

POLICY-BASED EXTRACTION AND INSERTION FOR DOCUMENT STREAMS IN CONCURRENT AND COMPUTER-AIDED ENGINEERING

Ivo Georgiev¹, Iliya K. Georgiev², Olga M. Beltcheva³

¹Ambaric Searchlight

Lafayette, CO 80026, USA
e-mail: ivogeorg@gmail.com

²Metro State College of Denver

Mathematical and Computer Science Department
Campus Box 38, Denver, CO 80217, USA
e-mail: gueorgil@mscd.edu

³Oracle Deutschland GmbH

Riesstrasse 25, 80992 Munich, Germany
e-mail: olga.dr.beltcheva@oracle.com

Keywords: *n-tier architecture, enterprise integration, engineering document stream, web services, web arrays, document access policy*

Abstract. *Continuous product lifecycle management spans a large distribution of service-oriented n-tier architectures to achieve an integrated concurrent and computer-aided engineering (CCAЕ) environment. The product lifecycle is supported by an engineering document stream.*

A document stream contains an enormous diversity of standards, protocols, and data structures. Integration is generally achieved by wrapping documents and service procedures in additional XML metadata markup. Dynamic exchange and update is done in a consistent and expressive form by web services.

The paper presents a simple yet flexible approach to extraction and insertion from an engineering document stream based on web arrays – containers of web resources and metadata markup. Web resources can be documents, services, or just data. A policy-based implementation guarantees conformity with requirements for enterprise information integration. This approach works locally and without centralization and so can be scaled and extended to any distributed computing environment including service-oriented architecture, grids, and the global cloud.

1 INTRODUCTION

Enterprise integration is crucial in sustaining growth in today's dynamic global networked economy. Products are conceived in one physical location, designed in another, prototyped in a third, manufactured and tested in a fourth, and marketed globally. Enterprise integration involves virtualization and management of web services in distributed domains of design, manufacturing, and administration environment^[11]. Exchanging of documents in enterprise n-tier architecture presents several challenges such as document lifecycle management that already exists in implicit form in every enterprise information system and affects a lot of industrial verticals. The implementation is a permanent optimization process with parameters information value, security requirements, service-level agreements, and existing storage infrastructure.

Web services in networked economy are to support the main phases in the lifecycle of documents – especially their creation, use and management. The implementation is strongly influenced by the application areas with enormous diversity of standards, protocols, and data structures. General approach is to wrap documents and procedures in XML^[7] and organize their dynamical exchange and update in a consistent form that is performable and extendable for use in a real-world enterprise environment.

There are a number of reasons why converting and archiving the design data in XML is a dominating technology. XML achieves interoperability between various design automation databases, and can provide some longevity to your design database. For example, if you need to resurrect an older design database after years, the licenses for that CCAЕ software will have long expired and you may not even be able to find a machine with that version of the CCAЕ tools and operating system anymore. However, using XML you can just use a browser plug in to view that database, or an XML import module to bring it into the current generation of design automation tools.

The sources of our research fall into the areas of web services specification and engineering document management:

a. Web services are object of several initiatives in WWW Committee (in www.w3.org) and Organization for the Advancement of Structured Information Standards OASIS (in <http://www.oasis-open.org/home/index.php>), whose authors are from the leading software vendors. Important documents are WS-Naming, WS-MF and WS-Metadata, WS Data Access interface, WS Security, WS Security architectures and WS Trust language, etc. They do not give exact

recommendations how to create enterprise n-tier architecture, but provide platform independent cooperation between the company software located on different computers and clusters. Interesting consideration are social networks, which could influence the integration in web-driven distributed systems.

b. A number of publications give different view and methods of engineering documents management in distributed engineering computing. Some authors ^[1] place special emphasis upon using the dynamics of the engineering documents to organize distributed design automation. Others ^[2] address the security paradigm of concurrent engineering. Several publications are related to open grid architectures to enable the construction of interoperable engineering grid ^[4,5].

Our research proposes an abstract view of the enterprise documents in *n*-tier architecture as a XML stream. Different application software packages extract from the stream the needed information using policies. The latter can be implemented as a thin layer of abstraction on top of the standard information integration services. The XML document stream is processed within web arrays that are XProc^[6] compatible pipelines. Such stream abstraction is based on the evolution and standardization of the service and grid oriented architectures and can be applied as enterprise integration approach in CCAE.

The paper is structured as follows. The foundations (web services and XML-based languages for CAE purposes) are described in section 2. Section 3 explains the XML stream modelling used for information web service prototyping. The stream processing based on web arrays is presented in section 4. Section 5 introduces our policy-based approach for insertion and extraction that allows for virtualization of the document stream. Section 6 concludes the paper.

2 FOUNDATIONS

2.1 Web services

Web services ^[5] are intended to be provided by an application to other software applications using web protocols and technologies. At their lowest level, web services are typically based on transmitting XML documents between clients and servers via HTTP. A web service accepts HTTP requests from a client application (not end user) and provides an HTTP response for each request received. The body of every HTTP request contains a document that is encoded in a XML vocabulary. This document specifies an operation to be performed by the web service and supplies input data for that operation. After processing a request, the web service returns the processing result as a XML message. Tools exist for writing client and server code at a high level that hides the implementation details. A key to make these tools possible is another XML vocabulary for describing web services, which identifies the operations provided by the service, what the input data for each operation is, and what output data is provided by the operation.

Web services let the engineering enterprise developers dynamically to grow applications by creating compound solutions that use internal organizational software assets, including enterprise information and legacy systems and combining these solutions with external components possibly residing in the networked economy.

2.2 XML documents in the networked economy

Storing and processing CCAE documents in XML format is a dominating archival solution because it is in full compliance with the requirements of networked economy and are easy to parse and validate. Analysing the mark-ups every software engineering application can access engineering documents that were built in different nodes for use in unanticipated contexts and select parts of the documents according to its content.

On one side, for global integration one needs to mark up manually various conventional documents, database tables, messages from mobile devices and sensors, etc. Using these mark-ups, the web services and activated software applications can interpret and reason with the information. Establishing semantics over the marked-up information pathways plays a key role for numerous engineering tasks such as comparing models to understand the semantic differences, mapping terms and entities to ensure product data integration, or simply reusing entities and queries to product data to prevent duplication of work.

On the other side most engineering tools present the engineering and business information in domain specific XML vocabularies. Engineering societies in all areas create XML variants of the design and manufacturing languages. Some of them related to the distributed CCAE are the following:

a. The Process Specification Language (PSL) ^[10] defines a neutral representation for manufacturing processes that supports automated reasoning. Process data is used throughout the life cycle of a product, from early indications of manufacturing process flagged during design, through process planning, validation, production scheduling and control.

b. STEP-XML^[8] (Standard for Exchange of Product model data) provides a mechanism that is capable of describing mechanical product data throughout the life cycle of a product, independent from any particular system. The nature of this description makes it suitable not only for neutral file exchange, but also as a basis for implementing and sharing product databases and archiving;

c. Integration of XML and EDIF- XML technology providing some synergy with EDA/EDIF electronic design specific tools. Electronic Tools Company has developed an XML Document Type Definition (DTD) called

EdaXML.dtd based on the EXPRESS Information Model used in the design of EDIF 4.0.0. Thus the EdaXML.dtd can represent anything that EDIF 4.0.0 can represent, including schematic diagrams, net list connectivity, printed circuit boards (PCB), and multi-chip modules (MCM).

d. The Systems Modeling Language (SML) is a general-purpose graphical modeling language for specifying, analyzing, designing, and verifying complex systems that may include hardware, software, information, personnel, procedures, and facilities. In particular, the language provides graphical representations with a semantic foundation for modeling system requirements, behavior, structure, and parametric, which is used to integrate with other engineering analysis models.

e. Materials Markup Language ^[9] enhance the e-business needs of materials producers, designers, fabricators, quality assurance, and failure analysis specialists by developing an XML-based standard markup language for the exchange of all types of materials technical information, while retaining its high quality and integrity despite its complexity.

3 DOCUMENT STREAM MODELLING

Engineering documents are distributed in several different locations of enterprise infrastructure. Document clients have a number of restrictions to directly access the data due to the disjoint administrative policies that provide accounts between sub-organizations. Replication the data to a common data warehouse might theoretically be best for product lifecycle management but might be not practical. Data needs to be accessed directly. We proceed to propose an abstract model of engineering document dynamics as a technological basis.

We model the set of engineering documents for an end-to-end production process as an *abstract XML stream*. The stream provides both metadata for access control and references to document databases. Figure 1(a) presents general view of the structure of the stream. Every part of the stream contains metadata describing specific enterprise information. On the figure two parts are schematically presented: for mechanical product and for electronic device. The parts are individually encrypted. Every client's (designer, manufacturer, administrator, salesman, etc.) application software can use web services to get the stream but can decrypt only the parts that are oriented to his enterprise role. After authentication and proof of permissions the client software can open the metadata and parse it for availability of the needed version of the specific documents. Afterwards the client has open access the required document databases for the duration of the session.

The stream model provides three system-architectural advantages. First, since the stream exists independently of the local client systems, it eases the storage and processing requirements on them. Second, since the stream contains the minimum sufficient data and control for every enterprise role, it enables complete modularization and distribution of the provided services. Third, based on the above two assumptions it allows the continuous in-vivo evolution of the engineering process and eliminates step-wise system upgrades. It is further advantageous to model a *continuous* document stream. The advantages are: (a) quick iterations and continuous integration on all production levels; (b) dynamic optimization of the production process to track market fluctuations; (c) general stream cyclicity, necessary for rapid prototyping and ever shrinking product lifecycles; and (d) asynchronicity and concurrency of service provision wherever there are no explicit constraints. The benefits for the CCAE environment are as follows: (a) anticipation of full bandwidth utilization of the distributed enterprise; (b) processing of product data exceeding any local storage capacity; (c) real-time processing; (d) statelessness (state is implicit as a point in the steam), which eliminates artificial temporal and sequential interdependencies among functional subsystems.

Figure 1(b) shows the cyclical document stream in parallel with the continuous iterative product lifecycle.

4 WEB ARRAYS AS A DOCUMENT STREAM TECHNOLOGY

Web arrays ^[14] are generalized extensible recursive Web resources with a concrete representation as self-describing XML (e.g. Atom format) documents. As generic objects they favor REST implementations. As XML resources they are processed by XProc pipelines. While the array abstraction is the most generic and simple vehicle, more complex data organization can be easily superimposed.

Web arrays are collections of web resources an authorized engineer interacts with while manipulating the engineering documents for a particular functional stage of the production process. As the engineer selects a sequence of relevant resources, they are encapsulated in the context of a web array. Such a free-form collection can achieve arbitrary complexity as a direct result of the engineer's purpose and is the clearest reflection of his perception of meaning.

The web items that the engineer put into a web array can be:

- a. Document references that are provided by the document stream. Such references could be information or other web services.
- b. Some URL's of company's web sites, scientific and technical sources (for example, Wikipedia page).
- c. Human readable text or multimedia files with some manuals, readme files, engineering comments oriented to the required collaboration process.

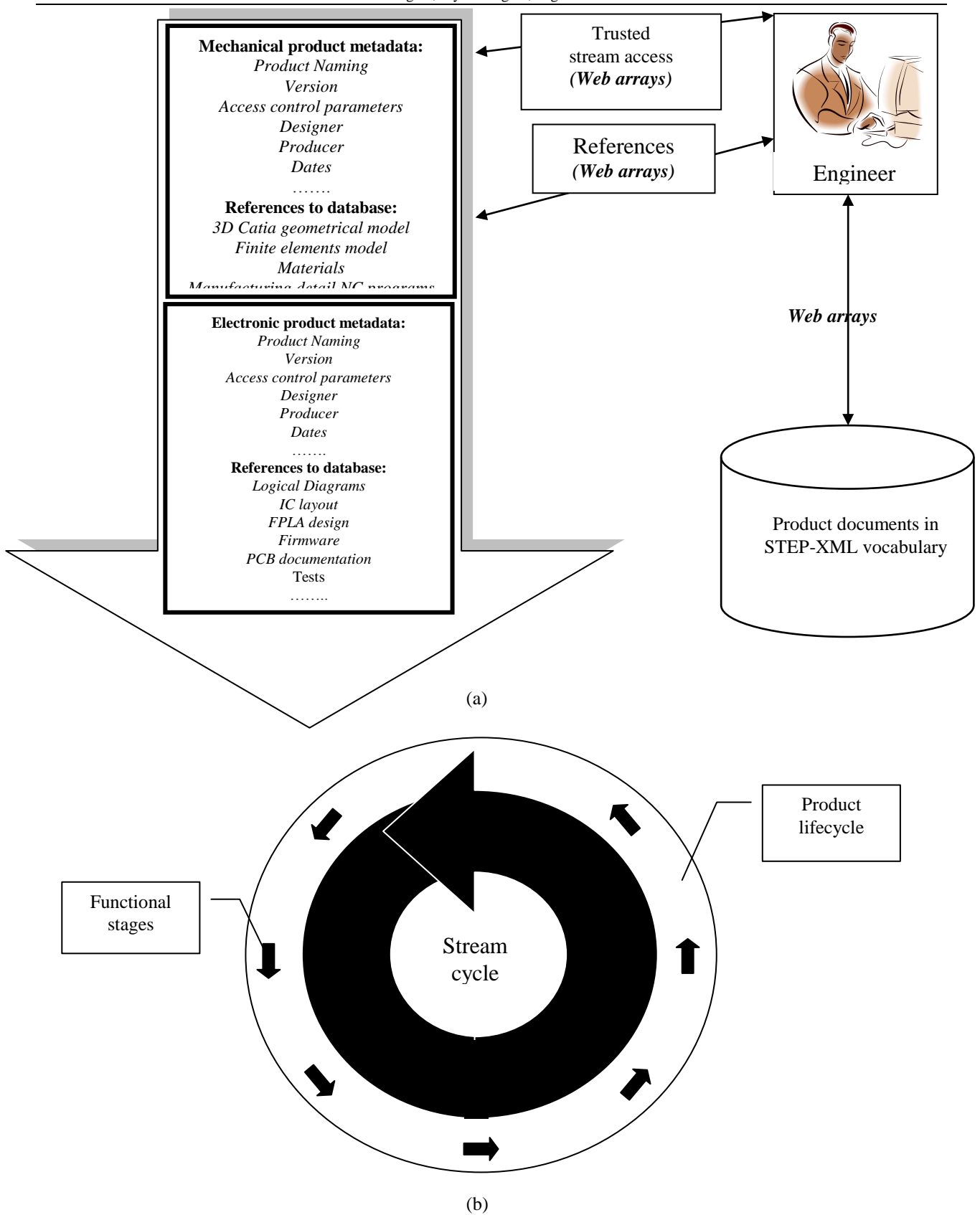


Figure 1. Abstract model of the engineering document stream.
(a) Local view.
(b) Globally, the stream is continuous and cyclical.

Figure 2 shows a single example of the content of a web array for the electronic engineer to get the logical and functional test for some FPLA device.

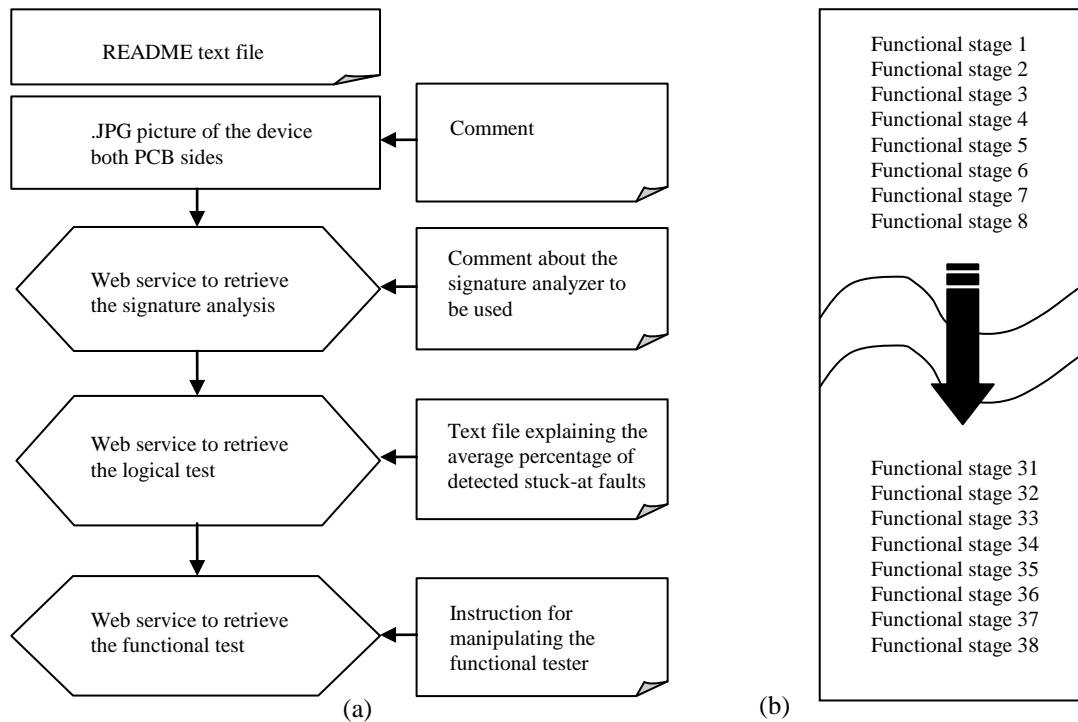


Figure 2. (a) Web array example. (b) The web array as a running record of the engineering process.

4.1 Web array roles in CCAE

A web array can be assigned one or several use roles to define its purpose or function explicitly. Different web services can manipulate and process web arrays with particular use roles. In the extreme case, a web array may have no visible (document) content but contain processing instructions for a number of services.

Different services referenced in the web array may process different entry contents or entry-referenced datasets in a web array. Each client is specialized to process the data, instructions, and documents relevant to a particular functional stage or the CCAE process.

Web arrays are well suited to perform the following roles: (a) service composition threads; (b) orchestration vehicles for CCAE (analogous to full scores for orchestral music); (c) authentication keys and certificates; (d) running records for CCAE process (incl. initial blueprint, prototypes, bill of materials, test suite results, etc.). See Figure 2(b). As functional stages become increasingly modularized and democratized, web arrays allow iterations in the process where previously there have not been any.

4.2 Stream extraction and insertion with web arrays

Web arrays are manipulated via HTTP verbs on simple URLs. The core of the service contains XProc pipelines^[6] and an action dispatcher. The XProc pipelines apply various general transformations to the incoming and outgoing XML streams. The action dispatcher invokes the functional modules, which are broken off as separate subservices, according to the actions specified in the service requests. The extensibility of web arrays allows them to be much more than units of information where static resources are linked in a new context. Web arrays can reference dynamic data sources and web services just as easily. The literal annotation allowed in web arrays can in turn be used to bind data and function in the unified execution context of the web array as a unit of processing.

Web arrays are meant to generalize the following behavior: while the web array document format is standard and can be manipulated by generic services, the contents may be arbitrary and meant to be processed only by specialized, authorized, and monitored functional locales.

As mentioned, product lifecycles are more often than not cyclical and the stream abstraction captures that by being continuous. Figure 3 illustrates how web arrays implement stream extraction and insertion at functional locales. Figure 3(a) shows how a web array carrying credentials and policy attributes is used as key to trigger an extraction process for required data and control information. Similarly, a web array is used to record the results and apply quality stamps to the results of the performance of the functional stage of the local service. Figure 3(b) shows a “horseshoe” abstraction of the U-shaped local XProc pipeline. Web arrays are active at the attachment points and suggesting a new paradigm – distributed participatory attachment to the CCAE document stream. Figure 3(c) shows how the stream

cycle can be represented as a partially ordered collection of concurrent horseshoes corresponding to partially ordered sets of functional stages in the engineering process.

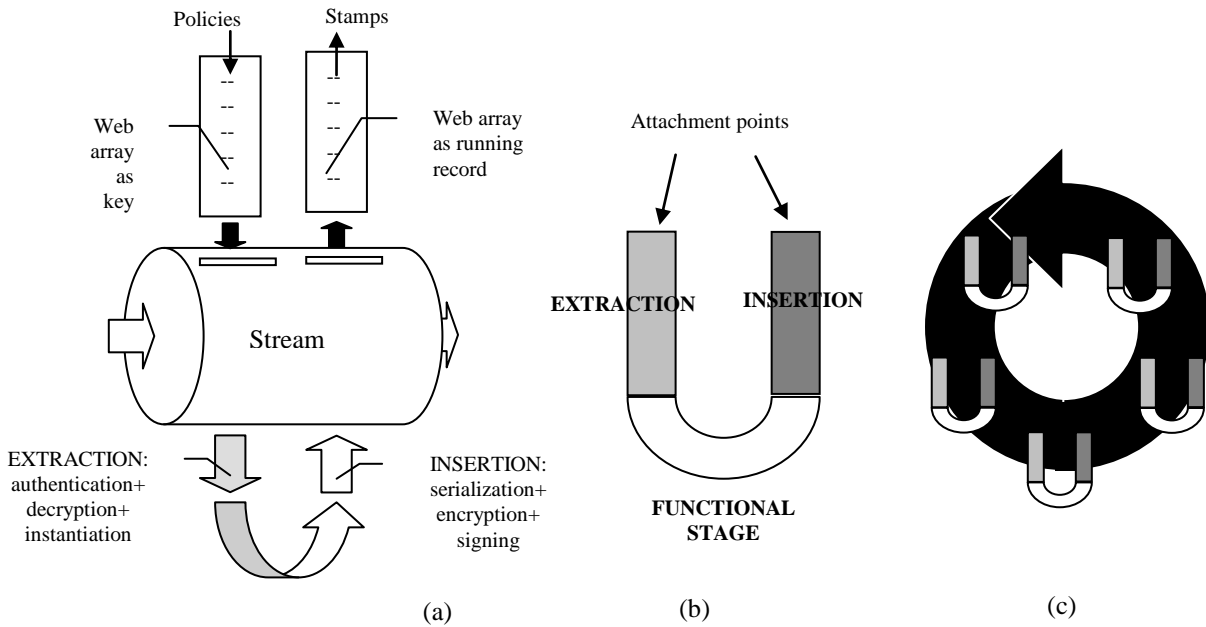


Figure 3. (a) Stream insertion & extraction. (b) XProc horseshoe. (c) Stream cycle of concurrent horseshoes.

While globally the stream is continuous, there is a local “horizon” at each functional stage. Figure 4(a) shows that the local horizon is nothing else but the collection of outstanding tasks which can be performed at the current locale. Attachment to the stream is done by selective bidding on tasks and is at the local controlling authority (human or automated). If the attachment is successful, a portion of the stream is redirected through the local horseshoe pipeline. The performance of the functional stage is equivalent to stepwise downstream advancement in the forward direction of the lifecycle, as shown on Figure 4(b).

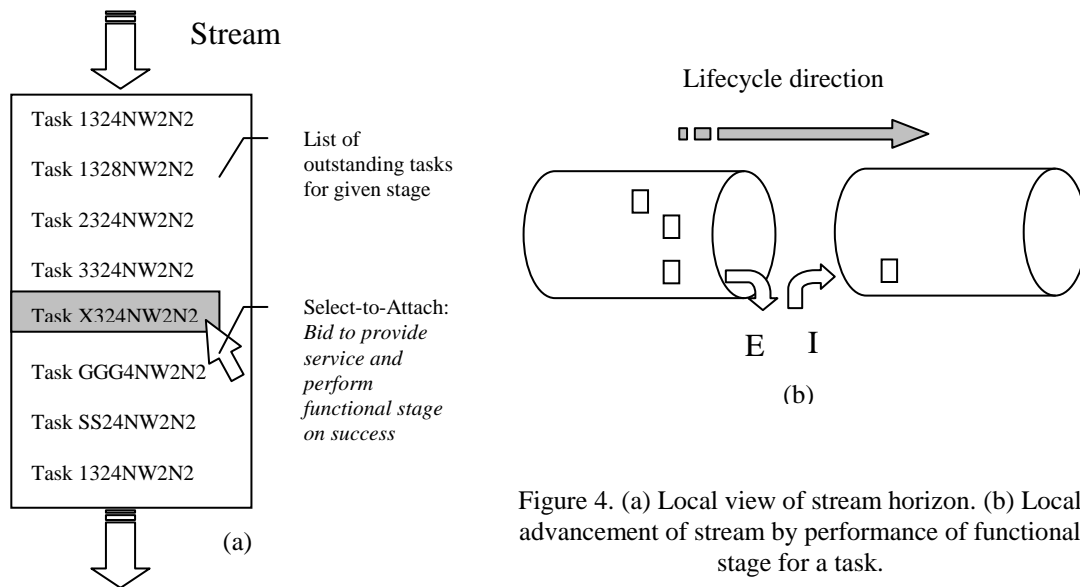


Figure 4. (a) Local view of stream horizon. (b) Local advancement of stream by performance of functional stage for a task.

4.3 Enterprise infrastructure and deployment in CCAE

The required enterprise infrastructure necessary to deploy web arrays for document stream extraction and insertion in CCAE can be overlaid on incumbent platforms. First, on the usage side of web arrays, service providers for each functional stage have to implement horseshoe pipelines. Second, on the authoring side of web arrays, there is need of a process engineering platform with composition and orchestration capabilities. Lastly, lifecycle control authorities need to be installed and configured. This distribution of the infrastructural components allow for relatively painless update and management of the extraction-insertion process and policy sets. The centralization of control ensures proper orchestration and coordinated recovery in case of emergency.

5 POLICY-BASED EXTRACTION AND INSERTION FOR DOCUMENT STREAM MANAGEMENT

In the interest of maximum flexibility in CCAE systems, stream extraction and insertion permissions should not be statically determined at the beginning of a cycle, but instead allow for dynamic evolution throughout the process. We propose a policy-based implementation of the stream manipulation technology. Policies should be considered as rule sets which provide the right behavior at the attachment points to the document stream. Policies for extraction and insertion for document streams are created as supplement to the document lifecycle management policies in CAE^[13]. Such approach facilitates the design and development of document stream applications as web services and allows for conformity with Information Integration Services in a SOA environment.

Policy implementation is XML metadata-based. It relies on metadata attributes and metadata dependency descriptions. As exposed in^[12], metadata management is considered as pervasive with all web services for document stream implementation, and not only as a component at the bottom of the information management stack (Fig. 5(a)). The document stream contains directly included metadata or references to metadata, stored in a metadata repository. Insertion and extraction policies for document streams are built on top of the policies for lifecycle management: data access policies, data migration policies, and compliance policies.

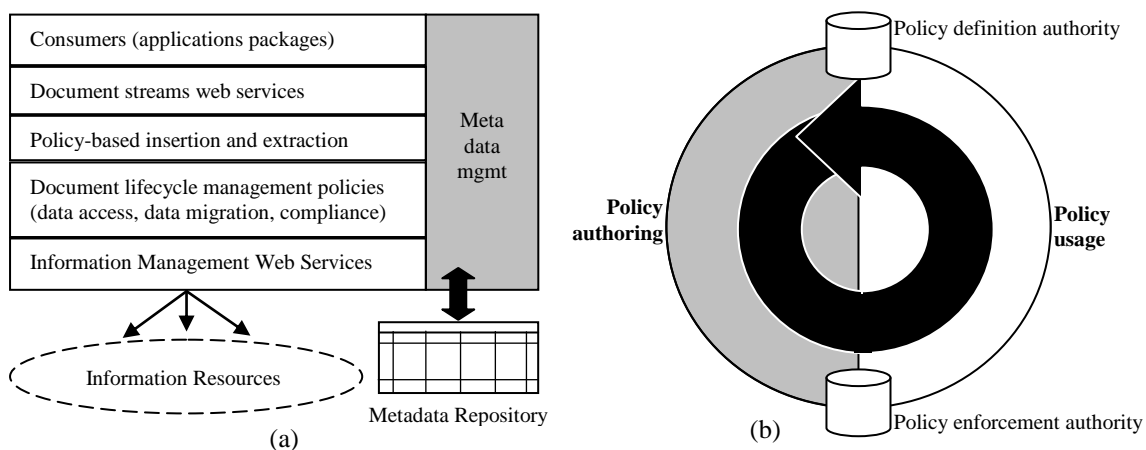


Figure 5. (a) Policy-based insertion and extraction services for document streams. (b) Separation of policy authoring and usage (horizontal), and definition and enforcement authorities (vertical).

Insertion policies are applied each time a document is inserted into a document stream. They prepare and insert into the document stream the set of accompanying metadata thus creating the prerequisites for subsequent use of the stream – viewing, document extraction etc. The metadata values are determined on the basis of the underlying metadata level (document lifecycle management), and the following specifications:

- Product identification - to which product refers the document.
- Engineering or business purpose of the document.
- Phase of the engineering.
- Requirements for document grouping: some documents can be used only in relationship to others. The insertion of a document into a stream implies insertion of other related documents. This is the case of “compound” CAE documents, in which to a single document multiple reference materials may belong.
- Access rights to documents (privacy) inherited by the underlying lifecycle management metadata level.
- References to metadata connected to applied data migration policies.
- Other attributes: product naming, version, designer, producer, dates, specific content etc.

Extraction policies fetch and analyze the document metadata, and are responsible for the generation of the document stream for the consumer – CAAE application software packages. The metadata of each document is analyzed; the data access, compliance and other policies, encoded during the insertion are applied, and as a result a virtual view of the document stream is generated for different enterprise roles. A document stream may contain documents that belong to different engineering processes and to a different process phase (active, passive). At the first step the extraction policy determines the access rights to the document for the actual consumer. It uses the references to the access control metadata set by the lifecycle management policies, the stored business purpose of the document and the process status (active, inactive), and applies rules for mapping the access rights of the current consumer’s role to the fetched metadata. For example, a designer can modify a document that belongs to the active design phase, but can only read (view) a document that has been moved to the inactive design phase (after design freeze). Documents, which serve the active manufacturing, can be accessed by support engineers only for reading. As a result of extraction policy application some documents of the stream may become “invisible”: for example a product assembling engineer

is not allowed to see documents of products that are in active redesign phase. Access rights may refer not only to the document content, but also to its attributes – for example information about dates, versions etc. may be hidden. At a second step, if access is permitted, the metadata connected to the references to data sources is evaluated and the document is retrieved by invocation of the underlying information management services. This approach makes possible the reuse of already implemented services for document lifecycle management and standard information integration services.

Since web arrays are already used to contain and/or reference the minimum necessary data and metadata required for the performance of services along the product lifecycle, they are necessarily also the vehicle of choice for the conveyance and enforcement of access control policies. And since web arrays already favor distributed XML pipeline processing along the document stream, the benefits carry over to the policy management process. On one hand, centralization of control, crucial for CCAE, is ensured. On the other hand, it is easy to maintain a sanitary separation of the policy authoring and usage phases, as well as the policy definition and policy enforcement authority locales (Fig 5(b)).

6 CONCLUSIONS

As the basis of an abstract documentary representation of the product lifecycle, we adopt the model of a stream as a running collection of engineering documents concomitant to the CCAE production process. We manipulate the stream with web arrays, abstracted web resources that contain metadata and document references. The metadata part is encrypted for authorized access and provides product specifics of the engineering documentation. The references part is collection of web resources which enable the actual document retrieval from the enterprise databases.

Policy-based stream extraction and insertion based on metadata management allows for CCAE specific extension of the middleware layer between the application packages and data sources and facilitates a service-oriented implementation. We put web arrays as an enabling technology to distribute and extend engineering document lifecycle management. Web arrays are an immediately useful enhancement of web bookmarking, which is easily accessible to the general engineer. As a universal information carrier, they can bind together the numerous web resources into a coherent and growing computing environment. They are a natural candidate to provide a mechanism for a distributed bid-to-service engineering economy.

REFERENCES

- [1] Wand, Y., Nnaji, B., Ciang, W. (2005), "Document-driven Design for Distributed CAD Services in Service-oriented Architecture", *Proceedings of International Design Engineering Technical Conference, IDETC '05*, Long Beach, California, USA, 24-28 September.
- [2] Woerner, J., Woern, H. (2005), "A Security Architecture Integrated Co-operative Engineering Platform for Organised Model Exchange in a Digital Factory Environment", *J. Computers in Industry*, Vol. 56, pp. 347-360.
- [3] WWW Committee (2007), "Mathematical Markup Language (MathML)", <http://www.w3.org/Math/>
- [4] Sobolewski M., Kolonay, R. (2006), "Federated Grid Computing with Interactive Service-oriented Programming", *Advances in Computer Engineering*, v.14, SAGE Journals Publisher, pp. 55-66.
- [5] Grimshaw, A., et al, (2009), "An Open Grid Services Architecture Primer", *J. Computer*, Vol. 42, pp. 27-34.
- [6] Walsh, N., Milowski, A., (2007), "XProc: AnXML Pipeline Language", *W3C Working Draft*, available at www.w3.org/TR/xproc/
- [7] WWW Committee, (2000), "W3C Extensible Markup Language", <http://www.w3.org/XML/>
- [8] Peak, R., Lubell, J., Shrinivazan, V. and Waterbury, S. (2004), "Step, XML, and UML: Complementary Technologies", *J. of Computing and Information Science in Engineering*, vol. 4, pp. 379-390.
- [9] OASIS, (2007), "Materials Markup Language", <http://www.oasis-open.org/committees/>
- [10] NIST, (2008), "Process Specification Language (PSL)", <http://www.mel.nist.gov/psl/>
- [11] Dreibelbis A., Hechler E., Milman I., Oberhofer M., van Run P., Wolfson I. (2008), "Enterprise Master Data Management. An SOA Approach to Managing Core Information", IBM Press.
- [12] Selvage M., Wolfson D, Handy-Bosma J. (2005), "Information management in Service-Oriented Architecture", IBM Press.
- [13] Beltcheva O., Georgiev I. (2008), "Some application aspects of document lifecycle management in Computer Aided Engineering", *Proceedings of the 3rd International Conference "From Scientific Computing to Computational Engineering 3rd SCCE*, Athens, 9-12 July 2008, Copyright © IC-SCCE.
- [14] Georgiev, Ivo, Georgiev, Iliya (2007), "P2P Computing with Web Arrays", *Proceedings of the Web XTech 2007 Conference "The Ubiquitous Web"*, 15-18 May, Paris.