

ITW – An Architecture based on Distributed Web Components for Multimedia Resource Discovery

Sabin Corneliu Buraga

Faculty of Computer Science

“Al. I. Cuza” University of Iași, Romania

`busaco@infoiasi.ro`

`http://www.infoiasi.ro/ busaco/`

Abstract

The paper presents a multi-platform and multi-language architecture used to discover (multimedia) resources. This platform, called *ITW*, is based on different distributed Web components, such as Web services and Web agents, exploiting the relations between Web sites' resources. All involved information within the system is XML-based, by using different languages for expressing associated metadata and temporal relationships established between multimedia resources available on Web.

1 Introduction

The actual World-Wide Web space is primary composed on pages (markup documents) with information in the form of natural language text and multimedia intended for humans to read and understand. Computers are used mainly to render this hypermedia information, not to reason about it. Information retrieval has become ubiquitous with the WWW's development and information needs no longer to be intended for human readers only, but also for machine processing, enabling intelligent information services, personalized Web sites, and semantically empowered search engines – this is the seminal idea of the *Semantic Web* [24].

Agent paradigm is one of the promising technologies for information retrieval in general and for multimedia resource discovery in particular. Another emerging technology is denoted by *Web services*, collections of operations that are network-accessible through standardized XML messaging.

The paper will present *ITW* – a multi-language distributed platform used for multimedia resource discovery, using a hybrid architecture that uses software agents, Web services, and other entities such as CGI (Common Gateway Interface) scripts. The ITW system exploits the temporal relations established between Web sites’ resources and uses a RDF/XML-based model for semantic representation of metadata and additional information that involves time.

2 ITW Internal Architecture

The main goal of ITW system is to offer a heterogeneous interoperable infrastructure [18], based on Web components, for multimedia resource discovery. Through a Web portal-like interface, the user will be able to formulate complex queries that involve time. The information and the associated RDF metadata offered by ITW system will be stored on independent servers. Even if one server is shutdown, the system will be able to continue its execution, providing the same capabilities. The system is flexible enough to interrogate – “on-the-fly” – newly plugged-in running services.

The key idea in ITW’s design rationale process is to offer a flexible multi-language and multi-platform architecture for (multimedia) resource discovery based on *semantic information* associated with the resources.

In the discovery process, one of the important issues is to deal with time. Using a RDF-based model presented in section 2.1, the software agents involved in the process of resource discovery will be able to reason about the spatio-temporal relations established between certain resources localized in different Web sites. Of course, our approach can be applied to (semantic) service discovery, in the context of Grid computing [26].

The general architecture of the ITW system consists of two main entities (see figure 1):

- *ITW agents* – their role is to effectively discover distributed multimedia resources stored on different sites; these agents are intended to be implemented within a multi-agent system, such as *Omega* [15] or *GAA* [33];

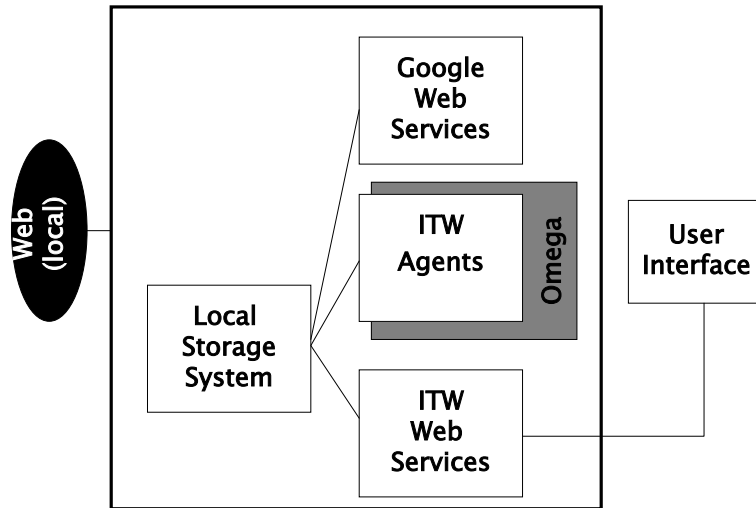


Figure 1: ITW Internal Structure

- *ITW Web services* – their role is to give desired information about the resources and the access to these resources; the Web services can be invoked by different software applications (e.g., user agents – such as Web browsers or special clients –, other agents or Web services).

2.1 ITW Web Agents

The ITW Web agents are intended to be developed within a multi-agent platform, called *Omega* [4, 15], a framework that offers an addressing space for the Web objects and a mechanism for remotely accessing the Web distributed resources (that can be viewed as objects).

To enable the flexible querying and accessing mechanisms about the distributed Web resources, *Omega* offers a facility for serialization – in an independent manner – the data and metadata (objects) processed by the system. For serialization, we adopt XML Schema [25] and SOAP (Simple Object Access Protocol) [27] – for details, see [4].

Additionally, for each object, different metadata constructs are attached to specify several semantic properties [2], [14] and [15]. These descriptions are written in RDF (Resource Description Framework) [29].

The agents of the *Omega* system have the following tasks to be accomplished in their activity of discovering multimedia resources on Web:

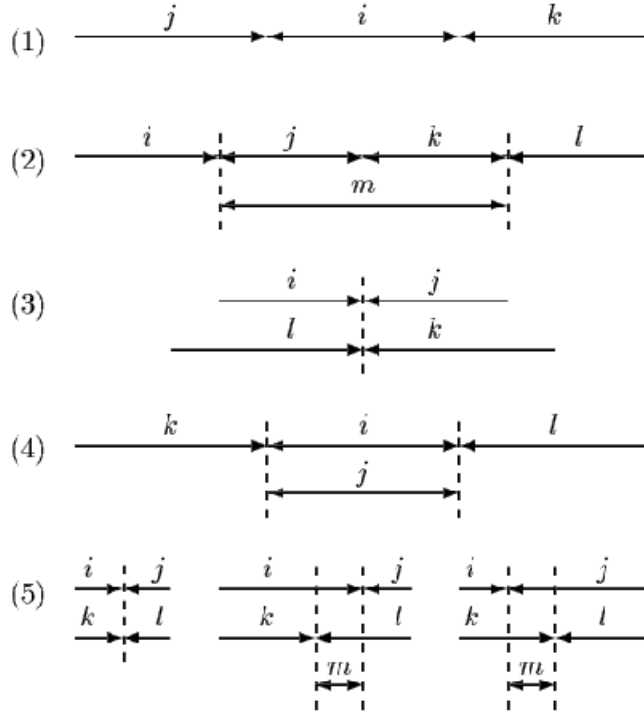


Figure 2: Axiomatization of time periods

- using different XML constructs expressed in our defined *XFiles* [12] and TRSL (Temporal Relation Specification Language) [14] languages, for each Web resource a RDF document is generated (see section 2.1.1).

The *XFiles* constructs are used to retain all important metadata that can be associated with a Web resource: location, type (e.g., XHTML page, JPEG image, VRML virtual 3D world, etc.), owner, method of access, timestamp of last modification and so on. For more details, see [11] and [12].

The TRSL constructs are used to store temporal information regarding the relationship between two resources. The ITL (Interval Temporal Logic) [5, 6, 7] model is adopted. Given any two temporal intervals, there are thirteen mutually exclusive relationships. To model these relations, one primitive relation *Meets* is introduced. Intuitively, two periods m and n meet if and only if m precedes n , yet there is no time between m and n , and m and n do not overlap. The axiomatization of time periods is presented in figure 2, where i , j , k , l , and m are logical variables restricted to time periods.

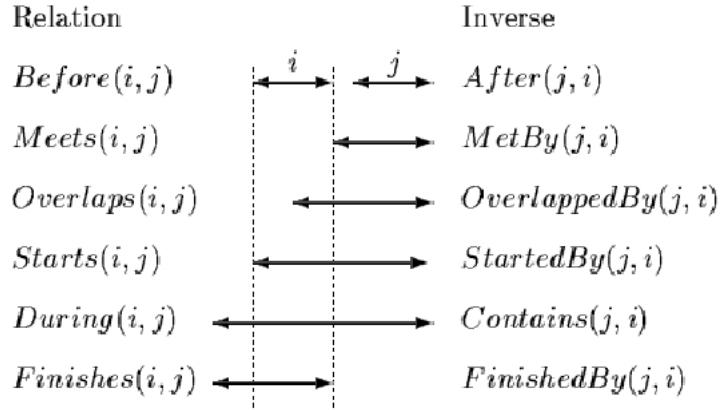


Figure 3: The possible relations between time periods (equality not shown) [8]

A complete range of the thirteen intuitive relationships that could hold between time periods is defined (see [5, 6]). For example, one period is before another if there exists another period that spans the time between them, for instance:

$$Before(i, j) \equiv \exists m : Meets(i, m) \wedge Meets(m, j) \quad (1)$$

Figure 3 illustrates each of these relationships (the equality relation is not shown). We can use the following symbols to stand for commonly-used relations and disjunctions of relations [7]:

$$\begin{array}{ll}
Meets(i, j) & i : j \\
Before(i, j) & i \prec j \\
During(i, j) & i \sqsubset j \\
Before(i, j) \vee Meets(i, j) & i \prec: j \\
During(i, j) \vee i = j & i \sqsubseteq j
\end{array}$$

Between two periods of time, a *Disjoint* relation can be established: two intervals are *disjoint* if they do not overlap in any way. We can write this as “ $i \bowtie j$ ” and define it by $i \bowtie j \equiv i \prec: j \vee j \prec: i$.

A period can be classified by the relationships that it can have with other periods of time. A period that has no sub-periods can be called a *moment* and a period that has sub-periods, an *interval*. Also, a notion of *time point* is defined by a construction that specifies the beginning

and ending of periods (moments and points are distinct). A *discrete time model* can be given, where periods map to pairs of integers $\langle I, J \rangle$, where $I < J$. Moments correspond to pairs of the form $\langle I, I + 1 \rangle$, and points correspond to the integers themselves. A similar model build out of pairs of real numbers does not allow moments.

Using ITL model, certain specific XML-based constructs are defined. These constructs are elements and attributes of the *TRSL* (*Temporal Relation Specification Language*), in order to express the relations *Before*, *Meets*, *Overlaps*, *Starts*, *During*, and *Finishes* that can be established between the last modification timestamps of the Web resources (see the example from section 2.1.1).

These temporal relationships can help to determine the dynamics of that Web sites' content and can be used, among others, by the ITW Web agents in multimedia resource discovery activity. The updating or querying actions can depend on the temporal relations that can be expressed by TRSL and RDF constructs. The TRSL language allows intervals of time (beginning and ending of periods) and, of course, moments.

For each time relation, TRSL offers an element that corresponds to a specific relation (e.g., `<Meets>` element for *Meets* relation). The beginning and ending of time periods are denoted by `begin` and, respectively, `end` attributes that are inspired by Synchronized Multimedia Integration Language (SMIL) [9]. Also, TRSL defines the `dur` attribute for specifying a known or predictive time period (this can allow Web agents to reason about different actions that may need to be performed). The source and destination (viewed as operands) of a temporal relation can be expressed by RDF constructs, too.

The syntax (specified by an XML schema) and the semantics of our defined TRSL language are detailed in [14]. An alternative to TRSL is TimeML [28], a language that is manily focused on expressing time events.

- using RDF constructs regarding the temporal relations, one important goal to achieve is to mentain these relations (e.g., if a Web resource is in relation *Before* with another one, the agent will try to mentain this relation, by checking on regularly basis the metadata associated with the involved resources). For this relation, the user could specify certain actions to be executed by using TRSL constructs.
- the internal behaviour of the multi-agent environment is modeled by

BDI_{CTL}^K logic [31, 35]. This model uses the *belief*, *desire* and *intention* notions associated to the individuals of the agent population that is executed within the multi-agent system. For more details about the BDI agents model, see [10, 32],

To assure a proper communication language between agents and a flexible deployment of the environment, we define an XML-based agent-communication language over SOAP messages (for details, see [15]).

2.1.1 Example

An example of a metadata XML-based file associated to a video resource (e.g., a film trailer or a speech excerpt) follows:

```
<?xml version="1.0" ?>
<rdf:RDF>
  <rdf:Description
    rdf:about="http://www.site.org/Video.mpeg">
    <!-- temporal information -->
    <temporal:link
      begin="2003-05-09T17:15:00"
      end="2004-05-09T10:00:00"
      linkType="temporal">
      <temporal:Before dur="7D">
        <rdf:Description
          about="http://mirror.org/video/1.mpeg">
          ...
        </rdf:Description>
      </temporal:Before>
    </temporal:link>

    <!-- metadata -->
    <meta:Properties>
      <meta:Location
        ip="193.231.30.225" port="80">
        www.infoiasi.ro
      </meta:Location>
      <meta:Auth type="basic">
        UsersGroup
      </meta:Auth>
      <meta:Owner>
```

```

    <meta:Login>
      web
    </meta:Login>
    <meta:Password method="md5">
      ...
    </meta:Password>
  </meta:Owner>
</meta:Properties>
</rdf:Description>
</rdf:RDF>

```

The *XFiles* and TRSL syntactic elements are denoted by `meta` and, respectively, `temporal` namespaces. The RDF assertions are included into the `rdf` namespace – we omit the actual URI (Uniform Resource Identifiers) of these namespaces.

The document defines a *Before* relation between two multimedia resources. The video resource denoted by `http://www.site.org/Video.mpeg` is considered the original one, but a copy is located at another Web address (URI): `http://mirror.org/video/1.mpeg`. Each update of the original video implies another update of the copy after 7 days (in order to maintain the *Before* temporal relation).

Metadata information includes the address of the storage machine, the owner of the resource and the authentication mechanism, by using *XFiles* constructs.

Of course, any other useful information – such as DCMI (Dublin Core Metadata Initiative) or RSS (Rich Site Summary) elements [39] – can be specified.

2.2 ITW Web Services

The Web services part of the application consists of the following distributed components (see also figure 4):

- a number of m local Web services to provide information about the resources stored on the local Web (i.e. the intranet, the extranet or the public Web site of an organization);
- a number of n external Web services, developed by third-party organizations (e.g., Google’s public Web service – see section 3).
- a local storage system, in order to store RDF documents that contains metadata and relations between found resources.

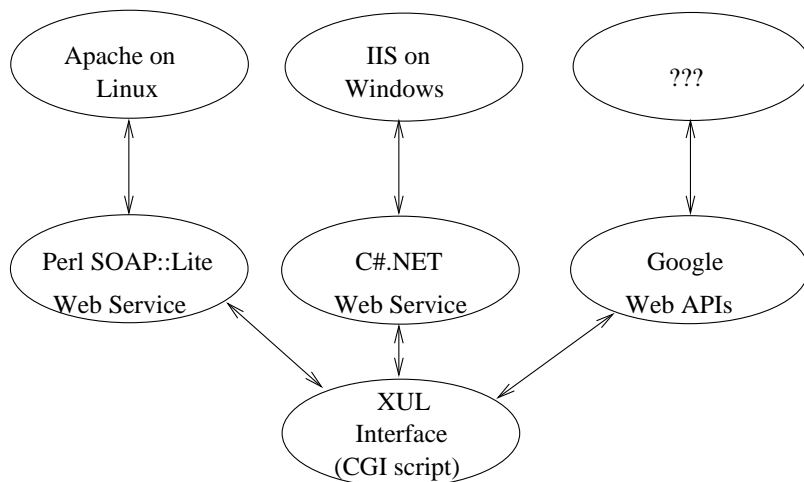


Figure 4: ITW Actual Architecture

The physical localization and execution of the $(m + n)$ Web services are transparent for the final user. These Web services can be considered as one entity (very similar with a computational Grid [26]) by the ITW system.

If the IP addresses of the local Web services are fixed, then these services can be independently invoked by other external entities. Of course, the information exchanged by Web services and their clients is stored as Simple Object Access Protocol (SOAP) messages [41]. The ITW system provides the Web Services Description Language (WSDL) [41] descriptions of the local designed Web services and uses the available WSDL documents in order to invoke external Web services.

The ITW Web services uses RDF documents about the discovered resources. These document are automatically generated by the ITW agents presented in section 2.1 and are stored within the storage system.

A more simple approach is to use a Web robot, that regularly runs to update this information about the local multimedia resources of a certain Web site.

2.3 ITW User-Interface

The user-interface part of the ITW system consist of one CGI script that uses Extensible Markup Language (XUL) [30] in order to provide a flexible query user-interface. The XUL language provides various types of widgets (components) used to build complex graphical Web interfaces. These widgets are similar with the current approaches used in graphical user interface

development environments (e.g., Borland Delphi/Kylix, Glade, or Microsoft Visual Studio .NET).

To define the user interaction on different controls of the Web interface, a protocol must be adopted. One of the best solutions is to adopt the XUP (Extensible User-interface Protocol) [41], an XML-based language developed for the purpose of communicating user interface events in a platform- and display-neutral format.

The queries formulated by the user are stored into an adapted version of our defined WQFL (Web Query Formulating Language) [19] format, in order to specify additional information regarding the search (e.g., relation with another resource, method of access, resource type, etc.).

In [16, 17], we investigate the use of Perl-like regular expressions to provide a more flexible way to specify a query, in the context of structural and semi-structural search activity on the WWW space.

3 Actual Implementation of the ITW System

The ITW's actual implementation – in prototyping stage – includes the following components [18]:

- one CGI script written in Perl that generates the XUL constructs and functions as a generic client for the involved discovery Web services; the XUL documents are supported by Mozilla/Netscape browser. Using Extensible Stylesheet Language (XSL) [1] transformations, the XUL document can be transformed in other XML-based languages, such as XHTML or WML, in order to support any user-agent, following the idea presented in [13];
- two ITW Web services available on Linux and Windows platforms; one Web service is implemented in Perl [20, 34] by using SOAP::Lite module [36] and Apache Web server [42]; the other is implemented in C# language on Microsoft .NET Framework [38], using IIS (Internet Information Services) Web server;
- one external Web service freely provided by Google [37] in order to discover world-wide multimedia resources.

The implemented elements were tested on Windows XP – using .NET Framework 1.1 – and different Linux distributions (RedHat 9.0 and Mandrake 9.1) – using Perl 5.8 and Apache 2.0. For XML processing, the SAX (Simple API for XML) functionalities of the XML::Parser Perl module were

used. The access to Google's Web service was given through a special unique key and the WSDL file provided by the search engine [37]. The ITW agents are implemented in C++ language, under Visual Studio 6.0 programming environment.

The actual storage system consists of two freely available relational database servers: MySQL (under Windows) and PostgreSQL (under Linux).

The ITW system can be considered as a distributed platform for discovering multimedia information within the intranet of an organization, such as an enterprise Web portal – see [22] and [23].

4 Conclusion and Further Work

A multi-language architecture, named *ITW*, for multimedia resource discovery was presented. The ITW system consists of an ensemble of Web services, software agents and other components (described in section 2) and uses a RDF/XML model (introduced in [14]) for semantic representation of metadata and temporal relations between Web (multimedia) resources. This model is based on ITL logic [6, 7].

The actual implementation was focused on the development of the local Web services – deployed on Windows and Linux platforms. For remote access to resources, the Google's search Web service is used (see section 3).

Instead of the XUL user-interface, another more flexible approach is given by the XSL (Extensible Stylesheet Language) [41] transformations, that can be applied on the server-side (e.g., using the Apache AxKit [42] facilities) in order to offer an optimal interface for each user-agent (Web client or Web service).

To gain processing speed, in the next version of the system the XML/RDF generated constructs can be stored within an XML native database, such as Apache XIndex [42]. Another important aspect is to cache the results obtained from an interrogation in order to satisfy the client requests with the highest level of quality. To accomplish this goal, further work will be focused on serving differentiated content by using a tiered XML-based database storage service [21].

Also, we intend to integrate the ITW system into the *tuBiG* Grid-like architecture [3], to place the project in the context of Grid computing [26, 40].

Acknowledgement

Some of the results presented in this paper represents the fruit of excellent scientific collaboration between the author and *Lenuța Alboai*e and *Sînică*

Alboaie (Institute of Theoretical Computer Science, Romanian Academy – Iași branch) and *Petrică Găbureanu* (graduate of the Faculty of Computer Science, “Al. I. Cuza” University of Iași, Romania).

We express our gratitude to *Marcin Paprzycki* (Computer Science Department, Oklahoma State University, USA).

References

- [1] S. Adler *et al.*, *Extensible Stylesheet Language (XSL) Version 1.0*, W3C Recommendation, Boston, 2001: <http://www.w3.org/TR/xsl/>
- [2] S. Alboaie, S. C. Buraga, L. Alboaie, “An XML-based Object-Oriented Infrastructure for Developing Software Agents”, *Scientific Annals of the “Al. I. Cuza” University of Iași*, Computer Science Series, tome XII, 2002
- [3] L. Alboaie, S. C. Buraga, S. Alboaie, “*tuBiG* – A Layered Infrastructure to Provide Support for Grid Functionalities”, in M. Paprzycki (ed.), *Proceedings of the 2nd International Symposium on Parallel and Distributed Computing*, IEEE Computer Society Press, 2003 (to appear)
- [4] S. Alboaie, S. C. Buraga, L. Alboaie, “An XML-based Serialization of Information Exchanged by Software Agents”, *Proceedings of the 7th World Multiconference on Systemics, Cybernetics and Informatics – SCI 2003*, Orlando, USA, 2003
- [5] J. Allen, “Time and Time Again: The Many Ways to Represent Time”, *International Journal of Intelligent Systems*, 6 (4), 1991
- [6] J. Allen, “Maintaining Knowledge about Temporal Intervals”, *Communications of the ACM*, 26 (11), 1983
- [7] J. Allen, P. Hayes, “Moments and Points in an Interval-based Temporal Logic”, *Computational Intelligence*, 5 (4), 1989
- [8] J. Allen, G. Ferguson, “Actions and Events in Interval Temporal Logic”, Technical Report 521, The University of Rochester, New York, 1994
- [9] J. Ayars *et al.* (eds.), *Synchronized Multimedia Integration Language (SMIL 2.0)*, W3C Recommendation, Boston, 2001: <http://www.w3.org/TR/smil20>

- [10] F. Brazier *et al.*, “Modelling Internal Dynamic Behaviour of BDI Agents”, in P. Schobbens, A. Cesta (eds.), *Proceedings of the Third International Workshop on Formal Models of Agents – MODELAGE’97*, Lecture Notes in Artificial Intelligence – LNAI, Springer-Verlag, 1997
- [11] S. C. Buraga, “A RDF Description of Distributed File Systems”. *Scientific Annals of the “Al. I. Cuza” University of Iași*, Computer Science Series, tome IX, 2000
- [12] S. C. Buraga, “A Model for Accessing Resources of the Distributed File Systems”, în D. Grigoraș *et. al* (eds.), *Advanced Environments, Tools, and Applications for Cluster Computing. NATO Advanced Research Workshop, IWCC 2001 – revised papers*, Lecture Notes in Computer Science – LNCS 2326, Springer-Verlag, Berlin, 2002
- [13] S. C. Buraga, “An XML-based Approach in Designing and Building of Web User-Interfaces”, in I. Ivan, I. Rocșa (eds.), *The Proceedings of the 6th International Conference on Economic Informatics*, INFOREC Printing House, București, 2003
- [14] S. C. Buraga, G. Ciobanu, “A RDF-based Model for Expressing Spatio-Temporal Relations Between Web Sites”, in *Proceedings of the 3rd International Conference on Web Information Systems Engineering – WISE*, IEEE Computer Society Press, 2002
- [15] S. C. Buraga, S. Alboaie, L. Alboaie, “An XML/RDF-based Proposal to Exchange Information within a Multi-Agent System”, in D. Grigoraș *et al.* (eds.), *Proceedings of NATO Advanced Research Workshop on Concurrent Information Processing and Computing*, IOS Press, 2004 (to appear)
- [16] S. C. Buraga, M. Brut, “A Proposal for a Web Structural Search Language Based on XML Technologies”, *Scientific Annals of the “Al. I. Cuza” University of Iași*, Computer Science Series, tome X, 2001
- [17] S. C. Buraga, M. Brut, “Different XML-based Search Techniques on Web”, *Transactions on Automatic Control and Computer Science*, vol. 47 (61), No. 2, Politehnica Press, Timișoara, 2002
- [18] S. C. Buraga, P. Găbureanu, “A Distributed Platform based on Web Services for Multimedia Resource Discovery”, in M. Paprzycki (ed.), *Proceedings of the 2nd International Symposium on Parallel and Distributed Computing*, IEEE Computer Society Press, 2003 (to appear)

- [19] S. C. Buraga, T. Rusu, “An XML-based Query Language Used in Structural Search Activity on Web”, *Transactions on Automatic Control and Computer Science*, vol. 45 (59), No. 3, Politehnica Press, Timișoara, 2000
- [20] S. C. Buraga, V. Tarhon-Onu, Ș. Tanasă, *Web Programming in bash and Perl* (in Romanian), Polirom, Iași, 2002
- [21] H. Chen, A. Iyengar, “A Tiered System for Serving Differentiated Content”, *World-Wide Web: Internet and Web Information Systems Journal*, no. 6, Kluwer Academic Publishers, 2003
- [22] M. Cioca, S. C. Buraga, “New Tools for Human Resource Management in e-Business: Combining UML Language, Reference Architectures and Web Programming”, *IEEE International Conference on Industrial Informatics – INDIN’03 Proceedings*, Banff, Alberta, Canada, 2003
- [23] M. Cioca, S. C. Buraga, “Instruments and Web Technologies for Implementing Architectures and Integration Informatics Systems in Virtual Enterprises”, *CD-ROM Proceedings of the 3rd International Conference on Research and Development in Mechanical Industry – RaDMI 2003*, Herceg Novi, Montenegro Adriatic, 2003
- [24] J. Davies, D. Fensel, F. van Harmelen (eds.), *Towards the Semantic Web*, John Wiley & Sons, England, 2003
- [25] D. Fallside (ed.), *XML Schema*, W3C Recommendation, Boston, 2001: <http://www.w3.org/TR/xmlschema-0/>
- [26] I. Foster, C. Kesselman (eds.), *The Grid: Blueprint for a Future Computing Infrastructure*, Morgan Kauffmann Publishers, 1999
- [27] C. Gorman, *Programming Web Services with SOAP*, O’Reilly & Associates, 2001
- [28] R. Ingria, J. Pustejovsky, *TimeML Specification 1.0*, 2002: <http://time2002.org>
- [29] O. Lassila, R. Swick (eds.), *Resource Description Framework (RDF) Model and Syntax Specification*, W3C Recommendation, Boston, 1999: <http://www.w3.org/TR/REC-rdf-syntax>
- [30] I. Oeschger, *XUL Programmer’s Reference Manual*, Mozilla.Org: <http://www.mozilla.org/xpfe/Xulref.zip>

- [31] A. Rao *et al.*, “Formal Methods and Decision Procedures for Multi-Agent Systems”, Technical Report no. 61, Australian Artificial Intelligence Institute, 1995
- [32] A. Rao, M. Georgeff, “BDI Agents: from Theory to Practice”, *Proceedings of the 1st International Conference on Multi-Agent Systems – IC-MAS’95*, San Francisco, 1995
- [33] J. Vidal, P. Buhler, “A Generic Agent Architecture for Multiagent Systems”, Technical Report no. 011, University of South Carolina, Swearingen Engineering Center, 2002
- [34] L. Wall *et al.*, *Programming Perl* (Third Edition), O’Reilly & Associates, Cambridge, 2000
- [35] M. Wooldridge, N. Jennings, “Intelligent Agents: Theory and Practice”, *Knowledge Engineering Review*, 1995
- [36] * * *, *Comprehensive Perl Archives Network*:
<http://www.perl.com/CPAN/>
- [37] * * *, *Google APIs*: <http://www.google.com/apis/>
- [38] * * *, *Microsoft .NET Framework*, Microsoft Corporation, 2003:
<http://www.microsoft.com/net/basics/framework.asp>
- [39] * * *, *Purl Organization*: <http://www.purl.org/>
- [40] * * *, *Semantic Grid Project*: <http://www.semanticgrid.org/>
- [41] * * *, *World Wide Consortium’s Technical Reports*, Boston, 2003:
<http://www.w3.org/TR/>
- [42] * * *, *XML Apache Project*, 2003: <http://xml.apache.org/>