Situation Spaces in Context-aware Business Solutions

Jerzy Bartoszek
Technical University of Poznań, Institute of Control and Information Engineering, Pl. M. Skłodowskiej-Curie 5, 60-965 Poznań, Poland
Jerzy.Bartoszek@put.poznan.pl

Grażyna Brzykcy
Technical University of Poznań, Institute of Control and Information Engineering, Pl. M. Skłodowskiej-Curie 5, 60-965 Poznań, Poland
Grazyna.Brzykcy@put.poznan.pl

Abstract

A partial solution of content-oriented processing is proposed. We combine ideas of tuple spaces, spaces of XML documents, the theory of situations and logic programming into one coherent mechanism. This mechanism can be used to build context-aware business solutions, such as enhanced workflow systems, semantic search engines and integrated intelligent user environments that ought to satisfy all the above mentioned features.

1. Introduction

A final form of information system results from an interaction of at least two constituents. On the one hand, the form is justified by our desire for data processing, and on the other – by a current state of the hardware and software technologies and the available computing infrastructures. Accordingly, methods of system designing and implementation but also concepts, models and architectures used in these processes, are continuously adopted and improved for better reflection of the characteristics of real problems (represented by data) and the ways of solving them (data processing).

One of the essential problems for today's enterprises is being able to access the right information at the right time and to achieve this goal in a useful and user-friendly manner. For decision-makers the real problem is not a lack of information, but rather efficient extraction of required information from a great number of information sources. They need the new business solutions that are semantic-oriented and context-aware.

Current business solutions for SMEs (medium-sized enterprises), particularly search tools, are inadequate for global information structure mainly because of their syntax-orientation. Moreover, the way in which the information is stored and the way in which it can be accessed have not been designed to be interoperable. As the amount of information one has available to make economic decisions keeps growing, the advanced search engines enabling intelligent access to heterogeneous information sources ought to be provided. The SEWASIE system (SEmantic Webs and AgentS in Integrated Economics) [Bergamaschi 2005], result of the European IST (Information Society Technologies) project, is a good example of the new generation solutions. This system is a collection of information nodes and specialized agents that are mediators between information sources and users. Information nodes provide a virtual view of information sources via some metadata (e.g., ontologies, indexes). The semantic enrichment of data stores is the basis of structured and efficient (reduction of the overall transaction costs) communication in the system. The extended features of the SEWASIE system include integrated searching, negotiating and monitoring.

By means of this paper we attempt to support the efforts of research community and present a partial solution of content-oriented processing. We propose to use a general concept of situation and high-level abstraction of communication and synchronization at the information source level. Our solution combines ideas of tuple spaces, spaces of XML documents, the theory of situations and logic programming into one coherent mechanism. This mechanism can be used to build knowledge-based multi-agent systems, which are distributed, XML-oriented, and equipped with reasoning machinery. It is particularly suited for context-aware business solutions, such as enhanced workflow systems, semantic search engines and integrated intelligent user environments that ought to satisfy all the above mentioned features.

For today's complex information systems situatedness, openness and locality in control and interactions are distinctive characteristics. Situation theory [Barwise 1983], [Devlin 1991] faces these requirements and seems to be a suitable theoretical foundation of software systems. Moreover, with its orientation on meaning and information content, situation theory accurately matches the basic principles of global semantic networks. The basic concepts of this theory are recalled in section 4.

Partiality and relevance of information together with context-dependence and extensive reification are other essential hypotheses of situation semantics. All these properties make situation theory particularly convenient for using with open systems that are not fixed and possibly not completely available.

Other interesting model originates from tuple spaces and the Linda coordination language. A short description of Linda and some examples of systems with tuple spaces are contained in section 2. An evolution of tuple space computing to the form of Semantic Web middleware [Tolksdorf 2004a] is also related there.

In this work we indicate possibility of building a bridge between well-founded theory and efficient implementation of tuple spaces computation model. We make a suggestion to use tuple spaces to represent situations. In our solution we go one step further and assume that tuple spaces are enhanced with Prolog-like deduction.

In section 5 we present an idea of taking an advantage of situation theory in XMLSpaces [Tolksdorf 2004] and we are going to propose Situation Spaces. The conception is a result of our works on coordination in agent and workflow systems and comes from our experience of employing declarative programming to diverse problems (e.g., [Brzykcy 2003]).

2. Meaning and context in information systems

Information is universally used in different sciences and seems to be a way of conceiving various aspects of the world. An exciting study of the information-based way of analyzing actions performed by an agent (also human being) one can find in [Devlin 2005]. In this paper the utilization of information stance in a broad area of different sciences, particularly social sciences, is also suggested. Problems of human interaction, considered in this draft, are similar to those found in contemporary information systems. In accordance with commonly accepted characteristics (e.g., [Bradshow 1997], [Estrin 2002], [Zambonelli 2003]), today's software systems should be built of autonomous components (named agents) that can perceive and affect the environment (may be partially). An agent with locally defined flow of control and with some form of context-awareness (e.g., locality in interactions) can cooperate with other agents in an open system.

Problems (data) may have a different form and a choice of data representation can affect the way data are computed. However, information content (meaning of data) should not depend, at any rate, on an assumed shape of data. On the other hand, agents may perform many actions more efficiently provided that sense (meaning of data) is known. This is a very basic requirement of Semantic Web, which is a vision of a distributed network of data (knowledge) resources that can be automatically and effectively processed with respect to their semantics. The intensive efforts undertaken in order to realize this vision bear witness to the significant value that is put down to information meaning. We appear to be on the way of a radical shift in programming paradigm to the attitude that is oriented at information content (content-oriented processing).

Meaning is defined as the function of an item in the world that is perceived by an agent. But perception is merely partial because of restricted cognitive resources of an agent, its limited knowledge and the world dynamics. Therefore interpretation of data and knowledge is always partial and sensitive about context. So, explicit representation of context and contextual knowledge maintenance is an alternative, which is difficult to avoid in information systems. To equip a system with these abilities a reification mechanism need to be extensively used.

The content-oriented approach to information retrieval and routing is also clearly visible in practical works. We can see the partial specification of the Semantic Web vision that takes a form of various standards (e.g., for resources – URI, RDF, ontologies – OWL, documents – DOM, mark-up languages – XML). Software engineers and scientists are spending a great deal of effort elaborating subsequent standards.

It is worth noticing that all the above mentioned features of information are captured and formally described in situation theory [Barwise 1983], [Devlin 1991].

Another important task in the Web is to realize global resources, where information is published and persistently stored. To properly solve the problem, the model of tuple spaces is proposed. Information is there represented as a shared space in which data can be placed and retrieved by agents. Access to the space is realized by means of some straightforward operations oriented at information content (tuple space computing). These systems take advantage of the Linda model [Gelernter 1992] that is a simple abstraction of synchronization and communication mechanisms. Linda is a suitable solution to heterogeneity of agents, protocols and processes in distributed open environments.

In modern applications the primary importance of explicit representation of context is already recognized and yet various context models have been published [Strang 2004]. Different attempts to formalize context are also undertaken (e.g., [Giunchiglia 1993], [Akman 1996], [Baclawski 2003], [Gangemi 2003]). In business applications Giunchiglia's approach to context may be particularly suitable for software agents because the main focus is on context reasoning. A context consists of all the knowledge that is used by an agent for deliberation about its goals (making effective decisions in current situation).

3. Tuple space computing

Tuple space computing originates in the Linda system [Gelernter 1992]. All the activities undertaken in the system consist in flow of tuples. Agents can exchange data and synchronize with each other using a built-in mechanism of pattern matching over a shared memory of tuples. An agent can insert a tuple t using the out(t) operation. Another agent can retrieve a tuple that matches a given template T (and remove it from the tuple space) using the in(T) operation. Tuples can also be evaluated in separate processes. Tuples, operations and distributed tuple spaces have been implemented in Prolog-D-Linda [Sutcliffe 1990], [Sutcliffe 1991]. In this system tuples are expressed as Prolog facts, and tuple spaces are Prolog databases. Additionally, tuple spaces can contain rules (i.e. Prolog clauses). By means of the rules one can describe the whole sets of tuples not enumerated explicitly.

The XMLSpaces conception [Toksdorf 2004] is an extension to the Linda systems. In XMLSpaces the main role is played by XML documents. A pattern matching on ordinary tuples of the Linda systems is replaced by extended matching flexibility on nested tuples and on various data types for fields of tuples. In original concept of XMLSpaces Prolog-like rules are not considered. In paragraph 4, we will try to demonstrate how it can be changed.

XMLSpaces have been extended to so called WorkSpaces [Tolksdorf 2002] and Semantic Web Spaces [Tolsdorf 2004a]. The WorkSpaces architecture uses XMLSpaces to implement workflow systems. It combines distributed processing and workflow concepts. Process definitions and process relevant data are XML documents. These documents are transformed by means of XSL technology. Documents containing definitions of processes are transformed to set of XML documents, which describe process' steps (process' activities). These steps are expressed as XSL rules and are executed by an XSL processor. In each step of process execution, the WorkSpace engine [Tolksdorf 2002] reads XML document containing actual activity, takes an appropriate input document, applies to it XSL rules indicated by the activity and generates an output document. Then the next activity is taken from a WorkSpace and the process execution is continued.

In Semantic Web Spaces elements of the XMLSpaces and relations between them can be described by means of RDF and OWL metadescriptions. These metadescriptions are treated as XML documents and are contained in a space. An agent before using documents in a given space (may be previously unknown to it) can and should read these metadescriptions. Taking advantage of this additional contextual knowledge agent can improve efficiency of searching, reasoning or other activities.

On the other hand, another powerful theory is known – the situation theory [Barwise 1983], [Devlin 1991], where similar concepts play a crucial role. The implementations of situation theory [Erkan 1995] and the Linda model (e.g., [Sutcliffe 1990], [Sutcliffe 1991]) are known as well. Because we adopt these concepts in XMLSpaces, let us shortly present them.

4. A short overview of Situation Theory

The most basic concepts of situation theory are infons and situations. If R is an n-place relation and a1, ..., an are objects appropriate for the respective argument places of R, then a tuple <<R, r1→a1, ..., rn→an, 1>> denotes the informational item that a1, ..., an are standing in the relation R, and a tuple <<R, r1→a1, ..., rn→an, 0>> denote the informational item that a1, ..., an are not standing in the relation R. r1, ..., rn describe roles of objects a1, ..., an and they are also called the names of arguments of R. Objects like <<R, r1→a1, ..., rn→an, p>> are called infons (p is called its polarity and is equal to 0 or 1).
Some arguments of R may be missing. Minimality conditions for R indicate which groups of argument roles of R need to be filled in order to produce a well-formed (well-defined) infon. If σ1 and σ2 are two infons with the same relation

R, and σ2 has at least the same arguments as σ1, then σ1 ≤ σ2 (σ1 subsumes σ2). In this way infons can represent partial information.

A situation is a part of reality, which can be picked out by a cognitive agent. Situations make certain infons factual. Taking into consideration a situation s and an infon σ, it is written s ⊨ σ when s supports σ (σ is true in s). If I is a finite set of infons and s is a situation, we write s ⊨ I, if s ⊨ σ for every σ in I.

To indicate that the situation s1 is a part of the situation s2, it is written s1 ⊆ s2. The part-of relation is antisymetric, reflexive and transitive. It provides a partial ordering of the situations.

Objects considered in situations are classified into different types. The basic types include temporal locations, spatial locations, individuals, relations, situations, infons, parameters and polarity. If s is a situation, p is a parameter and I is a finite set of infons (involving p), then there is a type [p | s ⊨ I] of those objects to which p may be anchored in s, so that all conditions in I are satisfied.

It is written o : T to indicate that object o is of type T. Taking into account a situation parameter S and a set I of infons, there may be a corresponding type [S | S ⊨ I] of situations in which the conditions in I obtain. This process of defining a type from parameter S and a set I is known as situation type abstraction.

In situation theory, the "flow of information" is realized via constraints. A constraint S1 ⇒ S2, where S1 and S2 are situation types, corresponds in essence to the infon <<involves, S1, S2, 1>> of some (meta)situation. Cognitively it means that, if there is a situation s1 of type S1, then there is a situation s2 of type S2. Awareness of this constraint is what enables a cognitive agent that perceives situation s1 to infer that the situation is part of a larger actuality in which situation s2 should be perceived. It should be noted that, although the types S1 and S2 may involve parameters, the constraint is parameter-free infon that links two specific situation types. In general, any constraint may be dependent on a set B of background conditions under which it will convey information. This is written as [S1 ⇒ S2] / B.

It was shown by Akman and Surav [Akman 1996] that situation theory can be used both as a knowledge representation scheme and to support contextual reasoning. They proposed an extended situation theory where contexts are modeled with situation types and constraints.

The Prolog-like language PROSIT (PROgramming in SItuation Theory) [Erkan 1995] provides forward and backward chaining constraints respected by some kind of situations.

5. An idea of Situation Spaces

Now we present some realization of a vision of context-aware information processing. Our solution is aimed at efficient gathering, storing and processing of information and all these services are sensitive to information content and information meaning. The proposed infrastructure for semantic network consists of situation spaces (which are situations themselves). By analogy to terms previously used for sets of spaces we name it Situation Spaces.

In our solution the situation theory plays a role of formal, well-founded base for tasks of knowledge processing and representation. This theory gives a small set of concepts with individuals, relations and situations, which are suitable for world modeling. Another value of this theory is a possibility of homogenous representation of data, metadata, inference rules and other constraints by means of situations and reification mechanism. As a consequence, using the same concept of situation one can express various aspects of context, that are ontologies applied by the agents, accessible resources, their localization (place and time), knowledge shared by the agents, agents' mental states etc. Context may be easily modified as situations are modeled by means of sets of infons.

One more crucial feature of situation theory is ability of expressing partial knowledge.

As situation theory has been devised to create a theory of meaning and information content it is particularly suitable for agents' communication and knowledge exchange. Moreover situation semantics is regarded as one of the best tools of analyzing the meaning.

It is also worth noticing the presence of some implementations of situation theory (see [Erkan 1995]).

An idea of tuple spaces and computation model based on Linda can be seen as superior to traditional models. We find it essential to use this model in Situation Spaces. This approach simplifies communication and coordination tasks in open distributed systems and has been successfully applied in diverse domains (e.g., XMLSpaces [Tolksdorf 2004]). The set of operations characteristic for this model is appropriate for elements of set manipulation and information retrieving is possible by pattern matching. Due to mentioned features the model may be used in open distributed systems.

Usefulness and versatility of tuple spaces are confirmed by many examples of their applications, such as software integration, agent coordination and workflow systems. There also exist implementations of tuple spaces: XMLSpaces, WorkSpaces, XMLSpaces.NET.

We assume that all the data in Situation Spaces take a form of XML documents. This representation of data has al least two advantages: standardization and realizability in distributed environments, for instance, on .NET platform. XMLSpaces are originally implemented on this platform [Tolksdorf 2004].

The decision of including logical inferences into the Situation Spaces appears to have a minor value because any inference engine may be placed instead. However, we decided to choose verified Prolog solutions.

Let us outline the key implementation decisions and assumptions expressing our solution.

In Situation Spaces infons are implemented as complex, hierarchical tuples in XMLSpaces. In fact they are XML documents with metadata (semantically typed). The idea of implementing knowledge source as a tuple space is explored.

Prolog-like rules are implemented in the similar manner. They represent intentional facts and constraints imposed on facts and are suited for information flow. They may be used in workflow management systems [Tolsdorf 2002] to express actions in processes.

Rules contain predicates (literals) of the form: space_name: predicate (arg1,...,argn), where the prefix space_name indicates the space, in fact the context, in which the predicate must be evaluated. If it is the current space, the prefix can be omitted.

We assume that situations may be contain infons, which are metadescriptions. In this manner the idea of self-described situations can be realized. The essential types of metadescriptions are ontologies, which describe terms and concepts of the given situation.

Situations can be treated as XMLSpaces. They have names and URIs.

Mental states of agents are also represented by situations, whereas constraints from situation theory provide the mechanism that captures the way agents make inferences and act. Agent actions can be defined by Prolog-like rules.

6. Conclusions

Our goal is to take advantage of the Situation Spaces concepts for distributed workflow systems in which agents realize processes in heterogeneous environments. In these efforts we are supported by recent research on making use of information stance in social sciences [Devlin 2005]. As workflow systems are frequently used in business and social areas (for example, in SMEs and in labor and social welfare organizations) the situation theory seems to be an adequate framework. In these systems flow of information between agents (artificial or human beings) is very important. Documents and utterances can be understood only with the respect to some (real and abstract) situations and contexts (e.g., ontologies, localizations, accessible resources)

Situation Spaces are implemented as context-dependent advanced search and deduction engine. It provides intelligent access to heterogeneous information sources on the Web. There are many benefits of using this engine: a reduction of transaction costs, efficient deduction and search, context-dependent communication facilities. Within the business solutions Situation Spaces can support integrated, context-aware and personalized data acquisition.

7. References

[Akman 1996] V. Akman, M. Surav, "The Use of Situation Theory in Context Modeling", *Computational Intelligence* 12 (4).

[Baclawski 2003] K. Baclawski, M. Kokar,, C. Matheus, J. Letkowski, M. Malczewski, "Formalization of Situation Awareness", In: Kilov, H. (ed.): *Behavioral Specifications of Businesses and Systems*. Kluwer Academic Publishers, Boston/Dodrecht/London.

[Barwise 1983] J. Barwise, J. Perry, *Situations and attitudes*. MIT Press, Cambridge, MA.

[Bergamaschi 2005] S. Bergamaschi, C. Quix, M. Jarke, "The SEWASIE EU IST Project", *SIG SEMIS Bulletin*, Vol. 2, No. 1, Feb. 2005.

[Bradshaw 1997] J. Bradshaw (ed.), *Software agents*. The MIT Press.

[Brzykcy 2003] G. Brzykcy, J. Martinek, A. Meissner, P. Skrzypczynski, P., "Control Aspects of the Blackboard Agent Architecture for a Mobile Robot", *Control and Cybernetics* 32 (4).

[Devlin 1991] K. Devlin, *Logic and information*. Cambridge University Press, New York.

[Devlin 2005] K. Devlin, D. Rosenberg, "Information in the Social Sciences", Draft: June 20, www.stanford.edu/~kdevlin/HPI_SocialSciences.pdf.

[Erkan 1995] T. Erkan, V. Akman, "Situations and Computation: An Overview of Recent Research", In: Griffith, J., Hinrichs, E.W., Nakazawa, T. (eds.): *Proceedings of the Topics in Constraint Grammar Formalism for Computational Linguistics*. Copenhagen, pp. 77-104.

[Estrin 2002] D. Estrin, D. Culler, K. Pister,, G. Sukjatme, "Connecting the Physical World with Pervasive Networks", *IEEE Pervasive Computing* 1(1) pp. 59–69.

[Gangemi 2003] A. Gangemi, P. Mika, "Understanding the Semantic Web through Descriptions and Situations", In R. Meersman, Zahir Tari et al. (eds.): *Proceedings of the International Conference on Ontologies, Databases and Applications of SEmantics*, Lecture Notes in Computer Science (LNCS) 2519, Springer Verlag.

[Gelernter 1992] D. Gelernter, N. Carriero, "Coordination Languages and their Significance", *Communication of the ACM* 35(2), pp. 97–107.

[Giunchiglia 1993] F. Giunchiglia, "Contextual Reasoning", *Epistemologica* 16, pp. 345-364.

[Strang 2004] T. Strang, C. Linhoff-Popien, "A Context Modeling Survey", *Workshop on Advanced Context Modelling, Reasoning and Management as part of UbiComp 2004 – The Sixth International Conference on Ubiquitous Computing*, Nottingham.

[Sutcliffe 1990] G. Sutcliffe, J. Pinakis, "Prolog-Linda – An Embedding of Linda in muProlog", In: Tsang, C.P. (ed.): *Proceedings of the AI'90 – the 4th Austrialian Conference on Artificial Intelligence*, p. 331–340.

[Sutcliffe 1991] G. Sutcliffe, J. Pinakis, "Prolog-D-Linda: An Embedding of Linda in SICStus Prolog", Technical Report 91/7, The University of Western Australia, Department of Computer Science.

[Tolksdorf 2002] R. Tolksdorf, "Workspaces: A Web-Based Workflow Management System", *IEEE Internet Computing*, 6(5), pp. 18-26.

[Tolksdorf 2004] R. Tolksdorf, F. Liedsch, D. M. Nguyen, "XMLSpaces.NET: An Extensible Tuplespace as XML Middleware", In: *Proceedings of the 2nd International Workshop on .NET Technologies'2004*.

[Tolksdorf 2004a] R. Tolksdorf, L. Nixon, F. Liebsch, "Semantic Web Spaces", Technical Report B-04-11, Freie Universität Berlin.

[Zambonelli 2003] F. Zambonelli, H. Van Dyke Parunak, "Towards a Paradigm Change in Computer Science and Software Engineering: A Synthesis", *The Knowledge Engineering Review*, vol. 18(04).