TRIPLE SPACES FOR AN UBIQUITOUS WEB OF SERVICES

Position Paper: W3C Workshop on the Ubiquitous Web, Tokyo, Japan, March 9-10, 2005

Author:

Reto Krummenacher
Thomas Strang
Dieter Fensel

Digital Enterprise Research Institute (DERI)
University of Innsbruck, Austria & National University of Ireland, Galway
Email: {firstname.lastname}@deri.org

ABSTRACT

Triple Space Computing is a communication and coordination paradigm for Web services that combines space-based computing and the Semantic Web. Triple Spaces aim at providing a web that is optimized for machines, thus simplifying the sharing of data and the coordination of devices and services in dynamic and heterogeneous systems. The more ubiquitous the Web becomes, the more the information processing tasks have to be performed by services hidden in the World Wide Web. Services will have to coordinate their efforts, understand each other and exchange information over the Ubiquitous Web to allow human users access to the world's information spaces whenever and wherever they find themselves - or rather, access to the right service for the right person at the right time and at the right place (and with the right price).

INTRODUCTION

The goal of the Ubiquitous Web is to enhance the World Wide Web (WWW) towards a framework of distributed applications interacting across heterogeneous networks and integrating a large variety of information sources and sinks. The interaction patterns, processing goals and information dissemination have to dynamically adapt to user preferences, device capabilities and environmental conditions to disappear from a human user's perception [Weiser, 1991]. This calls upon many complex communication links. Most of the available IT applications depend largely on synchronous communication links between information producers and consumers. Tuple Spaces [Gelernter, 1985] allow distributed sharing of information by various devices without requiring synchronous connections. The Semantic Web [Berners-Lee et al., 2001] extends the Web with machine-processable semantic data, allowing data exchange in heterogeneous application fields. Combining the two introduces a new communication platform that provides persistent and asynchronous dissemination of machine-understandable information [Fensel, 2004], especially suitable for distributed services published and described as Semantic Web services. We call this combined communication platform **Triple Spaces**: RDF *Triples* provide a natural link from the Semantic Web and Tuple *Spaces* to Triple Space Computing.

Triple Spaces provide a basis for an infrastructure that addresses data and interaction heterogeneity, and also greatly decreases the communication overhead common to the task of gathering (context) information and managing ubiquitous computing applications. Triple Spaces is tailored to the needs of machines: (1) optimal storage facilities for semantic data, (2) data and process mediation, (3) integrated reasoning engines. The goal is to develop Triple Space Computing (TSC) to become the web for machines as the World Wide Web is the Web for humans.

UBIQUITOUS TRIPLE SPACES

We currently investigate different approaches to implement the Triple Space Computing paradigm. One possibility investigated is to fully use a shared object space middleware and to extend it with Semantic Web technologies like data mediation and reasoning based on semantically annotated data [TSC, 2005]. Another very promising approach is the use of the well-established Web infrastructure to access the space. The big advantage here is the fact that even small devices already support Web browsers or at least the HTTP protocol and (a priori) no additional software is required to grant access to the space. We will now discuss the challenges implied by Ubiquitous computing: varying and small devices, weakly connected networks, big numbers of interacting nodes and heterogeneous data. We will furthermore take a closer look at how ideas and technologies of the human Web can be used to build Triple Spaces.

Conceptual Space Architecture

TSC requires that all resources (data and spaces) are named by URIs and persistently stored, i.e. the TSC paradigm follows the principle of *persistently publish and read based on unique identification of resources*. This conforms well with the current WWW. After discussing the proposed overall architecture (cf. Figure 1), we are going in more detail to the access model and the data model defined for ubiquitous Triple Space Computing (uTSC). There will be so-called light clients, heavy clients and server nodes. It is worth noting that these terms may in fact be slightly misleading, as all nodes are consumers, as well as producers of semantic information. The difference lies therefore rather in their nature of application than in the actual functionality.

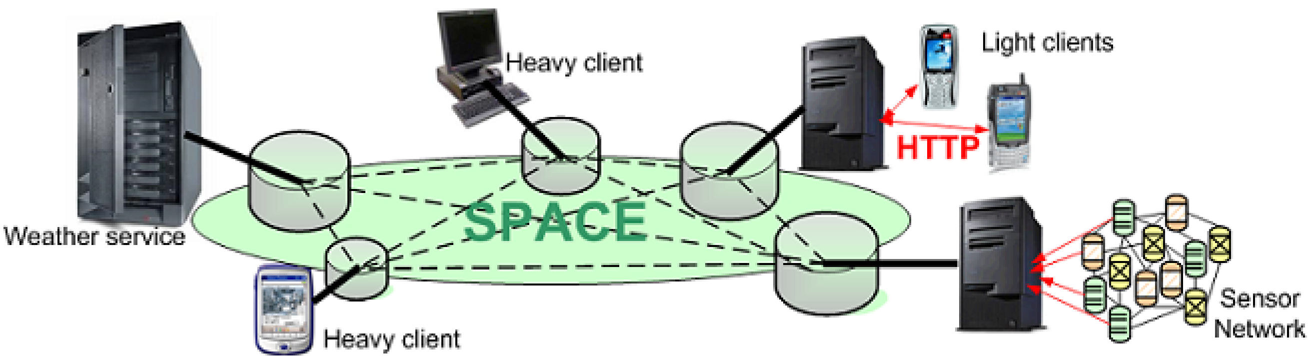

Figure 1: uTSC architecture for Ubiquitous Web of services

All participants are at the same time potential producers and consumers of semantic data. The middleware, in Figure 1 simply called *SPACE* can either be a shared object space implementation, as mentioned above, a P2P-network (e.g., [Nejdl et al., 2001]) or a distributed RDF data stores with enhanced access modules (e.g. [Harth et al., 2005]). Triple Spaces provides consequently a middleware that hides the internal application complexity from the users. The essential elements to look at in this paper are the means to access the space and to provide data. To make the point we shortly discuss the three types of participants: (1) light client, (2) heavy client, and (3) server nodes.

Light clients: These are typical low-performance devices that do not have enough resources to actively participate in the space, i.e. they do not keep a share of the space. By use of a known access point (possibly a proxy server) they can access the space preferably without the need for special software, i.e. by HTTP calls to the access point.

Heavy clients: These are devices that are powerful enough to host a share of the space; PDAs, Tablet PCs or desktop computers. They run the space implementation locally and store a share of the data. This allows for local access to the space, as it is for example planned in the Web Service Execution Environment WSMX [Cimpian & Zaremba, 2005]. The advantage is that the space becomes an integrate part of the application and the application of the space. Although heavy clients are primarily seen as nodes that use the space to gather information and to communicate with (Semantic) Web services they can also host services for other peers and merge hence basically with the server nodes.

Server nodes: These peers are the big information providers, which of course does not exclude information consumption. Two examples shown in Figure 1 would be a Web service that regularly publishes the latest weather news, or a server that collects data from a network of sensors, annotates it and publishes it in the Triple Space.

The space is a shared infrastructure in which heavy clients and server nodes participate. In this way access to at least parts of the space can be guaranteed even if access to the entire space is never ensured. Entire access to the space is never ensured due to the dynamic network topology and weak connectivity of mobile systems. Information like weather forecasts can thus be collected without requesting it from a service as the data resides somewhere in the space. Servers do therefore not get flooded by requests from user applications and user applications do not need to synchronize or even know the contract of a service. uTSC provides an information space that decouples sources and targets in time (asynchronism) , reference (anonymity) and space (disconnectedness) and also partly in vocabulary (mediation of semantic data).

uTSC Access Model

The idea of space-based computing and the major strength of the paradigm is the local access to resources that might be stored remotely. On the architecture image it was shown that access over HTTP should be granted for easy and flexible access to the information space. However, for more sophisticated applications an API is defined that abstracts from the underlying implementation and thus from the middleware and storage. The core interfaces are shown in Table 1. The idea is to support at once the template matching pattern from the tuple space community and also the access to resources by URI as in the Web.

write(URI ts, Graph triples):URI	Write an RDF graph containing one or more triples to the Triple Space ts. *returns:* the URI that identifies the written graph.
read(URI ts, Template template):Graph	Read a set of triples matching the given template. *returns:* the RDF triples that were matched by the template.
read(URI ts, URI graph):Graph	Read the RDF graph identified by the provided URI. *returns:* the RDF graph that is identified by the URI.
take(URI ts, Template template):Graph	Take (read and remove) one RDF graph matching the given template. *returns:* one of the graphs that have matching triples.
take(URI ts, URI graph):Graph	Take the RDF graph identified by the provided URI. *returns:* the graph that is identified by the URI.

Table 1: Interfaces for local access to Triple Spaces

Note that in uTSC, as it is currently defined, we have three types of resources: (1) Triple Spaces, (2) RDF graphs, and (3) RDF triples; we require the spaces and the RDF graphs to be named with an URI. The templates are proposed to be given by (simple) graph patterns in N3, as they are defined for the *WHERE*-clauses of SPARQL [Prud'hommeaux & Seaborne, 2005].

The given interfaces (Table 1) represent only the minimal set of the proposed access patterns. We also defined within the TSC project methods for transactional access to the space, for blocking read and take, as well as a support for publish/subscribe based [Eugster et al., 2003] access to the data. This combined Web (via HTTP) and Tuple Space (via API) access methods allow for flexible and ubiquitous interaction model depending on the application field, the device capabilities and the environmental conditions.

uTSC Data Model

uTSC fully adopts the syntax [Klyne & Carroll, 2004] and the semantics [Hayes & McBride, 2004] of RDF. Even though the triples are the smallest unit of information in RDF we do not consider it as a full resource in TSC, and require therefore solely RDF graphs to be identified by URIs (where nothing hinders a graph from only containing a single triple). There are several reasons why a graph should be named by a unique identifier and why it makes no sense to go as far as finding an identifier per triple by default

Use of "objects": interactions based on graph URIs allows for the exchange of "objects" (interesting among others for the exchange of Semantic Web service descriptions: e.g., WSMO [Lausen et al., 2005], OWL-S [Martin, 2004])

Meta-data: meta-data about graphs can easily be added in an annotation space, where meta-data per graph is enough ([Krummenacher et al., 2005a] requires e.g. knowledge about the source and the writing time of triples to guarantee a minimal amount of trust)

Context modelling: ontology-based context models are the natural extension of graph meta-data and a core research area of Ubiquitous computing [Strang & Linnhoff-Popien, 2004]

Complexity: the three points above show that there is no need for a more fine-grained approach, and that providing an URI per triple and thus e.g. context information per triple would lead to unnecessary complexity

For scalability reasons it is not possible to simply assume a Triple Space to be equal in size and functionality to the whole World Wide Web. To guarantee at least local scalability so-called virtual Triple Spaces are defined. These spaces are envisioned to cluster interaction partners. Particular data is communicated over particular spaces. Consequently consumers and providers of semantic data would inaugurate spaces for services (e.g. ubiquitous computing applications or B2B process integration), as humans have Web site to communicate with other human beings. This analogy leads us to the conclusion that there is no need for a new addressing system, just because we deal with machines.

Contextualized uTSC Model

Context information becomes a crucial aspect in moving the Web towards the Ubiquitous Web and Web services towards a network of thousands of applications that are published, composed and invoked as Semantic Web services. TSC is foreseen to provide an ideal middleware to coordinate the producers and consumers of context data [Krummenacher et al., 2005b]. For a middleware to be effective in highly dynamic environments it is important that the infrastructure not only stores/communicates context information, but that it actually makes use of it as well. Context information in terms of formal facts and rules (see e.g., [Hawke, 2005]) allows to encode user preferences, device capabilities or network status for the use in the Semantic Web. The context rules can then be applied to distribute the data within the space to provide allow technologies like prefetching or smart updates to better cope with the challenges of ubiquitous applications mentioned previously. In that way we could improve the local scalability and service even in setting were devices get frequently disconnected or the local performance is very poor.

Related Activities

There are two known projects around Semantic Tuple Spaces, namely sTuples [Khushraj et al., 2004] and RDFSpaces [Tolksdorf et al., 2005]. The sTuples project enhances a JavaSpace implementation with a generic Semantic Tuple containing an object field of type DAML-OIL Individual that extends the JavaSpace Entry interface. Fundamentally, a non-semantic implementation of Tuple Spaces was augmented with Semantic Web technology. Thus, the underlying data model is a mix of non-semantic and semantic technology; compared to clean semantic triples/quads in Triple Spaces that fully support Semantic Web technology in the Web sense. All in all, sTuples point in the same direction but clearly does not make use of and address Ubiquitous computing. RDFSpaces are a continuation of previous work with XMLSpaces [Tolksdorf & Glaubitz, 2001] done by the same group. XMLSpaces are an extension of IBM's implementation of the classical Linda model in order to support the management of XML documents as types of tuples and tuple fields. This approach too is conceptualized for the current Web and clearly not to cope with the challenges of the Ubiquitous Web.

SUMMARY OF POSITION

In summary our position with respect to the Ubiquitous Web:

Merging Ubiquitous computing and Web computing
The Ubiquitous Web is the inevitable merge of Web computing and Ubiquitous computing, where mobility and adaptivity meet scalability and global coverage.

The Web around (Semantic) Web services
In the Ubiquitous Web services, in particular Semantic Web services have a central role in adapting the Web to user preferences and the dynamic conditions of devices and environment.

Context-aware middleware
Context-awareness becomes very crucial for the Ubiquitous Web. Having a middleware that takes context information into account allows for optimized services to the users.

A Web for machines
For the Ubiquitous Web to come true the services need a web for machines, e.g. a globally accessible network of Triple Spaces, just like the WWW is the highly scalable Web for humans.

A flexible coordination paradigm for machines and humans This machine-tailored middleware is envisioned to take over coordination, mediation, and basic reasoning tasks. In that way it allows the Ubiquitous Web to fast and flexibly integrate new sources and sinks of the managed semantic data.

REFERENCES

[Berners-Lee et al., 2001] T. Berners-Lee, J. Hendler, and O. Lassila: *The Semantic Web*. The Scientific American, May 2001.

[Cimpian & Zaremba, 2005] E. Cimpian, and M. Zaremba (eds.): *Web Service Execution Environment (WSMX)*, W3C Member Submission, June 2005, available: http://www.w3.org/Submission/WSMX/.

[Eugster et al., 2004] P.Th. Eugster, P.A. Felber, R. Guerraoui, and A.-M. Kermarrec: *The Many Faces of Publish/Subscribe*, ACM Computing Surveys, 35(2), 2003: 114-131.

[Fensel, 2004] D. Fensel: *Triple-space computing: Semantic Web Services based on persistent publication of information*. Proc. of the IFIP Int'l Conf. on Intelligence in Communication Systems, Bangkok, Thailand, 2004: 43-53.

[Gelernter, 1985] D. Gelernter: *Generative Communication in Linda*. ACM Transactions on Prog. Languages and Systems, 7(1), 1985: 80-112.

[Harth et al., 2005] A. Harth, M. Magni, and St. Decker: *Scalable Distributed RDF Storage Infrastructure*. DERI Lion Deliverable 1.02, June 2005.

[Hayes & McBride, 2004] P. Hayes and B. McBride (eds.): *RDF Semantics*, W3C Recommendation, February 2004, available at http://www.w3.org/TR/rdf-mt/.

[Klyne & Carroll, 2004] G. Klyne and J. J. Carroll (eds.): *Resource Description Framework (RDF): Concepts and Abstract Syntax*. W3C Recommendation, February 2004, available at http://www.w3.org/TR/rdf-concepts/.

[Krummenacher et al., 2005a] R. Krummenacher, M. Hepp, A. Polleres, Ch. Bussler, and D. Fensel: *WWW or What is Wrong with Web services*, Proc. 3rd European Conf. on Web Services (ECOWS), Nov 2005: 235-243.

[Krummenacher et al., 2005b] R. Krummenacher, J. Kopecky, and Th. Strang: *Sharing Context Information in Semantic Spaces*, Proc. of the OTM 2005 Workshop on Context-Aware Mobile Systems (CAMS'05), Agia Napa, Cyprus, October 2005.

[Khushraj et al., 2004] D. Khushraj, O. Lassila, and T. Finin: *sTuples: Semantic Tuple Spaces*, Proc. of 1st Ann. Int'l Conf. on Mobile and Ubiquitous Systems: Networking and Services, Boston, USA, August 2004: 268-277.

[Martin, 2004] D. Martin (ed.): *OWL-S: Semantic Markup for Web Services*, W3C Member Submission, November 2004, available at http://www.w3.org/Submission/OWL-S/.

[Nejdl et al., 2004] W. Nejdl, B. Wolf, S. Staab, and J. Tane: *Edutella: Searching and Annotating Resources within an RDF-based P2P Network*, Proc. of the Semantic Web Workshop, 11th Int'l WWW Conf., Honolulu, USA, May 2002.

[Prud'hommeaux & Seaborne, 2005] E. Prud'hommeaux and A. Seaborne (eds.): *SPARQL Query Language for RDF*, W3C Working Draft, November 2005, available at http://www.w3.org/TR/rdf-sparql-query/.

[Lausen et al., 2005]H. Lausen, A. Polleres, and D. Roman (eds.): *Web Service Modeling Ontology (WSMO)*, W3C Member Submission, June 2005, available at http://www.w3.org/Submission/WSMO/ .

[Strang & Linnhoff-Popien, 2004] Th. Strang and C. Linnhoff-Popien: *A Context Modeling Survey*, Proc. of the Workshop on Advanced Context Modelling, Reasoning and Management at UbiComp 2004, Nottingham, England, September 2004.

[Tolksdorf & Glaubitz, 2001] R. Tolksdorf and D. Glaubitz: *Coordinating Web-Based Systems with Documents in XMLSpaces*, Proc. of the 9th Int'l Conf. on Cooperative Information Systems, Trento, Italy, September 2001: 356 - 370.

[Tolksdorf et al., 2005] R. Tolksdorf, E. Paslaru Bontas, and L.J.B. Nixon: *Towards a tuplespace-based middleware for the Semantic Web*, Proc. of 2005 Int'l Conf. on Web Intelligence, Compiege, France, September 2005.

[TSC, 2005] Triple Space Computing (TSC) project. Start: March 2005, Funding: Austrian Federal Ministry for Transport, Innovation, and Technology (FIT-IT), Web site: http://tsc.deri.at .

[Weiser, 1991] M. Weiser: *The Computer for the Twenty-First Century*. The Scientific American, 265(3), Sept 1991: 94-104.