Document Meta-Stream Technology in Networked E-engineering and E-science

I. Georgiev

Key Words: *Document management; XML stream of metadata; web services; web arrays; classified data filtering.*

Abstract. *Web engineering and scientific applications span a large amount of document distribution driven by an enormous diversity of standards, protocols, and data structures. Two trends are dominating: a. presenting the scientific and engineering data in XML documents; b. exchanging the documents by web services. This paper expands these trends by higher degree of document presentation and supporting technologies. Document presentation and management are based on document meta-stream that is a XML document, constructed by different section. Every section contains references to specific document repository. The sections are individually encrypted and provide information to classified users. The following technologies for the meta-stream processing are applied: a. XML streaming that does not need storage of the whole document; b. web arrays that are free-form collection of relevant web resources; c. cryptography classified filtering of the document stream that allows different access policies.*

1. Introduction

The document exchange is crucial for sustaining growth in today's networked engineering enterprises (*e-engineering*) and also in *e-science* for experimental scientific collaboration and discovery, fast access to endless computational resources and data repositories, and high-throughput communications. Documents are conceived in one physical location, designed in another, stored in a third, and distributed globally. Document lifecycle management already exists in implicit form in every information system and affects a lot of engineering and scientific verticals.

General approach is to wrap documents and procedures in XML vocabulary [http://www.w3.org/XML/] and organize their dynamical exchange and update in a consistent form. XML achieves interoperability between various design databases, and can provide some longevity to the database. Integration is generally achieved by wrapping documents and service procedures in additional XML metadata markup. Here are some of the XML languages in the mechanical and electronic industry that we used in our experimental implementation. STEP (Standard for Exchange of Product model data) [9] is capable of describing mechanical product data throughout the life cycle of a product. 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. EdaXML [http://xml.coverpages.org/xmlAndEDA.html] supports cross-platform design using electronic design automation tools from a variety of vendors.

The engineering and scientific application software uses web services [http://www.w3.org/Web services/] for remote resources access in a manner that is not visible to the end user. The web services let users store, manage, catalog, and share large data collection or rapidly evolve novel applications they cannot find anywhere else. Web services are to support the main phases in the lifecycle of documents – especially their creation, use and management.

The document management has to solve two significant problems. The first one is the volume explosion of the exchanged information that includes more and more multimedia documentation streamed not only to the desktops but to phones and other mobile devices. Until now the economy can receive seamless services because Internet still has powerful optical core but not the edge where the web services run below their full capacity. For instance, the scientific exchange is peer-to-peer, which needs large files and consumes very fast bandwidth. The second problem is that the engineers and scientists have to select useful information from the huge databases. In most cases most of the data has to be discarded, while only a part is needed at the time being. In almost all cases, especially in engineering, the access to the documents is classified and such classification is driven by several policies. The conventional technology is to store the whole document collection and to analyze which part is accessible to the current client. It requires transferring all streaming data to the client machine, adding significant cost of storage and communication and the risk of delay or even information lost.

The paper addresses those problems and the *contribution* can be divided in two major achievements:

a. Creation of the document meta-stream model. The model is a XML stream of metadata circulating through all possible customers (engineers, scientists, manufacturers, administration, etc.). The document meta-stream reduces significantly the data volume and provides access control and classification at runtime. The stream elements are object of different encryption and contain only metadata for classified access and references to the real document data bases. Different application software packages extract from the stream the needed information using policies. Such stream abstraction is based on the evolution and standardization of the grid oriented architectures and can be applied as enterprise integration approach.

b. Selection of proved technologies for meta-stream processing: a. XML streaming that performs parsing without creating a XML tree of the whole document that provides possibility to analyze encrypted documents; b. *Web arrays a*s a sequence of relevant resources; c. *Data filtering* of the document meta-steam giving access only to some of the meta-stream sections.

The paper is structured as follows. Section 2 presents the

structure of the document meta-stream. Section 3 defines the technologies used in meta-stream processing. Section 4 explains some results, section 5 concludes the paper.

2. Document Meta-Stream Modelling

As our motivating example, we examined the engineering documents that 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.

For such direct access to the documentation we propose a *compact XML stream* of metadata and references to document database. The metadata provide document management information that is reflected by the selected policy rules. The references to the documents are addresses of the web resources that are provided by the engineering databases. The document meta-stream is circulating through all of the hundreds machines of an engineering enterprise or scientific society. The meta-stream is divided in different sections (XML elements). Every section has policy metadata and references to a group of documents that are object of industrial hierarchy or scientific interests. Every stream section has one authorized author (writer) and several users (readers). In most of the cases the access to different stream sections is restricted. The author creates the contents of the section and is responsible for all updates. The users can have every-day suggestions for some improvements and updates of the contents of the section, but all the changes have to go through the author. Such approach satisfied the requirements of the centralized product lifecycle management.

Figure 1 presents general view of the structure of the meta-stream for some engineering enterprise that designs and produces complex products containing mechanical and electronic parts. Every section of the stream is encapsulated in XML begin and end tags. The section XML vocabulary could be different, but we suggest the selected vocabulary to follow some of the mentioned above XML engineering languages. On the *figure* three sections are schematically presented. General information section contains description of the engineering domain (product lists, development/manufacturing/testing specifics, etc.) and is recommended to be described in Process Specification Language namespace. The mechanical part section contains metadata (product documentation structure, versioning, access parameters, etc.) and follows the STEP-XML specification. The references are URL addresses, identifiers of web services and web arrays to the design and manufacturing documents. The electronic part section contains similar metadata and follows the EDIF-XML specification. The references are to logic diagrams, test sequences, layouts and so on.

The sections are individually encrypted. Every client's (designer, manufacturer, administrator, salesman, etc.) application software can use web services to get the whole stream but can decrypt only the sections 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 receives open access to 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 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.

3. Technologies for Document Meta-Stream Processing

3.1. XML Streaming

Some related works have influenced our approach for meta-stream processing. Review analysis of such works is presented in [1] together with an evaluating method of a large collection of XPath expressions. Table-driven streaming is proposed in [3]. Run-time monitoring of web services interaction based on XML streaming engine is given in [2]. Stream-based processing enhanced to exploit encoding specиыc optimizations such skip-to-tags is proposed in [11].

XML documents processing could be performed in two ways. The classical mode is based on XML document model, which stores the whole document, performs a tag parsing and creates a tree. Every query activates tree traversal and retrieval of the arbitrary parts of the document by the specifying the path to the desired nodes. Although reliable and strait forwarded, this method suffers from the fact that the whole document has to be downloaded and decrypted in the memory in order to construct the tree. For huge engineering or scientific (for example, genome) document streams, tree-based engines are unable to provide successful processing. Additionally, if the XML file is encrypted, parsing and creating of tree needs to decrypt the whole XML file, which is not acceptable for our application.

The alternative approach is the XML streaming. During the parsing the XML document is considered as a sequence of events that is led in the order of their arrival as it is parsed and updates its state. Now the engine without tree construction can perform some sieving algorithms that are based on different policies. Developing such processing is generally more complex than the tree-based method. Tracking of the nested elements has to be securely performed and presented as a result of the current query. In counterpart, the XML parsing does not need storage of the whole document and can parse encrypted document stream.

3.2. Web Arrays as a Document Stream Technology

Web arrays [8] are collections of web resources an authorized user interacts with while manipulating the engineering documents for a particular functional stage of the production

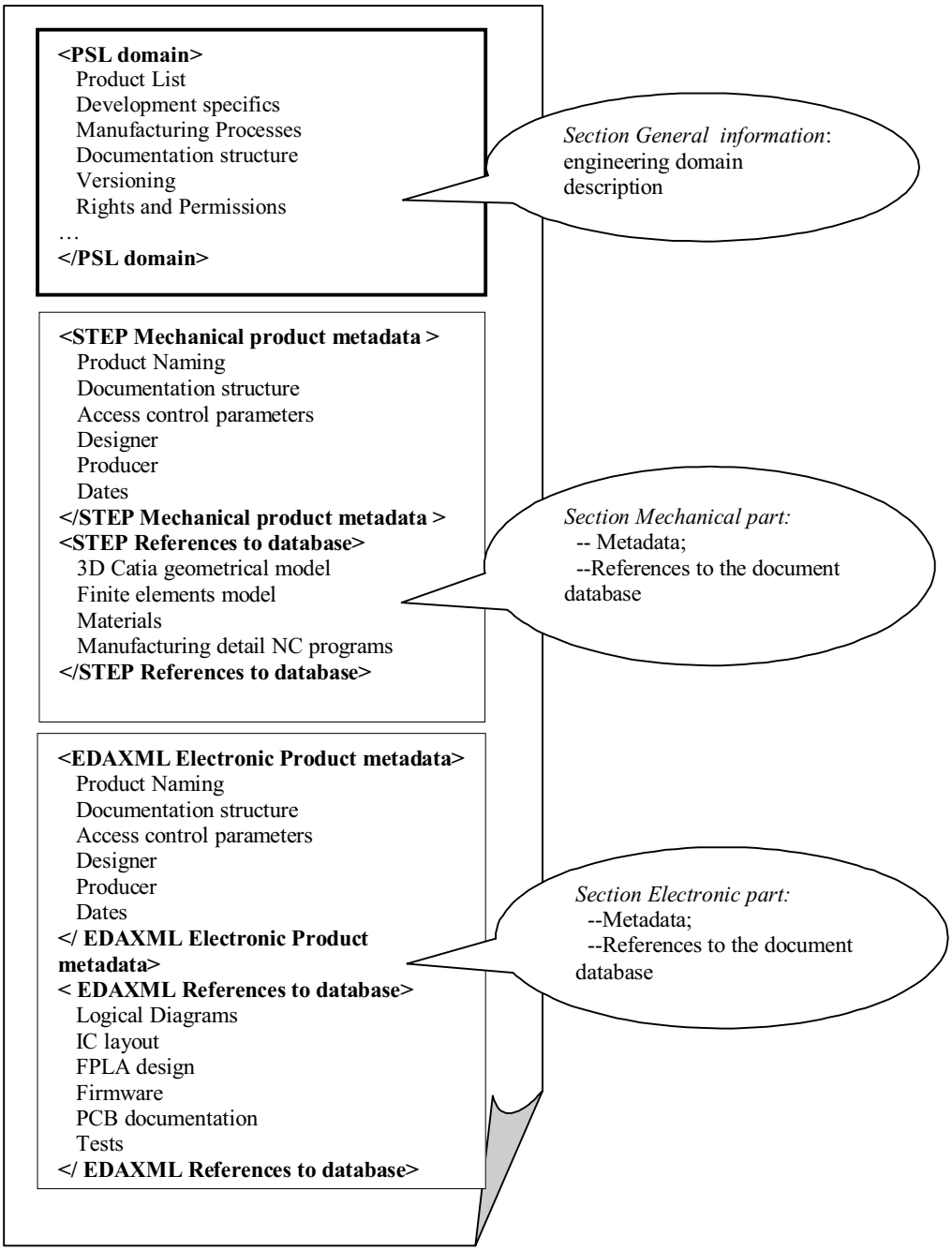

Figure 1. Abstract model of the engineering document stream

process. As the user 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 in 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 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.

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.

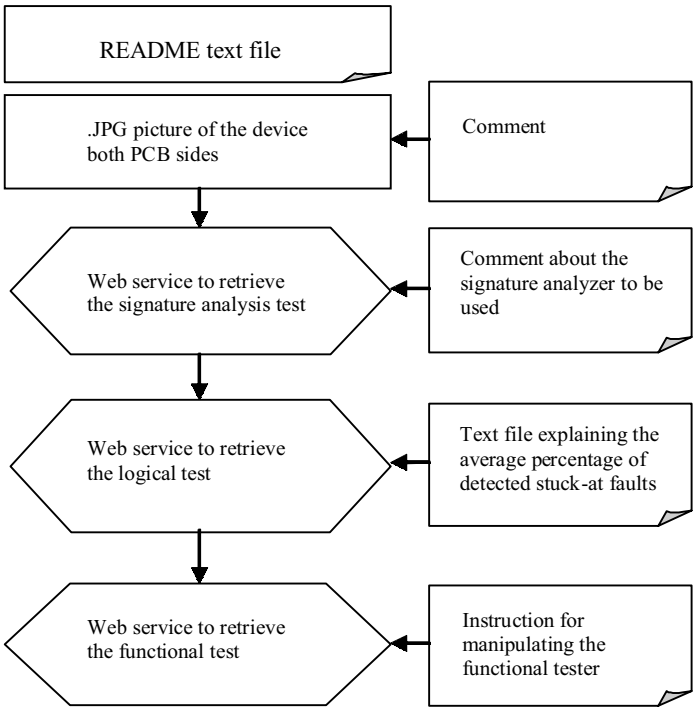

Figure 2. Web array example

3.3. Data Filtering of the Document Meta-Stream

The author and the users usually have access only to some of the meta-stream sections. Let us consider one meta-stream that has to distribute satellite pictures through different groups of scientists. Some of the pictures are raw. Others are already object of image processing and pattern recognition algorithms. Third are already upgraded by special interest groups with some computer graphics, for instance archeological objects, new architectural projects and so on. The special interest group of the archeologists has one authorized person (author), who filters all the proposed changes and keep the section intact. All others from the group are users and have access only to their section with pictures from the whole stream.

The whole documents meta-stream is encrypted and the parsing and the retrieval have to be classified. What that mean is, that if some user needs to extract information for which she/he is classified, the meta-stream is searched only for XML tags that are identifiers of the meta-stream element, which contains the references to the information permitted for such user. But there are hundred users distributed over the world, all can open and retrieve the same XML element from the meta-stream.

The technologies that we have selected for classified collection on streaming data are based on cryptography.

Both public key encryption and symmetric encryption are used. Public key encryption is based on two interrelated keys: the public key is known and available to everyone, the secret key

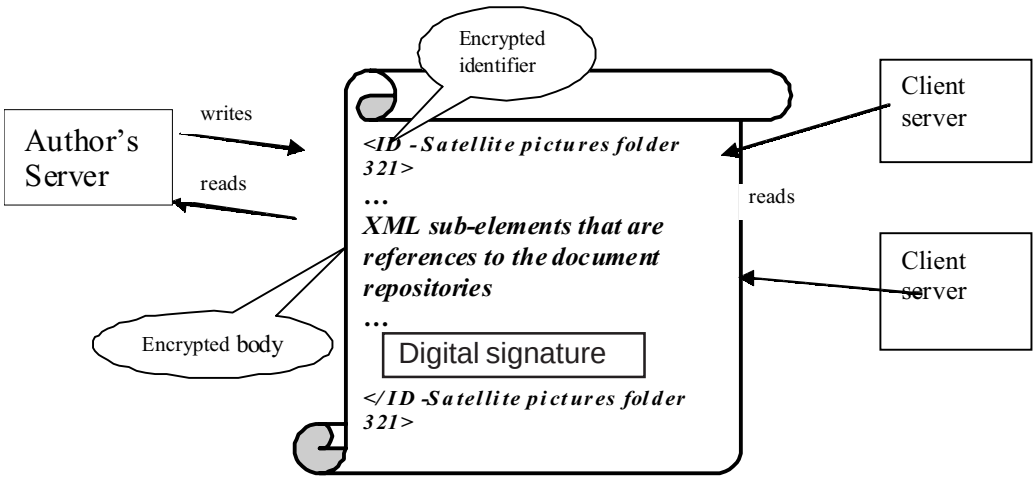

Figure 3. Meta-stream element encryption

is known by the decrypting algorithm. The information is encrypted using the public key and decrypted only by the secret key. Symmetric encryption and decryption use the same secret key.

Figure 3 is to help the explanation of the proposed stream element encryption. Every section in the document is individually encrypted by the author server and only authorized users can decrypt the contents. There are three problems to be addressed:

- classified processing of the begin and end tag of the XML element that contains the section of the document references;
- encryption of the section body (all sub elements);
- section integrity protection.

The first problem springs from the requirement that the entire stream is encrypted (every section individually). The client's servers cannot provide XML parsing over encrypted data and see the tags of the XML elements. Full decryption of the stream is not possible, because different sections are oriented for classified usage of different client servers. The client servers have to have the ability to parse the stream and to recognize the identifier of corresponding section. Here we accept with some simplification the method proposed in [6], called public-key encryption with keyword search. Additionally the work presented in [5] theoretically states that the classified XML search is provable protected from an adversary.

The tag is encrypted by the author server by a public key. The identifier of the tag together with the public key is concatenated to the tag information. The client server gives to the XML parser the so called trapdoor based on the secret key, which allows the decryption of the tag identifier. Such approach can be well incorporated in the XML streaming parser because it does not need storage of the whole stream in a secure environment.

The second problem, encryption of the section body, is preferable to be encrypted-decrypted by symmetric cryptography. The known problem here is the distribution and management of the symmetric keys, which has standardized solutions (for example Kerberos algorithm provided by all operating systems).

The third problem is the data integrity that can mean ensuring meta stream is complete and identically maintained during any operation (such as the transfer, storage or retrieval). The preservation of data has to be guaranteed for their intended use, or, relative to specified operations, the a priori expectation of data quality. Put simply, data integrity is the assurance that data is consistent and correct. Hash encryption enables data quality with a mechanism known as a *digital signature*. A digital signature verifies the authenticity of electronic documents and provides strong security. In order to create a digital signature, the author first generates a small unique digest of the section, generated by the hashing cryptography. Even a very minor change to the original data will cause the hash value of the modified section to be different from the original hash. By comparing the hash that was received with the hash calculated from the received document, the client server can verify whether the document was altered. The hash of the section signed or encrypted with the author's private key acts as a digital signature for that section and is concatenated to the section contents. The client will be able to verify or decrypt the signature by taking a hash of the message and verifying it with the signature that accompanied the message and the sender's public key.

We use in our work XML digital signatures [7] that enable a sender to cryptographically sign data, and the signatures can then be used as authentication credentials or a way to check data integrity. XML signatures can be applied to any XML resource, such as XML, an HTML page, binary-encoded data such as a gif file, and XML-encoded data. The standout feature of the XML digital signature is its ability to sign only specific portions of the XML document.

4. Experimental Implementation and Results

Leveraging the described model, we have created a framework for the meta-stream design implementing experimentally two prototypes of meta-stream. The first prototype was for a company that designs and produces some aerospace equipment with different subsystems. The company documentation occupies 2 terabytes, stored in four data repositories. The size of the meta-stream was 21 megabytes. The other prototype was for a scientific project that is supposed to create a model of water resources on continental level. The collected information (satellite pictures, GIS files, etc.) was about 67 terabytes. The experimental meta-stream occupies 17 megabytes. In general, the opinion of the developers about our prototyping approach has been very positive especially in the decision making circle.

Even at such an early stage, we were able to address serious challenges in meta-stream deployment. Most of them are related to the incorporation on the diversity of policies. Defining precise non-ambiguous policies and providing them to the author of the stream is an intense intellectual activity and requires declarative thinking. Fortunately, an increasing number of tools for interactive markup are available.

The implementation has shown further advantage to model a continuous document stream. The advantages are: quick iterations and continuous integration on all production levels; dynamic optimization of the production process to track market fluctuations; general stream cyclicity, necessary for rapid prototyping and ever shrinking product lifecycles; and asynchronicity and concurrency of service provision wherever there are no explicit constraints.

5. Conclusion

Document management in web-driven decentralized systems needs complex coordination because of the huge information volume, different policies and diversity of authors and users. In this paper, we focused on creating reliable and compact XML document meta-stream.

First the created meta-stream model is presented, that is a collection of XML sections (elements). Every section is individually encrypted and provides references to the document repositories. The access to every different section is classified according to some engineering enterprise or scientific society policy. The benefits for the e-engineering and scientific environment are as follows: (a) anticipation of full bandwidth utilization

of the distributed processing; (b) processing of 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.

Next, the proposed technologies for meta-stream processing are applied. The main processing mode is XML streaming, where the stream is analysed in run-time without need to create the XML tree. The clients of the meta-stream use web arrays to analyse the sections. Web arrays are interactively designed collection of web services, information sources, comments and other instructional information. The classified access to the stream section is manipulated by cryptography-based methods. The parsing of encrypted stream is based on the public-key encryption with keyword search. The body of the section is encrypted by symmetric key. The section data integrity is protected by digital signatures.

References

1. Green, T. J., A. Gupta, G. Miklau, M. Onizuka. Processing XML Streams with Deterministic Automata and Stream Indexes. – *ACM Transaction on Computational Logic* December 2004, No.4.

2. Halle, S., R. Villemaire. Runtime Monitoring of Web Services Choreographies Using Streaming XML. Proceedings of SAC 2009, March 8-12, Honolulu, Hawaii, 2009.

3. Zhang, W., R. van Engelen. A Table-Driven Streaming XML Parsing Methodology for High-Performance Web Services. Proceedings of the IEEE International Conference on Web Services, 2006, 197-204.

4. Freeman, E., D. Gelernter. Document Stream Operating System. US Patent App. 11/607,099, 2006.

5. Ostrovsky, R., W. Skeith. Private Searching on Streaming Data. – *Journal of Cryptology*, 20, October 2007, Number 4, 397-430.

6. Boneh, D., G. Crescenzo, R. Ostrovsky. Public Key Encryption with keyword Search. Advances in Cryptology - EUROCRYPT 2004, Lecture Notes in Computer Science, 3027, 2004, 506-522.

7. W3C, XML Signatures. <http://www.w3.org/TR/xmldsig-core/>.

8. Georgiev, Ivo, Iliya Georgiev. P2P Computing with Web Arrays. Proceedings of the Web XTech 2007 Conference The Ubiquitous Web, 15-18 May, Paris, 2007.

9. Peak, R., J. Lubell, V. Shrinivazan, and S. Waterbury. Step, XML, and UML: Complementary Technologies – *Journal of Computing and Information Science in Engineering*, 4, 2004, 379-390.

10. NIST, Process Specification Language (PSL*)*.
<http://www.mel.nist.gov/psl>, 2008.

11. Bayardo, R., V. Josifovsky, D. Gruhl, J. Myllymaki. An Evaluation of Binary XML Encoding Optimizations for Fast Stream Based XML Processing. Proceedings of the WWW Conference, New York, May 17-22, 2004.

Manuscript received on 14.09.2009

Iliya Georgiev *received M.S. degree from the Technical University of Sofia and Ph.D. from the Electrotechnical University „LETI“ of St. Petersburg. Since 2000 he has been Professor at the Metro State College of Denver, USA. Before that, he had 4 years in the software industry in the USA, 7 years as Professor at the Technical University of Sofia, and 20 years research and development at the Central Institute for Computer Technologies. His recent scientific interests include e-science and e-engineering computing, distributed architectures, multimedia and computer aided engineering.*

Contacts:
The Metropolitan State College of Denver
Department of Mathematical and Computer Science
Campus Box 38, P.O. Box 173362, Denver, CO 80217-3362, USA
email*: gueorgil@mscd.edu, ilgeorg@yahoo.com*
URL: http://clem.mscd.edu/~gueorgil/