

Selecting Services for Web Applications: The Open Hypermedia Case

Nikos Karousos^{1,2}, Manolis Tzagarakis¹, Nicolas Koumbarou²

¹ Research Academic Computer Technology Institute,
15600, Rion, Greece
{karousos,tzagara}@cti.gr

² Department of Computer Engineering & Informatics,
University of Patras, 26500 Rion Greece
koumparu@ceid.upatras.gr

Abstract. This paper proposes a framework for enabling the usage of Hypermedia services by web developers, and studies the working progress that provides Hypermedia services to the Web. It investigates and analyzes the requirements that will provide web application developers with the ability of easily inserting Hypermedia functionality into their applications. A Service Discovery Mechanism for finding and using services is defined, and solutions for increasing the usage of Hypermedia systems by web developers are proposed.

1 Introduction

World Wide Web is a huge collection of information. Thus, the need for tools able to efficiently manage, collect and organize this huge amount of information is constantly growing. Issues like searching, organizing, comparing, commenting and projecting the information, have gained significance in the past years. As long as the usage of the Web is rising, the need for effective information management systems proportionally increases. Web developers, who create applications that require services like information organization, authoring, versioning, annotation, sophisticated backtracking, information restructuring and similar services, can be helped by the usage of Hypermedia services into their applications.

Hypermedia systems are ‘B-class’ or ‘C-class’ tools [12], according to Engelbart’s definition. These systems can provide services that are useful to 3rd party applications including web projects. This will enrich business, scientific, web engineering, and personal applications with hypermedia functionality [9]. Although web engineers have already paid great attention to the web application development procedure [10], the Open Hypermedia Community has not approached yet, either to the standardization of a methodology, or to the development of tools that can drive web developers to a quick and simple way to embody Hypermedia functionality into their web applications. Moreover, Open Hypermedia Systems has not reached a satisfactory level of publicity and the Hypermedia service discovery issue still remains unaddressed.

This paper proposes a framework¹ for enabling web developers to use Hypermedia services into their applications. By bringing together the successfully applied, service oriented web engineering techniques and the Open Hypermedia Systems' services, we study the design of both a Hypermedia Service Discovery Mechanism and a set of tools, in order to create a web developer support framework. Consequently the web developers' needs and requirements concerning searching, selecting and using hypermedia services, are properly addressed.

In the rest of this paper, we focus on the web development process and we try to analyze what are the needs of web developers, regarding the use of services. Next, we take a closer look into the relationship between Hypermedia systems and Web Development (chapter 2). In chapter 3, tools for discovering and using hypermedia services, towards the enhancement of web application with Hypermedia functionality, are proposed. Finally, we discuss the future work and we conclude.

2 Methodology and Tools: a close relationship

Towards establishing a reliable methodology when dealing with Web Applications, the availability of tools turns out to be crucial. When talking about methodology, the issue is not only sufficiency of tools. In particular, a methodology outlines a procedure where the creation of web applications is efficient: i.e. guaranties that the desired result is achieved with the least possible effort every time it is repeated. It is rather questionable whether a sound methodology can be established by having only tools that suffice.

Thus, close to any attempt establishing a methodology, the efficiency of the available tools has to be examined. In the area of Open Hypermedia, methodologies fail mainly, due to the fact that basic tools are missing. Such 'holes' in the development procedure have been rather traumatic for the acceptance of the Open Hypermedia Systems.

In the next section, we outline the issues of developing Web Applications when the Open Hypermedia Case is considered.

2.1 Service Oriented Web Development.

The network-intensive nature of web applications means that these applications rely heavily on services that provide the required functionalities. Although, how exactly service-oriented architectures are used during web application development, is a rather neglected issue, compared to standardized technologies emerged, including Web Services [26] and Simple Object Access Protocol (SOAP) [27].

¹ We define 'framework' as an extensible structure for describing a set of concepts, methods, tools, technologies, and structural changes necessary for a complete application design and development process.

However, hand-on experience with such technologies is still lacking. This is largely due to the fact that convenient infrastructures, for using these technologies are still missing.

We refer to the attempt to create a framework for integrating service-oriented architectures, into the web application development process. The creation of a Web application, such as a Web Portal, a Database Web Application, etc. demands the remote usage, or the integration of foreign services into the specified Web Project. Researchers in the area of Web engineering have already approached this requirement from multiple scientific sides, including system architecture, productivity, economics, security and performance. However, from the developer's perspective, it's worth mentioning both the steps a developer follows whenever he wants to use a service and the requirements that emerge.

When a developer intends to use foreign services to his application, he: a) searches for the appropriate service through the network, b) understands the server interface and the way the server operates, c) implements the required communication level into the application in order to communicate with the remote service and d) encapsulates the adopted service as a set of functions that can be used by his application. The developer's requirements that emerge from this procedure are presented below.

2.2 Developers' requirements

Web developers demand low cost services, which are easy to discover and easy to use/integrate into their applications.

Firstly, a methodology for easy discovering services should be considered. A good example of such a mechanism is the Universal Description, Discovery and Integration of Web Services – (UDDI) [25]. Through UDDI a developer can search into categories or query for a desired service. In many cases, the need of advanced discovery techniques arises, when meta-search and both functional and not - functional client requirements are served [11].

Assuming that a desired service is found, the user (developer) should get information about how the service operates and which is the application interface (API). This automatically generates another requirement, which is, an easy and understandable description, by both the human and the computer, of the service. In particular, the service should carry meta-information that is self-descriptive.

At this point, some widely accepted issues like trust, security reliability are raised as basic requirements towards the efficient operation of the application.

Another requirement demanded by the developer, is the existence of a driver that makes the usage of the service into the application, as simple as possible. An example of such tools is the demo clients that are usually shipped together with the API or the source of some services, as well as the code generators (for example in RMI, CORBA etc.), that automatic create the client/server communication.

Finally, whenever a developer wants to include functionality into his application by using a new service, he demands the simplicity of the functions (of the service) that he calls. Complex functions have a complex result and not understandable service provision for potential users. Another reason, that discourages the developers to use such a

service, is the high maintenance cost, since a simple change of a complicated function will require large amount of changes in the integrated application.

Briefly, a web developer needs to have a support framework which will provide easy-to-find, easy-to-understand and easy-to-use 3rd party services into his applications.

2.3 Open Hypermedia Systems and the Web

From the Open Hypermedia perspective, the World Wide Web and the 3rd party web applications were always targets for hypermedia service provision. The design of both Component Based Open Hypermedia Systems (CB-OHSs) and service oriented systems facilitated those efforts [22]. The Multiple Open Services project [29], was the next step, in which the division was not into structural servers, like CB-OHSs, but into services. Finally, the widely adopted concept of Structural Computing [21], and the corresponding developed systems, like Themis [2] outline this trend. Although, the hypermedia systems improved their interoperability, the unawareness of Web Projects for the provided Hypermedia services, still remains a crucial aspect.

On the other hand the universal popularity of the World Wide Web enforced the Hypermedia researchers to make several integration efforts [1, 7]. Those integration efforts are mainly ad-hoc integration efforts, without using any developer framework, or developer support tool. The lack of a concise methodology makes such integrations error-prone and difficult to maintain. Moreover, in most of them, the whole application interface of the Hypermedia system is provided. In that case the potential user of the system is forced to integrate the whole system into his application, just for one service. Most of the well known Hypermedia systems, like Microcosm [15], with the Distributed Link Service (DLS) [8], DHM [14], with DHM/WWW [13], and Chimera [1, 3] have a web integration solution.

The Babylon Web Service [17] is another approach for providing Hypermedia services into the Web. In this project, the Babylon Taxonomic Hypermedia system is used to provide taxonomic services into the Web, through the development of a Web Service. In addition, another attempt was made for mapping SoFAR system to Web Services with respectful results [4].

2.3 Why Hypermedia systems are not used efficiently in the web development?

It is widely accepted from Hypermedia community, that Hypermedia services didn't have a global usage [23]. One of the main reasons for that, is low publicity. A huge mass of web developers is unaware of both the existence of the Hypermedia systems and the required methodology of adding Hypermedia services into their Web applications. Furthermore, Open Hypermedia Systems do not present themselves in the web in order to enlighten someone who wants to be informed over the Hypermedia area. Unfortunately, the most common way for someone to learn about OHSs is from conferences, publications and other developers' out-of-band.

Another reason is the usually high complexity of Hypermedia systems. Consequently, the provision process of an autonomous OHS service is not an easy hypothesis. A Hypermedia developer cannot simply produce a service with a set of functions and a Hypermedia user (a web developer) cannot understand a hard system that must communicate with it. Both structural operations and multiple perspectives over the same data, increase the complexity of such Hypermedia systems.

In many cases, a web developer who wants to use only one Hypermedia service is indebted to get and have access (and to pay) for all the OHS usability [24]. That, is ought in high connectedness between all components of the Hypermedia system. In this case, the cost and the complexity of the OHS service provision are multiplied. The goal is the user to be provided with simple, flexible, scalable and well-defined services.

Another important issue is that Hypermedia extensions and web integrations are ad-hoc implementations, created by non-standardized methodologies. This implies that there is not a simple and standard way to use a Hypermedia service. Each time a developer who wants to integrate an OHS with the Web, is forced to invent his own techniques. This issue is an emerging important problem in the area of CB-OHSs, where components are dynamically added or even modified, requiring from the developer to make new integration efforts every time. The lack of concise methodology makes such integrations error-prone and difficult to maintain.

As foresaid, it is entailed that Hypermedia community should be focused on developer support. There aren't any established methodologies or mechanisms for finding services in order to help both web developers finding and using Hypermedia services into their applications and Hypermedia developers using web technologies. It is widely accepted that both worlds would benefit from the staple usage of Hypermedia services: the Hypermedia community would create services for wide available usage and the web community would benefit from the enriched structuring facilities offered by typical Hypermedia services.

3 Towards Tools for Discovering and Using Hypermedia Services

Both a discovery mechanism, which is constituted by a standard platform, and a set of tools, in order to enable the enhancement of Web applications with Hypermedia functionality, are proposed. A necessary condition to operate the discovery mechanism is that OHSs and generally CB-OHS should be outfitted with a new feature. This feature is the Hypermedia server ability to describe itself. In that way, Hypermedia systems are equipped with introspection capabilities.

3.1 Service Discovery Mechanism.

Architecture. There are two different approaches for the discovery mechanism: the centralized and the distributed service, discovery architectures.

In the centralized architecture when a server starts operating, it registers to a Hypermedia service directory server. When a potential user wants to find a Hypermedia service in order to integrate it with his web application, he queries the service directory and he waits until service directory responds with a list of services in which the user may find the desired service. It is important to detect if the service directory has a well-known location of services.

In the distributed architecture multiple server directories do exist. In order to discover the information about appropriate servers from a distributed server directory system, each server directory interoperates with others by passing queries. Furthermore, a new approach of peer-to-peer (P2P) server discovery systems, based on a full P2P architecture has increased the information distribution.

Which architecture should be chosen? Initially, it's worth mentioning that service discovery mechanism is strongly dependent on the specific characteristics of servers it works with. Moreover, because no Hypermedia service discovery mechanism has been adopted yet, a simple discovery mechanism should be designed. Thus, firstly a centralized service discovery mechanism should be built in order to fulfill all the required needs of the current Hypermedia systems, with the condition that in the future this mechanism has to be extendable in order to be able to support future transformation to a hybrid approach (a mixture of global and distributed: the model being global, the services being distributed) or to a fully distributed service discovery system.

Introspection Capabilities of OHSs. The term "Introspection Capability" of a Hypermedia system is defined as the awareness of all of its provided Hypermedia services and of the required information for service invocation. More specifically, the system must be aware of its API and the semantic of its operation. Moreover, the system has to answer questions like "who are you? What services can you provide? And how can I have access on this specific operation?"

In order to address introspection ability in the Hypermedia servers, a Hypermedia Service Description Language (HSDL) should be defined. This language can be based on emerging standardized markup languages, such as the Web Service Description Language (WSDL) [28] and the Resource Description Framework (RDF) [19], or it can be a new XML-based ad-hoc language. The basic concept of the Semantic Web [5] is that the data may not be only machine-readable and human understandable, but also machine understandable [20]. Therefore, Hypermedia service description language should have an XML format.

HSDL aims to fully describe a Hypermedia service, and to provide all the necessary information to web developers who want to select and use the Hypermedia Service. A standard structure of HSDL is presented below:

- **Service General Information.** It includes information about the service name, the service provider, the service scope and finally structural specific information such as the Hypermedia domain (e.g. spatial) etc.
- **Service Locating and Accessing Information.** It includes the host and port (or other location information) and the required communication protocol(s) between client and Hypermedia server.
- **Service Interface Information.** It includes analytically descriptions about the set of the available operations that the service supports. These operations are defined as functions that can be called remotely from a client and can return a result. Furthermore the set of the necessary data types that is passed or returned as function parameters is defined.
- **Service Behavior Information.** This type of information refers to the behavior of the Hypermedia server, after each request, and the dependences between the server operations.
- **Service Comments.** When developers read the HSDL while trying to understand the way the server operates, the service commentary is useful.

Finally, the HSDL should be extendable and open in order to be able to augment new language characteristics and specifications for supporting specific servers or servers of new Hypermedia domains. In addition the description of the naming methodology that every OHS use has to be also supported of the HSDL.

The current research on both the Web Services and the WSDL can be the first step for the creation of the HSDL. The extension and the alteration of the WSDL could be a very interesting task, aiming to the definition of the first version of the HSDL.

Hypermedia Service Registry. When a Hypermedia service is ready to operate, then the system administrator or the OHS by itself should: a) fill a registry record with some required fields that describe the service and b) register the service by passing a request with the registry record to the service directory.

A service registry should consist of a set of fields that are able to describe the service and can be queried from Hypermedia or Web users who targets to discover an appropriate service. A list of the most important required fields in the registry record is presented below.

Table 1. List of the most important fields in the registry record.

Registry Fields
<i>Owner Information</i>
Owner Name, Address, URL, etc.
<i>OHS Information</i>
OHS Name, OHS Description
OHS Access Point
<i>Domain Information</i>
Domain Type
<i>Service Information</i>

Service Name, Service Description
Required Communication Protocol
Service Status
Service Location
HSDL Location
<i>Registry Information</i>
Registry Unique Name / ID
Register Date, Duration, Description, etc.

Except from the specification of the data that will be provided in the registry record, some extra metadata concerning the service authentication, the license policy etc. should be defined. These metadata can be clustered into semantic chapters, e.g. metadata of the service for the technical issues, for the economic issues, for the usage or for semantic issues.

The periodic checking of the advertised Hypermedia servers' status is a significant issue that must be addressed for the service discovery mechanism. If an already registered Hypermedia service does not work, the corresponding registry record should be deleted from the service directory or should not be available to the public.

Hypermedia Service Query Language. A Hypermedia service query language should be defined. Using this language, a Web developer should be able to search and locate services. This language could be based on a common used communication protocol and must be used by both humans and applications. In that case, the Hypermedia directory system will be a service by itself that will respond to searching requests in a standard common protocol. The query could be a list of Hypermedia service registries that matches all the requested criteria.

Furthermore, a set of effective ranking algorithms should be applied to the match-making procedure of the mechanism, in order for the service to result the optimum appropriate Hypermedia services for the requests.

Hypermedia Service Discovery Usability. When a creation of the service discovery mechanism becomes true, a Hypermedia service directory interface should be created, in order to help web developers querying easily for a service. Hypermedia service directory interface should operate as a visual representation tool, which classifies the services into categories and provides them to the public through tree-views with multiple perspectives.

Also, such a tool will help Hypermedia community to inform easily the whole web community about the provided Hypermedia services. An optional requirement is the existence of an available demo for each published service. With this way, the user will be able to see exactly what request a published Hypermedia server can serve and test if a service really does what he expects.

3.2 Discovering Hypermedia services: a step-by-step example.

A standard methodology should be followed in order to present discovery results to Hypermedia or Web application users who aim to search for an appropriate service.

1. Users search for a Hypermedia service in Hypermedia Service Directory Interface by filling in required fields and requesting the Discovery Server.
2. Using the Hypermedia Service Query Language over the services information (defined by the HSDL) the discovery server searches and locates appropriate Hypermedia services.
3. A list of appropriate Hypermedia services is returned as a responding result (i.e. in XML format) to the Discovery Interface. The result is reformatted and then it is presented to web developers.
4. By selecting a Hypermedia service, information about the service (general, for locating and accessing, for the interface and its behavior) should be provided.
5. In order to understand, to communicate and finally to use the selected service users can download the HSDL service definition file and a demo-client application.
6. After the successful communication establishment, Hypermedia service processes all the web application requests and sends the appropriate response back to the requester.

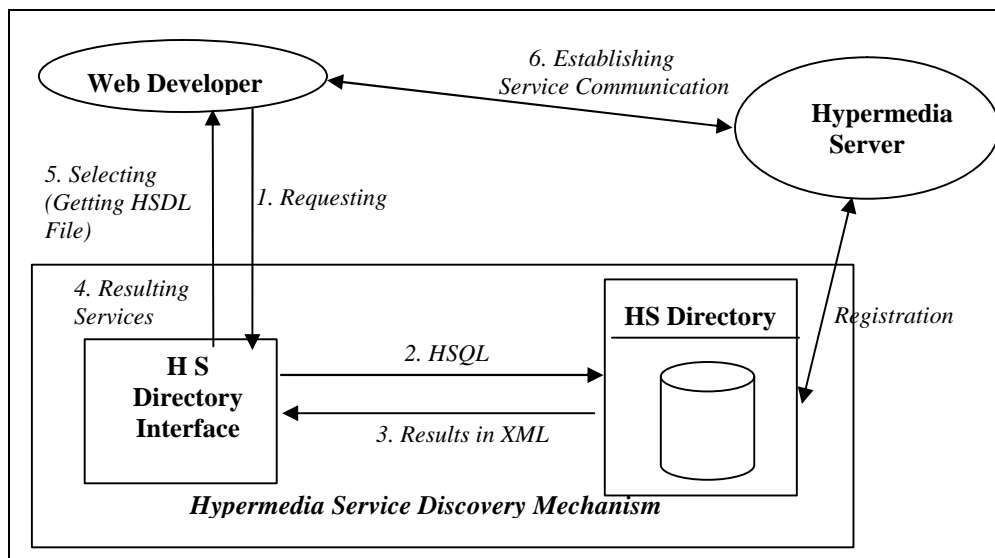


Fig. 1. Discovering hypermedia services.

3.3 Extra Tools

Except for the discovery mechanism, some extra tools should be created, in order to provide new benefits to the proposed mechanism.

Hypermedia Service Exploring Tools. Giving the opportunity to have access to OHS, explore for available services of a particular Hypermedia system and navigate all Hypermedia systems for available services, Web developers will have a global view of the Hypermedia entity. Therefore a visual Hypermedia Exploring Tool should be useful to developers. In addition, this will contribute to the understandability of the entire Hypermedia system.

Communication Protocol Awareness. Web developer should feel confidence about the communication protocol that has to be used, in order to communicate with specific Hypermedia server and use Hypermedia services. Thus, approaches that utilize widely accepted communication infrastructures such as SOAP should be accepted or the ability of dynamically locating and loading protocols –at run time - should be investigated. Such dynamic behavior can already be witnessed in applications such as browsers and video players that automatically download plug-ins. As a result, when the discovery mechanism returns a list of matched registry records, for each selected service that uses a well-known communication protocol, the required protocol access information will also be provided.

Automatic Client Skeleton Creation. Tools for the automatic generation of the client communication layer should be developed. These tools will serve a wide range of devices and platforms. When a standardized Hypermedia service description language takes place, some sophisticated and effective automatic client creation tools should be created.

4. Current status and future work

The developer support in OHSs has been the main target of our work for the past two years [17, 18, 16]. Specifically the definition of both the service registry record and the HSDL (with respect to structural computing principles) are our current tasks. The following goal is to design and implement the first hypermedia service registry directory. Also, the definition of a simple communication protocol for the directory should be made here. Applying Web Service in association SOAP as a first step seems considerably interesting. The Web-based directory interface creation that will publish the registry directory to the web should be implemented next, in order to create the first on-line OHSs' service directory. In addition, a notable approach of the discovery

mechanism is the p2p based architecture in relation to the currently addressed p2p based hypermedia systems [6].

5. Conclusions

As long as the volume of the information distributed and used in the Internet increases, the need for useful and easy-to-use 3rd party services in many Web Projects is growing up. The Hypermedia community can help and offer services that emerge as a result from its systems and researches. For the efficient provision of such services, a useful Developer Support Framework should be developed with characteristics like Hypermedia Service Discovery Mechanism, Hypermedia Service Introspection and a set of tools and standards focused to the support of the Developer. Moreover, the effort for a wider broadcast of the whole action of the Hypermedia research will ease the audience's information procedure. Concluding, rethinking the design of Hypermedia systems from the developer's perspective, can both help the facilitation of the web development procedure and the growth of the OHSs' usage.

Acknowledgments

The authors would like to thank Ippokratis Pandis (Athens Information Technology Institute) for his valuable contribution.

References

1. Anderson, K. M. (1997). Integrating Open Hypermedia Systems with the World Wide Web. In Proceedings of the 1997 ACM Hypertext Conference, (Southampton, UK).
2. Anderson, K. M., Sherba, S. A., Lepthien, W. V. (2003). Structure and behavior awareness in themis. In Proceedings of the ACM Hypertext 2003 Conference, pp.138-147, (Nottingham, England).
3. Anderson, K. M., Taylor, R. N., and Whitehead, E. J. (1994). Chimera: Hypertext for heterogeneous software environments. In Proceedings of the ACM European conference on hypermedia technology (ECHT '94), Sept. 18-23, 1994, Edinburgh, Scotland, UK, pp. 94-107.
4. Avila-Rosas, A., Moreau, L., Dialani, V., Miles, S., and Liu, X. (2002). Agents for the Grid: A comparison with Web Services (part II: Service Discovery). In AAMAS'02, July, 2002, Bologna, Italy.
5. Berners-Lee, T., Cailliau, R., Luotonen, A., Nielsen, H. F., and Secret, A. (1994). The World-Wide Web. Communications of the ACM, 37(8), pp. 76-82.
6. Bouvin, N. O. (2002). Open Hypermedia in a peer-to-peer context. In Proceedings of the 13th ACM Hypertext Conference, (College Park, Maryland, USA).
7. Bouvin, N. O. (1999). Unifying Strategies for Web Augmentation. In Proceeding of the 10th ACM Hypertext Conference, pp. 91-100, (Darmstadt, Germany).

8. Carr, L., et al. (1995). The Distributed Link Service: A Tool for Publishers, Authors and Readers. In Fourth International World Wide Web Conference: "The Web Revolution". Boston, Massachusetts, USA.
9. Chiu, C. M., Bieber, M., Lu, Q. (2002). Towards Integrating Hypermedia on the Web. Proceeding of the 35th Annual Hawaii International Conference on System Sciences (IEEE 2002).
10. Christodoulou, S.P., Zafiris, P.A., & Papatheodorou, T.S.: "Web Engineering: The Developers' View and a Practitioner's Approach", Web Engineering, Software Engineering and Web Application Development, Lecture Notes in Computer Science, Volume 2016, 2001, pp.170-187.
11. Cotroneo, D., Di Flora, C., Russo, S. (2003). An enhanced service oriented architecture for developing web-based applications. Journal of Web Engineering, Vol. 1, No. 2, 128-146.
12. Engelbart, D. (1998). ACM Hypertext Conference '98 OHS Workshop Keynote Address, (Pittsburgh, Pennsylvania, USA).
13. Gronbaek, K., Bouvin, O. N., Sloth, K. (1997). Designing Dexter-based Hypermedia Services for the World Wide Web. In Proc. of Hypertext '97. Southampton, UK, pp. 146-156.
14. Halasz, F., and Schwartz, M. (1994). The Dexter Hypertext Reference Model. Communications of the ACM, 1994, 37 (2), pp. 30-39.
15. Hall, W., Davis, H., and Hutchings, G. (1996). Rethinking Hypermedia: The Microcosm Approach. Kluwer Academic Publishers, Norwell, Massachusetts, USA.
16. Karousos, N., Pandis, I. (2003). Developer Support in Open Hypermedia Systems: Towards a Hypermedia Service Discovery Mechanism. Proceeding of Metainformatics Symposium (MIS' 03), (Graz, Austria).
17. Karousos, N., Pandis, I., Siegfried, R., and Tzagarakis, M. (2003). Offering Open Hypermedia Services to the WWW: A Step-by-Step Approach for the Developers. In Twelfth International World Wide Web Conference WWW2003, (Budapest, Hungary), pp. 482-489.
18. Karousos, N., Tzagarakis, M. and Pandis, I., (2003). Increasing the Usage of Open Hypermedia Systems: A Developer-Side Approach. Proceeding of ACM Hypertext' 03 conference (Nottingham, United Kingdom).
19. Lassila, O., and Swick, R. R. (1999). Resource Description Framework (RDF) Model and Syntax Specification. W3C Recommendation, 1999. At URL: <http://www.w3.org/TR/REC-rdf-syntax/>.
20. Li, L., and Horrocks, I. (2003). A software framework for matchmaking based on semantic web technology. In Proc. of the Twelfth International World Wide Web Conference (WWW 2003), pages 331-339, 2003.
21. Nürnberg, P. J., Leggett, J. J. and Schneider E. R.: As We Should Have Thought. Proceedings of the 8th ACM Conference on Hypertext (Hypertext '97), Southampton, UK, April 6-11, 1997, 96-101.
22. Nürnberg, P. J., Leggett, J. J., and Wiil, U. K. (1998). An agenda for open hypermedia research. In Proceedings of the 9th ACM Conference on Hypertext, June 20-24, 1998, Pittsburgh, PA, pp. 198-206.
23. Nürnberg, P. J., and Schraefel, M. C. (2002). Relationships Among Structural Computing and Other Fields. JNCA Special Issue on Structural Computing, 2002.
24. Shackelford, D. E., Smith J. B., Smith F. D. The Architecture and Implementation of a Distributed Hypermedia Storage System. In Proceedings of the 1993 ACM Hypertext Conference, (Seattle, WA, Nov), ACM press, pp. 1-13.
25. Universal Description, Discovery and Integration of Web Services (UDDI). At URL: <http://www.uddi.org>.
26. W3C Simple Object Access Protocol (SOAP). At URL: <http://www.w3.org/tr/SOAP>.

27. W3C Web Services Architecture Domain. At URL: <<http://www.w3.org/2002/ws/>>.
28. W3C Web Services Description Language (WSDL). At URL: <http://www.w3.org/tr/WSDL>.
29. Will, U. K., Hicks, L. D., and Nurnberg P. J. (2001). Multiple Open Services: A New Approach to Service Provision in Open Hypermedia Systems. In Proceedings of the 2001 Hypertext, (Aarhus, Denmark), pp. 83-92.