XML Applications as Middleware and Metadata for Workflow in Print and Cross-Media Production

Simo Karttunen and Heikki Nikulin*

Keywords: WWW, publishing, printing, digital printing, workflow, extranet, HTML,XML, RDF, DOM, DOI, cross-media, format, content, metadata

Abstract

The content information for print or web documents never comes from one source but from various systems, often from web sites. Particularly reusable content elements may be coded as XML documents, recommended by the W3 Consortium. Like web publishing, cross-media publishers need universal middleware between content providers, content processing and design (prepress or web authoring), and print or web delivery. Print media are mostly output through offset and gravure technologies, or through digital printing lines (e.g. on-demand, short-run and variable data). Mass printing requires press frontend servers in which the page files are assembled into press-sheet-wide impofiles, prepared, checked, proofed and trapped for film, plate or direct-to-press output. File preparation for print is largely manual session-to-session batch processing with large files. The XML (extensible markup language) and the RDF (resource definition framework) offer universal object formats, extended linking, resource marshalling, logical and dynamic documents for workflow automation, and for content and document data management. Here, we report on the progress of the XML applications - both existing and concepts - for print and cross-media publishing, and for related print brokering and production. We analyse the performance and integration between XML and other solutions,. e.g. native formats, PDF , SGML and HTML. We propose how to integrate and connect systems to the different native formats, existing DTP systems and applications, press frontends, workflow and middleware solutions, and how to use XML and related standards to build intelligent web applications for publishing.

* **Research staff of VTT Information Technology**
PB 1204, 02044 VTT, Espoo, Finland
http://www.vtt.fi/tte

Introduction

A new wave of changes is driving content creation, publishing and cross-media production to use more standard routines and management. In web documents the links are used to refer to other sources related to the topic of the document. Print media has no real linking feature. A printed document is a series of images or pages in an ordered number of copies which may be identical or similar with variable data fields. Most printed products need not only document design and printing but also finishing, binding and cover design. Designing printed products is a complicated process with dimensions other than those of planning web portals and large web sites.

Printing production and the production workflows before the presses (prepress) are digitally controlled. The servers used in this production network can be connected to intranets or extranets. Extranets are private and secured networks for a group of companies sharing a workflow from content to printed products. In such networked print media, the workflow should be a **web application** for the obvious reason of openness and flexibility. The W3-recommended XML technologies open up new opportunities for the printing services to be developed into web applications.

Large jobs have to be printed mostly in **one location** where the presses are. There the press frontend workflow and their RIPs (raster image processors) prepare the final and checked impofiles for the film or plate output of the respective press sheets shortly before printing.

For this purpose there are proprietary workflow solutions [1] which collect, proof, impose, rasterize, screen and finally record the press sheets for films or plates. These workflows use common native and standard file formats, e.g. TIFF, PostScript and PDF, which carry the job files to the output. Conversions and filtering are needed in many cases.

This paper continues a series of papers on the workflow concepts in cross-media companies [2, 3, 4, 5, 6] with a study of the **XML applications** in publishing. We have already published [2] the concept of a Universal Print Application Server (UPAS). It is a web and XML application in which the XML document is used to link and collect the content information before sending it to output workflow systems. This XML-based workflow complements - rather than replaces - the common

PDF, or other similar press frontend systems [1], and helps them to become web applications.

Concepts of Media Content Handling

During its over fifteen-year existence, Adobe's PostScript has provided a *de facto* standard for coding documents for output independently of the application (text, graphics, images or page design). This format has been interpreted by the raster image processors (RIPs) of the leading vendors of film and platesetters. Later, the portable document, Adobe's PDF upheld this tradition with a more powerful feature set. The PDF includes workflow features and is clearly more oriented on the document, i.e. beyond-the-page level. The latest PDF version 1.3 improves workflow, metadata and colour management features [23].

Complex linking and large web sites require more advanced documents. In addition to portability the web needs logical, distributed and dynamic documents. The Apple OpenDoc, the original CORBA container, is a compound document with many of these properties. Similar neutral document approaches were introduced in the Java component families (JavaBeans and Enterprise Java Beans, EJBs). The Java CORBA environment provides a high degree of interoperability between servers, applications and documents and many other benefits, as pointed out by Orfali [9].

The price for this is that it also requires another concept of connectivity, i.e. the standard ORB and the IIOP (Internet InterORB protocol). Gradually, but not quickly, ORB products will find markets on the new basic OS platforms. In this paper, we only mention the **distributed component** approach, Java CORBA, reported by Orfali et al [9], and mainly discuss the opportunities of XML. The distributed components and XML are two standard approaches which complement one another .

The W3-recommended XML is a **metalanguage** which can be used to structure documents (like SGML and HTML do), to link and to style them (like HTML does) and to define documents, web sites, and practically any resources addressed on the web. The way this is done by designing applications, writing document definitions (DTDs), tagging documents or designing link elements has become a new software industry for web applications [7, 8, 10, 11, 22]. The last chapter refers to

several systems which are already available. XML and related technologies are also used to improve the print media workflow. On the other hand, XML is so new that many specifications are at the draft stage, and early applications have to be rewritten if they are based on drafts with later changes.

The benefits of XML have been analysed properly in many publications [7, 8, 18, 22]. In fact, there is also a lot of confusion in the **implementation of XML**:

• DTD, the document type definition, originally a concept of SGML, is a data model [7] which helps to **structure** the content and declares the internal and external resources for later XML parsing.

• DOM, the document object model [13], a platform and language neutral interface that allows programs and scripts to dynamically access and update the **content, structure and style** of documents.

• RDF, the **metadata** architecture and resource definition framework [12] is a method for including semantics (ratings, meanings, purposes) in the XML/RDF metadata of web applications.

DOM and RDF are briefly described in the Appendices 1 and 2 of this paper, to show what they add to the basic XML DTD applications. We recommend the W3 Consortium's information links to more interested readers and software developers.

Also vertical industrial approaches are built in XML applications, e.g. by using specific DTDs for their typical information structures. Cover's lists under www.sil.org numeous XML books [7, 8] and white papers [18, 22] tell about various approaches in applications ranging from chemical, medical, and vector graphics, to publishing, multimedia, virtual reality and I-commerce. In fact, XML technologies are steps which lead to web and media **automation** in which the web content is organized, tagged and identified for subsequent exchange of data between standard web applications and content stores.

Some specifications have been developed for workflows and for specific data exchange purposes [17, 18, 20, 22]. The XML technologies are object-oriented methods to manage all information and data, and to enable web and **document centric** processing in future media. Against this

background it is relatively easy and safe to promote this technology among all media professionals.

Process Definitions

XML is more than just a markup language, such as the SGML. We must have some system design and tools to make the XML work for our media products and production processes. In the **print workflow**, media elements pass through several interactive **sessions** and **batch processes** at workstations and servers. A connectivity concept is needed to move or to pick up elements of information, and to publish, and deliver data after the style, design, addressing and packaging processes.

It is clear that XML is mainly designed for web publishing, information services and I-commerce. There is, however, plenty of ready software also for the common processes of **cross-media** production and printing, as mentioned in this review.

To **build XML applications,** we map out some of the familiar processes to various solutions and standards which are available in the XML system industry. We do that here by setting a table as follows:

Connectivity	XML´s linking tools allow far more complete access and data exchange features, than the simple HTML links, to be packed into web applications. XML can be used as a web **middle ware** between XML-aware web servers. The traffic between media content providers, publishers and printers would require services on secured **extranets** and on the Internet.
Job structures	There is a natural possibility to use the XML DTDs to describe the logical **structures** of elements, pages and documents, and also as a workflow and data exchange tool between applications and servers. Files may contain logical elements of XML for linking, DB data links or XML documents with metadata and large native files in specific combinations.

Workflow	By linking up **content** - from the web sites or extranets - it is possible to produce XML which is self-describing and machine-readable. To become a processable page file, the XML document must be styled and printed.
Style	The style mechanisms supported by XML are based on XSL [7, 8] but other approaches, e.g. the older CSS, are not excluded from XML applications. See Appendix 1 for DOM , the document object model, and ref [15].
Tracking	A new IFRA project [20] to use XML for the IFRA Message Format (IMF) has been started. A product model (newspaper) and a state model may be included in the IMF objects, as originally done in the IFRAtrack 2.0.
Business processes	XML technologies are object-oriented methods for creating data exchange between applications, including business processes. Many vendors develop XML-based modelling and business automation for I-commerce[14, 18, 22], Netscape talks about servers for process automation in Net Economy [22]. XML middleware may be used both for production and business processes.
Media junction	XML applications are natural media junctions for web and print workflows and they support variable information, database linking and on-demand.
Archiving	Data stores in any document size, state of readiness, and on any hierarchical level is easy to arrange by using XML and DTD or other more advanced tools. Controlled access and other security features can be configured in an extra net. Future content management systems use XML technologies.

Content rating	RDF is a metadata architecture that uses XML syntax and also describes the resource semantics on the web. RDF makes it possible to rate the content and to exchange semantically rich metadata (Appendix 2).
Collections	RDF supports rich metadata which enables more intelligent and automatic search from web collections by providing a logical data model for content.
Push, privacy	RDF is the basis for RDF applications, such as PICS 2.0, a content-rating approach, P3P a privacy specification and UCP, user centric push. Many other future specifications will be using the power of RDF applications.
Identifiers	The identification of content elements and documents is an issue which must be solved in every domain. Unique identifiers are developed in the W3C under the Working Group XML Namespaces [19] as unique web names (i.e. URLs).
DOI	We also have **DOI**, the Digital Object Identifier (see www.doi.org), from the International Digital Object Identifier Foundation (IDF), which is a content standard useful for including metadata and resolution to multiple URLs. Doi is not only unique, like the URLs, but also persistent [21].

Process definition and data exchange between distributed applications and documents is a basic feature of XML since it was originally designed to define data types, entities, hierarchies and other object-related modelling aspects. To describe and organize web resources in a machine-understandable way, RDF incorporates semantic descriptions into the attributes of the RDF metadata. The RDF could also be used as a mediabank approach with highly structured services, for example, on business-to-business extranets.

Readers should remember that we are dealing here with **media production** in a distributed private workflow - not in a public domain, such as the Internet. The WWW may be a very useful source of information in many cases. In commercial media the **process owner** (e.g. a publisher or an ad agency) needs material to which he has legal access and user rights (e.g. copyrights). For his publishing process he needs partners such as photographers, repro houses, image banks, writers and news agencies with whom the legal issues of ownership are governed by contracts.

Then the only viable network for the virtual production **workflow**, such as the one proposed in our UPAS model, is the extranet, i.e. a properly secured, virtual private network (see the next chapter). If the printing plant - a contractor commissioned by the process owner - needs to link resources, e.g. ready job pages from the previous workflow partners' archives, he must have the access and the permission for the reuse, and then do so on the joint **secured extranet.**

XML as a Middleware and Server Tool

There are many implementation alternatives for XML technologies in publishing and I-commerce based on existing innovations [15, 16, 17, 18, 20, 21, 22]. This results from the metadata features and other XML benefits listed in several references [6, 7, 8, 10, 12, 13, 18]. In a recent discussion [18] people debate whether to accept XML as a **middleware** and not only as a document management and workflow tool. Even though the middleware features are an undefined set of things from connectivity to workflow management the XML middleware is a more requiring goal.

The need of such middleware was pointed out by us [6], by Bitstream [16] and Object Design [18], especially **for the production**. The IFRAtrack activity to convert IMF into XML is a middleware solution specialised in the management of newspaper production [20], which needs almost real-time tracking of the processes. XML and RDF support rich metadata and some of the further developments, like DOM and RDF, give even more support to, not only describe, move and organize contents, but also to take care of the business, i.e. prepare the orders, estimate the costs, to invoice, to register customers, to ontrol sales, to administrate personnel,

to control the resources, to sort out and support various physical and abstract assets, not only the contents.

Middleware has to offer enterprise, process and data models as **object services,** such as XML tools. This has been pointed out by many general integrators and middleware vendors, such as BEA, CA, HP, IBM, IBM/Lotus, Netscape, Object Design, Oracle, PeopleSoft and SAP. They and others have modelled general business, and must use web applications **on a large scale**. The XML is a logical choice for publishing business middleware. Recently [22] the whole networked business is presented as a continuous process to **create new services**. The new role of the enterprise service providers - ESPs, and not only ISPs - will be to collaborate via the web and control the exhange all relevant content and data.

But the question remains: How to implement XML ? It must be answered by listing additional integration elemnts and tools. In some cases we apply only the XML document and the XLink. In some other applications the use of XML DTD, style mechanisms, linking and RDF iis needed. More business planning and administration features will be needed in the future. Then the object model (DOM), XML Data and possibly Namespaces, ICE, UCP, XMI and other XML-based concepts [17, 18, 22] may be needed to build scalable web applications based on each publisher´s business and product specifications. At this early stage of affairs it is difficult to forecast the most succesful XML systems. The above discussion points out the principal developments.

UPAS

Recommended standards and drafts are the only viable way and our primary concern. We follow the work of XML, W3C, I-commerce and Web Economy *ad hoc* **Working Groups.** Our own **UPAS server concept** provided the means to test our ideas and to define demo systems. In this paper, we have defined a UPAS demo (see Appendix 3) to show how to run jobs through an **XML middleware** into a printing workflow. The server concept needs further development and we are ready to proceed.

The UPAS architecture is shown in **Fig1.** It allows any content to be linked to job documents. UPAS is a web and XML application in where the XML document is used to collect job content information **before**

sending it for output. If XML DTDs are used in the page content we have to style the pages and "print" the physical page files, i.e. the lefthand workflow column in **Fig 1**.

Digital printing and press frontend workflows are often based on PDF [1, 23]. The UPAS outputs PDF stacks for output processes, i.e. to press frontends, as they are today configured [1] in many vendors´ proprietary workflow systems, and does it in dimensions of press sheets and jobs. UPAS also uses ready PDF pages as the input or as digital files from print media – the right hand column in **Fig 1**.

Showing in detail how the workflow works is a long story. The UPAS and the respective **XML middleware** provide scalability, configurability and product flexibility. Such systems have to be customized for each company on the business extranet. We can already list some typical subprocesses and tools for the partnering actors of the total workflow:

Process owner
- Publishers and ad agencies configure and plan product families
- XML DTD is a framework for product structure descriptions
- Process owners provide and update workflow and scheduling data
- Tracking must supply information about the progress of the jobs
- In some cases, the final output-ready document is produced here
- Needs to check intermediate, proofed and final page files are vital

Repro house
- Camera and scanner systems must capture a continuous flow of images
- The content of images must be XML-tagged, linked and organized
- Later on the metadata could include semantics (i.e. as RDF)
- Later assemblies of images in the final pages must be visualised

Writers
- Normal office software packages and text processing applications
- Pages and jobwide files require systematic storing (XML) or not
- XML and DTDs could help the archiving and reuse of content
- To handle intellectual property rights requires special software
- It is necessary to exchange workflow data with other parties

Printing output

- At this stage the jobs are output as press sheets through press frontends
- XML documents are jobwide and hierarchically smaller files
- The process owner needs visual and tracking information of jobs
- Another DTD or a similar metadata solution defines the workflow
- At this stage PostScript, PDF and XML documents must interact

Up to this point the UPAS works as a **production middleware** based on XML linking and metadata. As shown in **Fig 1**, the portable document, PDF is complemented by the XML application, and no major changes are needed in the existing systems.

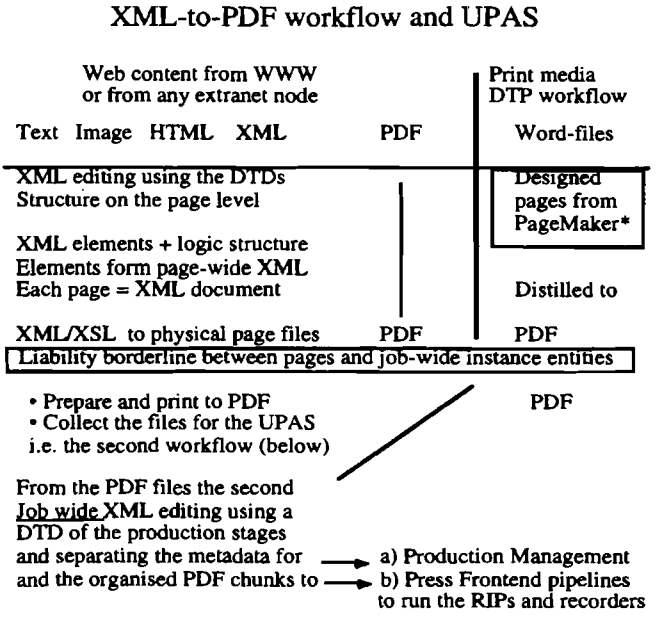

Fig 1 The UPAS architecture shows how the content from the web is input in job wide XML applications to hold advanced metadata and to interact with existing software and formats, e.g. the PDF.

XML software is available

XML is already a part of commercial web publishing software. Here we mention some of these products and point out their possibilities as workflow tools. B eginning from the basic editor/parser software mentioned in the [7 and 8] - often coming from the office, DTP and SGML vendors - we may expect more specific, advanced XML applications. Some of them offer a covering workflow for publishing applications, e.g. the PageFlex and NuDoc of Bitstream [16].

Other vendors offer general **middleware** approaches, like IBM, Netscape and Object Design [14, 18, 22]. As an indication of consensus, the W3 Consortium promotes the XML as a **data exchange** format [17] for the web applications in the interface tier (clients) as well as in the middle tier, e.g. web servers and application servers. There are also many XML-based specifications for bridging and specific data exchange purposes, as we have shown in ref [17].

There is a natural environment or platform for XML web applications, i.e. the **web server**. Therefore vendors who concentrate on web site and linking management are involved [10, 11, 12, 14, 16, 22] irrespective of whether they are also interested in content management. In a printing facility, a web server running under the UPAS [2] or similar XML applications could be the last point of document management before the final press frontend workflow with RIPs and plate and film recorders [1].

Another concept is **the application server**, for which both XML and Java CORBA offer relevant basic services based on object frameworks. Application servers add more web-based services to the enterprise computing.Vendors, such as Apple (MacOS X Server) , Bluestone (Sapphire Web 5.1), IBM (e.g. WebSphere) and IBM/Lotus, IONA (Orbix OTM 1.0), Microsoft (IIS 4.0 and the Transaction Server), Netscape (Application Server and Process Manager), Oracle (8i web DB and Application Server 4.0) and Sybase (Enterprise Application Server) have their special application server products. There is still some uncertainty about the object model(s) which will be used (i.e. COM+, SOM, DOM) in application servers, which are less standard than web servers. To buy one is a problem, even though big savings could be achieved with better application deployment and management.

In these environments, the services offered by XML technologies allow advanced metadata and system/object/application **interoperability.**

Normally the services offered by the application servers include many business actitivities such as scalability, application deployment, application management, load balancing, process automation, directory management, CORBA and ORB deployment, Java components (JavaBeans, EJBs, servlets, applets), and XML support in many cases.

XML-aware basic web clients (browsers) and web servers will make a rapid breakthrough in the web application markets. XML linking and data modelling will be used to complement HTML and PDF. The web users will use XML as an improved data exchange tool and configure business middleware in publishing and media production. Every system vendor and efficient user should find his own niche in these new web appliocation markets.

Conclusions

We have reviewed the new approaches to use XML as a general middleware and metadata tool for publishing applications. The W3 Consortium pushes XML and related technologies as standards for describing resources and contents on the rapidly expanding web.

The concept of static document requires changes to generate more advanced web applications. This means that document-centric processing will be the dominating way to operate the web and other publishing domains. XML documents with their extended linking, style mechanisms and document definitions (DTDs) are the basic scalable web applications. Most essential XML specifications reach the recommendation status in 1999. The vendors who also participate actively in the W3 developments offer their new XML software as more or less ready versions.

There is already a lot of XML software which we have reviewed briefly. XML is a metadata concept which allows clever and complementary solutions with the existing resources, such as proprietary desktop publishing packages, proprietary content and database resources. In some cases, XML middleware will be used as a method for transforming or bridging legacy systems into web applications.

In our paper, we also give a system definition, called UPAS, with many XML benefits. The web application UPAS complements the portable document, PDF, in print and cross-media production.

References

1. **David Zwang and Caren Eliezer** "Print Production Workflow, Part II: A View into the Competing Approaches". Seybold Report on Publishing Systems Vol. 27, 23 March 1998, pp. 1-23. This report compares nine commercial press frontend systems.

2. **Simo Karttunen and Heikki Nikulin** "Management of Data, Jobs and Workflow in Networked Media Production", Paper for the ICPS '98 Conference, Track 2, Antwerp, September 1998. Proceedings published by the ICPS in September 1998.

3. **Simo Karttunen** "Production Data in Media and Press Frontend Systems: Capture, Formats and Database Methods. Paper for the European Optical Society (coorganised with the SPIE and IS&T) **EurOpto** Conference, Berlin, Germany, 7-11 October 1996.

4. **Simo Karttunen, Caj Södergård, Heikki Nikulin and Tomonori Yuasa** "Process Timing and Colour Management Data Exchange between Intranet Servers in Media Production". Journal of Prepress and Printing Technology, **Vol. 1** (1997), No. 1, September 1997, pp. 10-21.

5. **Simo Karttunen and Heikki Nikulin** "Generic Models and Document Tools for Media Production and Data Management in Networked Production Servers", TAGA '97 Conference, Quebec City, Canada, 4-7 May 1997.

6. **Simo Karttunen and Heikki Nikulin** "Tracking Jobs from Servers and Workflow Applications in Media Production" Paper for the TAGA '98 Conference, Chicago, Illinois, USA, 26-29 April 1998.

7. **Richard Light** "Presenting XML", Samsnet, Indianapolis 1997. A basic textbook on XML.
Other XML sources:
- **W3C** Extensible Markup Language (XML) 1.0 , e.g. on www.w3.org , or
- **Ed Tittel** "What's the Point of XML?" A white paper on the basic features of the standard XML.
- **Robin Cover** at www.sil.org/sgml/xml
- The **link list** of W3 Technical Reports and Publications at www.w3.org/TR gives an overview of the recommendations, working drafts and notes of the web development task forces related to XML.

8. **Steven Holzner** "XML Complete" McGraw Hill, USA 1998 a textbook with CD-ROM sample XML documents, templates and a XML parser.

9. **Robert Orfali and Dan Harkey** "Client/Server Programming with Java and CORBA", John Wiley & Sons Inc., New York, 1997, 600 pages, www.omg.org .

10. Liora Alschuler and Mark Walter "Highlights from **GCA's XML '98**. A Trip Report" Seybold Report on Internet Publishing, Vol. 3 (1998), Nr 4, December 1998 , p. 13.

11. Eve Maler and Steven DeRose "XML Linking Language **(XLink)** and XML Pointer Language (XPoint)" W3C Draft Recommendations of 3 March 1999, both of them at www.w3.org/TR
• **Eve Maler** "XLink and XPointer: Overview" at www.oasis-open.org. Briefly about the coming **XML linking specifications:** Estimated publication of the recommendations in the second half of 1999.

12. Ora Lassila and Ralph R. Swick "Resource Description Framework **(RDF)** : Model and Syntax Specification" a W3C Recommendation, 22 February 1999, at http://www.w3.org/TR and at www.w3.org/RDF .
• **Tim Berners-Lee** "Metadata Architecture: Documents, Metadata and Links", a comment paper for the W3C. See also **IBM Alphaworks** "RDF for XML", a Java implementation (RDF processor) of the RDF Specification. www.alphaworks.ibm.com .
• VTT Information Technology is represented in the RDF Working Group of the W3 Consortium.

13. Robin Cover "The SGML/XML Web Page: W3C Document Object Model **(DOM)**, January 1999 at www.oasis-open.org/cover/dom.html
• **Vidur Apparao et al** " Document Object Model **(DOM)** Level 1", W3C Recommendation, Oct 98, at www.w3.org/TR

14. Mark Johnson "XML JavaBeans, Part 1: Make JavaBeans mobile and interoperable with XML", at www.javaworld.com/jw-02-1999/jw-02-beans.html Describes how to make Javabeans mobile in the XML by representing them as XML documents and shows how these two object technologies complement each other. **Mark Johnson** "XTech 99: Java and the XML Wave" reports about the use of Java to help XML technologies to break through and in general the news of XTech '99. www.javaworld.com/jw-04-1999

15. James Clark and Stephen Deach "Extensible Stylesheet Language **(XSL)**", W3C Working Draft , 16 December 1998 and XSL is also reviewed by Holzner in [8] Chapter 9.
• **Norman Walsh** " **XSL** Requirements Summary, W3C Working Draft , 11 May 1998
• **James Clark** "Associating Stylesheets with XML documents" W3C Proposed Recommendation
14 January 1999, all three specifications at www.w3.org/TR , also at www.w3.org/Style/XSL .

16. Bitsream: Three XML-based applications: NuDoc™ and PageFlex™" and MediaBank™, the last mentioned was sold to the web publishing vendor Inso, see www.inso.com. The PageFlex is an integrated software for the XML content, template-based style (with a proprietary non-standard style format, TSL) and XML formatted layout and output in PostScript and PDF. NuDoc™ is an advanced document composition engine. See the product data at www.bitstream.com.

17. XMI-based web integration and interoperability specifications are many. We only list some of them with short comments. In fact DOM and RDF, mentioned above, and described in the Appendices, also belong to this category:

- **ICE**, The Information Content Exchange, this protocol uses XML to automate the exchange of contents between publishing or business partners
- **DCD** for XML, the Document Content Description, a NOTE-level schema approach, using XML Data and RDF to include more features than the DTDs alone
- **WIDL**, the Web Interface Definition Language is an XML application for automating web access using DOM to map data elements
- **CIM**, the Common Information Model is a set of classes and instances (schema) in XML DTD for the web-based enterprise management concept (WBEM) of the Desktop Mangement Task Force (DMTF)
- **XMI and X Query** are new attempts to work on the integration of object models and query technologies with the XML and RDF applications.

18. Anon "Enterprise Applications using XML : Understanding the Business Benefits" White paper on the Object Design. www.objectdesign.com

19. Tim Bray, Dave Hollander and Andrew Layman "Namespaces in XML" a W3C Recommendation, 14 January 1999, at www.w3.org.TR

20. Stig Nordqvist and Fredrik Fällström, Private information about the new work started for converting the IFRA Message Format (IMF) - the tracker object of the IFRAtrack 2.0 - into XML. This means a more open policy, to turn the propriety PMS (production management systems) to web-enabled XML applications. There are already several vendors using IFRAtrack 2.0 (1997) and IMF. www.gt.kth.se and www.mwm.se/IFRAtrack .

21. DOI: the Digital Object Identifier, by the DOI Foundation, IDF, presented at www.doi.org.

22. Netscape "Web-Based Process Automation for the Net Economy", at http://home.netscape.com/processmanager/v1.0/whitepaper.html

23. Mark Walter "Adobe Gives Acrobat, PDF a Major Facelift" and "Acrobat 4: Adobe Bid to Make It More Than Just a Viewer", Seybold Report on Internet Publishing, Vol. 3 , March 1999, p.1-11.

Appendix 1

Document Object Model
DOM
Source: Cover [13]

The main points of the Cover's presentation
of DOM Jan 1999 [13] at www.oasis-open.org

• The Document Object Model of W3 Consortium is a platform and language neutral interface that allows programs and scripts to dynamically access and update the **content, structure and style** of documents.

• The document can thus be processed further and the results of that processing can be incorporated back into the presented page.

• The goal of the DOM group is to define a programmatic interface for XML and HTML. The DOM is separated into three parts: **Core, HTML, and XML**.The Core DOM provides a low-level set of objects that can represent any structured document. While this interface is by itself capable of representing any HTML or XML document, the core interface is a compact and minimal design for manipulating the document's contents.

• The HTML and XML specifications provide additional, higher-level interfaces which are used with the core specification to provide a more convenient view into the document.

• **DOM Working Group :** ArborText, IBM, Inso EPS, JavaSoft, Microsoft, Netscape, Novell, Object Management Group (OMG.org), SoftQuad, Sun Microsystems, Texcel, and W3.org.

• DOM provides a standard set of **objects** to represent HTML and XML documents, a standard object model, and a standard interface for accessing and manipulating them.

• Vendors can support the DOM as an **interface to their proprietary data structures** and APIs, and content authors can write to the standard

DOM interfaces rather than to the product-specific APIs, thus increasing **interoperability** on the web.

• In early January 1999, the W3C Document Object Model (DOM) Working Group released a new **Working Draft** specification for the DOM Level 2, Version 1.0 (WD-DOM-Level-2-19981228). The document is available in Postscript, PDF, plain text, HTML, and XML formats.

• Level 2 adds interfaces for a Cascading Style Sheets object model (**CSS,** see below), an event model, and a query interface. The now released Document Object Model Level 2 does not have all these interfaces. It contains interfaces for the Cascading Style Sheets object model, the Range object model, filters and iterators, and the Events object model. The other interfaces will be added in future versions of this specification. The XML style mechanism **XSL** [15] which is at the draft stage - will be interfaced in the later versions of the **DOM**.

• The Cascading Style Sheets is a declarative syntax for defining presentation rules, properties and ancillary constructs used to format and render Web documents. This document specifies a mechanism to programmatically access and modify the rich style and presentation control provided by the CSS.

Appendix 2

Resource Description Framework RDF

It is important to understand the difference between the basic XML as a syntax and language and the RDF which uses the XML syntax (could later on use other models) but works as a semantic organizer of resources, and as a framework for any machine-understandable web resources.

• RDF provides a **logical** data model for W3 resources while XML is a syntax, a physical data model. Analogy: In relational DBs the relational algebra is logical and the tab delimited tables the physical model.

• RDF is a foundation for processing metadata; it provides **interoperability** between applications which need to exchange machine-understandable information and data on the web.

• RDF emphasizes facilities to enable **automated processing** of web resources. RDF can be used in a variety of application areas; for example: in resource **discovery** to provide better **search engines,** in **cataloging** for describing the content relationships on any particular web site, in intelligent **software agents** to facilitate knowledge sharing and exchange, in content **rating,** in **digital signatures,** in describing **collections** of pages, in describing **intellectual property** rights of web pages, and in expressing the **privacy** preferences of a user, as well as the privacy policies on web sites.

• RDF metadata maximizes the interoperability of **independently** developed web servers and clients.

• One of the goals of RDF is to make it possible to specify the **semantics for data,** based on XML, in a standardized, interoperable manner.

• RDF and XML are complementary: RDF is a **metadata model** and it only addresses by reference many of the encoding issues needed for transport and file storage (such as internationalization, and character sets). For these issues, RDF relies on the support of XML.

• It is also important to understand that the XML syntax is **only one** possible syntax for RDF and that alternate ways to represent the same RDF data model may emerge.

• The broad goal of RDF is to define a **mechanism for describing** resources without any assumptions about a particular application domain, or a priori definitions for the semantics of any application domain. The definition of the mechanism should be domain-neutral, yet the mechanism should be suitable for describing information about any domain.

• RDF will have a **class system,** much like many object-oriented programming and modelling systems.
A collection of classes (typically authored for a specific purpose or domain) is called a schema. Classes are organized in a hierarchy, and they provide extensibility through subclass refinement.

Source: This brief RDF primer was freely edited from the Introduction [12] of the final RDF Recommendation of 22 February 1999, at www.w3.org/TR/REC-rdf-syntax

Specification of the UPAS demo

In a brief and simplified demo we produce regular orders for print jobs with a network printer. Jobs contain pages with typical media contents (text, image, web and database content). The contents are web files in native, PDF and XML formats, linked to the **jobwide document** on a server called UPASdemo. This logical document is supposed to be updated by the process owner (e.g. a publisher), and then converted into physical files for output at any time. Before the output, the documents are checked and the final document is sent as PDF chunks, for the printer output workflow.

Input job files and servers

Text files	Files including text blocks (Word) and ready pages (PDF)
Image files	Image files linked to the pages (e.g. TIFF or GIF)
Web site content	HTML pages on identified, ready-searched public web sites
Database	Fields to be used to update outputs Tables or Graphs on data pages (option)

Servers and XML application

A group of web servers hold the content files for the jobs and page documents on the web. In this demo no well secured extranet is needed. One of the servers is the "host" for the UPASdemo, the job-wide XML documents, file manager or DB and a typical press frontend workflow, i.e.the Open™ of Luminous [1].

1. Page level: XML is used to edit the document structure of **XML pages** by using "the inner" DTD and the listed types of contents. Only one inner, sub-page DTD is used in the UPASdemo. These page input processes are carried out on any of the servers (in practice some of the VTT IT´s web servers).

• The XML pages are XSL-styled and then printed in PDF, as shown in the lefthand column of **Fig1**.

• The ready PDF page files represent the PDF-coded web contents (the middle column of **Fig 1**)

• The states of the content files may be anything from empty or final native files to ready XML pages in PDF. In the ready state, the link (from the job level) connects the content page to the jobwide XML document (see **Fig 1**). These ready pages could also be distributed web content (in PDF or HTML) anywhere on the extranet.

2. Job-level workflow. This part of the XML web application uses the UPASdemo server which feeds the jobs, as soon as they are ready, into the printing workflow, in this case Open™ [1].

• The jobwide XML documents contain the proper number of pages (e.g. the number of pages on the press sheet and in the whole job). These press sheet and jobwide documents may also be distributed, if needed. They use another "outer" or jobwide DTD, including the possibility of several different press sheets (each with a given page number) and other workflow references. See the lower part of Fig 1.

• DTD is used to structure the outer hierarchy (from pages to press sheets and jobs) of the orders, and to include printing and finishing data in the XML metadata, which means that this demo is a web application. This job-specific XML metadata may parsed for tracking (bu some legacy PMS) and its PDF files printed.

• The output of any intermediate states of the press sheets and jobs are demoed with some simple proofing features in different stages.

3. RDF option: A separate metadata repository in RDF is used to track and store the output jobs. The linking elements connect any documents for the possible reuse processes.

- For the reuse the RDF format helps to describe, catalog and rate the job-specific contents.
- This is just a demo version of a mediabank or document and image repository.

Printing

Output printing is done on a low-resolution device and only in black-on-white. Before the printing the job files may be sent as a PDF folder into a proprietary workflow (i.e. Luminous Open™) for the press sheet assembly, imposition and other file preparation stages. Printing and proofing may not be defined as separate subprocesses in the demo system.

Integration work for real press frontends [1] would require system development by the vendor, knowing the system with its interfaces and source code. The vendors doing this would get their proprietary system transformed to a web application with a relatively modest effort.