rapidly gaining wide acceptance regarding two main potentialities:

- *Cataloguing and further distributing and recovering information on the client side*. Various components of the XML galaxy have been developed for that purpose, including XML-compliant browsers, XML documents searching tools, DTDs for design and control of various Web documents, style sheets for representation (with XSL: eXtensible Style-sheet Language), and the XML DOM (Document Object Model) standard object API specified by the W3C, that aims at giving developers programmatic control of XML document content, structure, and formats.
- *Data exchange and application integration at the enterprise level services*, where XML can be viewed as a standard way of passing data between many heterogeneous distributed application servers, as well as across multiple operating systems. We now detail this second XML target in the next section.

4.2 XML for Information exchange and applications integration

XML is both a technology for documents manipulation, and a technology for flexible and dynamic representation of complex objects and their interchange, along with insertion in distributed object frameworks, as XML documents can be exchanged using any underlying protocol. XML can be used to define the container for a message content, for any type of data provided by a repository. As regards data exchange and interoperability, XML can be used:

- Between datawarehouses and the middle layer (3-Tier architecture), for data integration and linking through XML documents conveyed or generated from multiples sources.
- Between the middle layer and the clients, for data delivery (XML over HTTP for Web clients, or delivery to other applications for further processing), data manipulation (via the DOM), creation of multiple views for XML data (e.g. with XSL), etc..

Thus, XML provides with data portability as a platform-neutral document description meta-language that offers means for data serialisation. It is worth noticing, at that point, that it can be quite beneficial to associate XML with Java, which offers code portability, as supporting the development of platform-neutral applications. The main points to be highlighted are:

- XML appears to have a potential role as a universally accepted format for the exchange of information between heterogeneous applications. XML is expected as one of the primary means for developers to design multi-tier applications in heterogeneous environments.
- XML is suitable for co-operative applications since dealing with documents and means to convey business knowledge. Especially, meta-information about the structuring of Web sites, databases, repositories and datawarehouses can be exposed through XML messages.
- XML provides a way of tagging data and objects as they are called for on a network, which provides as well with an automatic way of populating databases on the fly with XML (especially local databases for specific applications connected to Intra/Extranet). Besides, in order to deal with the semantics of data that compose the content of XML messages, current efforts are undertaken, among others in the W3C with DCD (Document Content Definition) and XML schemas, to define more semantics attached to an XML document content, including element names and rich data types. It is indeed now established that the concept of DTD, initially originated from SGML, presents shortcomings regarding the shipping of semantics about the contents of messages.

5. COMPARING MAJOR TECHNOLOGIES FOR USE IN AN OPEN CONSTRUCTION VIRTUAL ENVIRONMENT

This section exhibits some of the strong and weak properties of today technologies towards the basics required in the VE for information presentation, access and exchange, comparing these technologies with XML, in accordance with the criteria identified in section 3.

5.1 XML - HTML/DHTML

The positioning of XML against HTML and DHTML has been described extensively in the literature. HTML is used only for information presentation on the Web client desktop. DHTML (Dynamic HTML) is an improvement of HTML, that enables to consider a document as a set of independent elements (when HTML considers a document as a whole). This means that DHTML is able to deal with an underlying document model (the DOM) that identifies independent objects, the graphical properties of which can be modified using scripting programs. Operations on these objects are realised locally on the client side, thus leading to a reduction of the traffic on the network. Despite these extension supplied by DHTML over HTML, XML offers characteristics quite more powerful, allowing to deal with structure and semantics of the information, and to de-couple the form of the visualisation from the server where the data come from.

5.2 XML – STEP technology

XML is primarily a message-centred technology. It is adapted to the exchange of information, allowing to deliver various formats of documents presentation (or messages structuring), while the STEP standard, especially through EXPRESS, offers a technology for the description and data representation of product information including their semantics, and a lot of normalised models. Regarding pure information exchange as considered in STEP within Part21 (SPF), some of the benefits of XML are the following:

- SPF requires the exchange of a complete STEP model, where XML enables for exchanging parts of information whenever only this is required;
- Transmission of information to applications can be done over the Internet (e.g. using HTTP), which allows to cross firewalls, that is obviously not directly the case with SPF;
- A lot of XML parsing tools already exist, some of them being free software, while the parsing of SPF files calls for specific developed parsers.

On the other hand, XML messages do not vehicle semantics for their contents. This is also the case for SPF files, which need to be shipped with their corresponding EXPRESS schema (at least the first time). However, a fundamental issue with XML is that it does not allow to specify a given semantics for each tag used to structure documents. For instance, there is no automatic way in XML to specify a common semantics for the tags <*facility*> and <*equipment*> manipulated in distinct documents generated from separate applications. With XSL, it is simpler to create conversion between tags so as to convert a document using one DTD in another document using another DTD, provided that it is possible to relate tags between themselves. But in order to avoid such conversions (that could reveal costly at runtime), the best is clearly to agree on a common DTD: the objective is then to guarantee a common understanding of the message structure before any operation. Several groups in industrial vertical sectors are currently investigating or already defining such agreements. Then, two approaches can be envisaged in large-scale projects:

- exchange between a lot of applications through common agreed message formats (i.e. a single DTD);

- exchange and XSL-based transformation whenever required between few applications, and preferably when the frequency of exchange is low.

This states that STEP and XML clearly must be viewed as complementary technologies, and it appears valuable to specify and develop gateways allowing to exploit STEP (EXPRESS-based) information through dedicated technologies (like XML) dealing with heterogeneous and adaptive environments. In the AEC field, different work recently emerged, as AEC-XML (AECXML 2000), bcXML (bcXML 2000), a starting initiative within the IAI (named IAI-XML), and the work undertaken within the ISO STEP for an XML representation of EXPRESS-driven data (ISOe 1999). These works tend to define DTDs and XML-based mechanisms for STEP-based information exchanges. The future is probably for STEP to concentrate on semantics and structuring of product information (i.e. data models for the whole product life cycle), while taking benefits of technological developments around message brokering and the Internet, especially XML, thus providing a logical separation between semantics on one hand, data access, transfer and presentation on the other.

5.3 XML – CORBA and MOM

This section essentially deals with interoperability issues, where today information systems are heterogeneous (incorporating ERP tools, groupware tools, client/server applications, etc.), have to be opened (Extranets, Internet), and will be more and more integrated with enterprise workflow and processes within the VE. A key challenge indeed is not only internal communication between the enterprise systems, but the ability to easily and quickly exchange information with the outside. This is even more crucial in dynamic virtual environments as in the Construction industry, where partners are almost never the same from one project to another. These partners make use of either "fat" clients with large applications (along with their own data models, databases, etc.) or "light" clients (with at least a Web browser, and only small applications on the desktop), that will have seamless access to core business processes through Web sites. We hereunder review the criteria identified in section 3.

1st point: management of messages.

XML provides with *explicit messaging* between applications. An XML document can be viewed as a message that has to be parsed in order to identify the various elements of the document, then specific actions can be realised on elements, relying on a tree-based model (DOM) or an event-stream model (SAX). On the other hand, CORBA deals with *implicit messaging*: the messaging is transparent and managed at level of the ORB, it is automatically analysed in order to deliver the final result, under an object-oriented form that can be directly used by the target application. This could be interpreted as simplifying the organisation of the VE infrastructure, but CORBA does not stipulate anything about the internal mechanism used for communication: in general (except if using IIOP, see next point), this messaging is dedicated to the ORB in use, and only applications that have been explicitly connected to this ORB can inter-operate, through internal ORB's messages. This point reveals to be a strong constraint when considering access to applications in Extranets or over the Internet.

2nd point: protocol issues.

The XML documents can be exchanged no matter the underlying protocol. Indeed, XML is of higher level, providing with a language which is not a protocol, but can be used to define any protocol. On the other hand, CORBA is defined with the IIOP protocol. IIOP is a standardised protocol for exchanging messages independently of ORBs: this is a key point, as this allows to plug a new application theoretically without having an explicit knowledge of

the internal ORB messaging mechanism. Moreover, IIOP messages can be exchanged over the Internet, provided that potential firewalls that can be met on the way are able to deal with IIOP proxies (and not only with HTTP streams). Nevertheless, even in that case, management of IIOP messages is intrusive at level of applications codes as it need to be explicit, i.e. the application must say if it deals with IIOP messages, and consequently does not really fit with environments that should accommodate frequent changes of applications.

3rd point: shipping of large set of data *versus* small set of data

The construction sector often requires the shipping of large volumes of data (e.g. for a CAD tool or a full IFC-based application). XML is better adapted to manipulate such volumes, while probably CORBA best fits with application accessing few remote data. IIOP is not configured for dealing with large messages, it has been developed following a method-based access model, and deals with messages containing methods calls with their typed parameters.

4th point : Data duplication and consistency control (i.e. need for check-in/check-out mechanisms).

The main issue raised here is to know if checking/validating the type of information is required at the source or the target application, and is to some extent related to a quality of service. Here, distributed object computing technologies, to which CORBA belongs, ensure a deep type checking that is fully object-oriented, and reveal to be more effective. If the receiving application (the one dealing with its local proxy object and that makes the request) needs to check or validate the type of information it gets, CORBA ensures with IDL the well typing of the returned result, at least at level of all the types as manipulated in IDL, i.e. basic types (integer, string, etc.) as well as typed object structures (indeed, any object whose class has an IDL structure mapping). Moreover, CORBA, as the information is accessed/modified remotely, enables to deal with a unique point of conservation of valid information (on the server), consequently leading to what can be called a *permanent data coherency*. Additionally, this can be augmented with powerful transactional mechanisms.

On the other hand, XML only deals with strings, as DTDs define a hierarchical structure for composition of strings. Consequently, data serialisation with XML files leads to provide the target application with string-based information, where the semantics (as defined in the source application) of the serialised data is lost. Indeed, the only point is that a DTD provides with a tree structure that is a basis to then potentially rebuild a set of typed object-oriented structures that can partially mirror on the target application side a similar semantics for data coming from the source application side. Moreover, shipping the information through XML messages from one application to another in order to really operate on this information locally on the target application means that on the way back, there is a need to check and accept modifications on large volumes of data, i.e. *to manage potential data inconsistencies*.

5th point: asynchronous mode of Building Construction process models

Most of the building construction process models are based on an asynchronous mode for communication, i.e. the availability of the addressee where a request is emitted is not mandatory, at least in real time. From a more general point of view, it is highly preferable to reduce the needs for synchronisation between applications in order to improve their availability and their performance, which seems applicable in the construction sector. Whilst CORBA (as well as DCOM or HTTP) deals with synchronous communication types, XML and MOM-like middleware appear as essential technologies to tackle applications integration and workflow-based co-operation issues within the Construction industry.

An initial conclusion is that presumably CORBA (or DCOM) technologies are well adapted to local (even large) Intranets, but not easily customisable over the Internet, and probably will reveal oftentimes as an infrastructure offering too powerful mechanisms for client desktops (e.g. CORBA is not required so as to only deal with presentation concerns, where HTML and DHTML fit well). Java applets are an alternative, but have not been considered as fully satisfactory so far, possibly due to some trouble with code portability. XML then emerges as a solution for conveying data in a structured way, still relying on HTTP at level of transport for accessing any client platform. A future step then could be to access distributed objects from a Web browser using XML for sending requests and receiving answers, as this is currently investigated like XML-RPC (XML Remote Procedure Call), for instance.

6. CONCLLLUSION

This paper has introduced a survey of some Internet-oriented technologies for document- and message-driven applications in construction open dynamic virtual environments, as currently undertaken within the OSMOS project. It has highlighted some of the advantages of the XML language, supplying cost-effectiveness for implementing Internet and distributed applications based on XML software tools and components (both on client and server sides), and showed how XML seems to offer promising perspectives, at least in terms of interface with the outside of the information system, especially the Internet. Regarding the software market, it is worth noticing that a lot of actors (IBM, Oracle, Sun, etc.) integrate the parsing and generation of XML documents within their platforms. Numerous application servers use the XML format for sending information, and databases integrate as well an XML parser. In addition, freeware tools are accessible through the Web. Thus, as soon as an application is powered with XML, this should lead to minimal (or at least reduced) effort to exchange data.

The OSMOS consortium is taking an iterative and incremental approach to the development and deployment of the tools and infrastructure supporting the construction virtual enterprise. Three iterations will be conducted leading to the setting-up of teamwork service provider prototypes. The first iteration will implement techniques that enable teams to share and exchange unstructured documents (ranging from text-based to CAD drawings) using HTTP-based protocols over Intranets as well as the Internet. The second iteration will tackle XML-based structured documents, exchanged through HTTP, and will adhere to the approach described in the paper in relation to accessing product-based data representations (STEP, IFCs, etc.). Finally, the last iteration will validate the techniques used in the two previous ones, and will explore object-oriented communications on top of CORBA-based middleware.

However, the construction business processes need to be refined and re-adapted in order to deliver the expected benefits of the construction VE. In that respect, new models for business processes, working methods, organisation, contracts, and legal responsibilities related to computer support for co-operative work in a VE are being investigated and developed. Finally, the authors would like to thank the OSMOS consortium members for their contribution to the research, and would like to acknowledge the financial support of the European Commission.

REFERENCES

AECXML (2000). The Architecture, Engineering and Construction XML, Web site: http://www.aecxml.org/

BcXML (2000). The building construction XML, Web site: www.eConstruct.org

IAI (1997). Industry Foundation Classes, Version 1.5. Web site: http://iaiweb.lbl.gov/

ISOa (1994) Industrial automation systems and integration - Product data representation and exchange Part 1. Overview and fundamental principles. 1994. N° ISO/IS 10303-1.

ISOb (1994) Industrial automation systems and integration - Product data representation and exchange Part 11. Description methods: the EXPRESS language reference manual. 1994. N° ISO/IS 10303-11.

ISOc (1994) Industrial automation systems and integration - Product data representation and exchange Part 21. Implementation methods: Clear text encoding of the exchange structure. 1994. N° ISO/IS 10303-21.

ISOd (1995) Industrial automation systems and integration - Product data representation and exchange Part 22. Standard Data Access Interface. 1995. N° ISO/DIS 10303-22.

ISOe (1999) Industrial automation systems and integration - Product data representation and exchange Part 28. Implementation methods: XML representation of EXPRESS-driven data. 1999. N° ISO/WD 10303-28 (ISO TC184/SC4/WG10 N285).

MOF (1999). The Meta Object Facility Specification, OMG Document Version 1.3 RTF, 27 September 1999, Web site: http://www.omg.org/

Radeke E. et al. (1999). GENIAL: Final report, ESPRIT 22 28, 51 p. Web site: http://wwwgen.uni-paderborn.de/GENIAL/index.html

RDF (1999), The Resource Description Framework, Web site: http://www.w3c.org/RDF/.

Rezgui Y. et al. (1999). CONDOR: Final Report, ESPRIT 23 104, University of Salford.

SGML (1986). *Information-processing -- Text and office system* -- Standard Generalized Markup Language (SGML). - ISO 8879 document, 1986.

Stephens J. et al. (1999). VEGA Public Final Report, PFR-01 report, ESPRIT 20408 VEGA. Web site: http://cic.sop.cstb.fr/ilc/ecprojec/vega/home.htm