AN APPROACH TO DATA-DRIVEN ADAPTABLE SERVICE PROCESSES

George Athanasopoulos and Aphrodite Tsalgatidou
Dept. of Informatics & Telecommunication, National and Kapodistrian University of Athens, Athens, Greece

Keywords: Adaptable Systems, Service Compositions, Data-driven Adaptations, Context-aware Software Systems.

Abstract: Within the currently forming pervasive computing environment, services and information sources thrive. Instantiations of the service oriented computing paradigm, e.g. Web, Peer-to-Peer (P2P) and Grid services, are continuously emerging, whilst information can be collected from several information sources, e.g. materializations of the Web 2.0 and Web 3.0 trends, Social Networking apps and Sensor Networks. Within this context the development of adaptable service oriented processes utilizing heterogeneous services, in addition to available information, is an emerging trend. This paper presents an approach and an enabling architecture that leverage the provision of data-driven, adaptable, heterogeneous service processes. Core within the proposed architecture is a set of interacting components that accommodate the acquisition of information, the execution of service chains and their adaptation, based on collected information.

1 INTRODUCTION

Services and information sources flourish within the currently forming pervasive computing environment. Service-Oriented Computing (SOC) along with the emerging instantiations, e.g. Web, P2P and Grid services, promise to revolutionize the way applications and systems are built by fostering the provision of adaptable systems. In addition, the emerging Sensor Web (Botts et al., 2008), and the materializations of the Web 2.0 and Web 3.0 paradigms, e.g. Social Networking applications or Agent based systems, provide new types of information sources that can be exploited towards the provision of emerging types of systems and services. Within this frame, the development of adaptable service processes, comprising heterogeneous services and information stemming from existing sources, is an emerging trend; in the context of this paper, service processes are regarded as systems, which operate within a specific environment, and perform pre-specified activities, via the use of services, in an orderly manner producing and/or consuming related information during their execution. This trend, cannot be addressed by contemporary rigid approaches such as WS-BPEL (Alves, et al., 2007). Therefore, the development of adaptable service processes calls for novel approaches.

AI techniques (especially AI planning ones) have been extensively applied towards the provision of dynamically composed service oriented orchestrations (Jinghai & Xiaomeng, 2004). Most of them focus on the provision of automatically constructed orchestrations (or similarly called task plans) that comprise Web services solely. Along the same lines, the Context Aware Computing (CAC) research community has applied considerable efforts towards the utilization of contextual information for the adaptation of web service compositions, e.g. (Lirong, Zhongzhi & Fen, 2006). The majority of these approaches in both research fields address neither the interoperability concerns raised by the multiple instantiations of the service oriented computing paradigm nor the utilization of available and/or emerging information sources in tandem.

An emerging approach towards the integration of services that has lately received considerable momentum is based on the utilization of the Tuplespace model (Rossi, Cabri & Denti, 2001). Within this paper we present an approach which leverages the merits of tuplespace paradigm for the provision of data-driven, adaptable, heterogeneous, service compositions. A component, implementing the Tuplespace model, called Semantic Context Space Engine (SCS Engine), along with a Service Orchestration Engine and appropriate adaptation algorithms (i.e. Process Optimizer) provide the required functionality.

The rest of the paper is organized as follows: next we present an approach towards the provision of such a mechanism and the prime components of the proposed architecture; in the following we present a comparison of our approach against similar approaches. We conclude this paper with the presentation of our remarks on the presented approach and our future work plans.

2 DATA-DRIVEN ADAPTATION

The prime assumption of our approach is based on the observation that a service process, comprising heterogeneous services, should be able to utilize the information available within its environment and adapt its execution accordingly. Within this context, a process state is not solely depending on the values of its internal parameters, on the resulting outcomes from the invocation of its constituent services and/or on its internal operations but, also on information pertaining to the process environment; therefore, the latter should be taken into account during process development and execution.

To facilitate the provision of such adaptable processes we propose an approach that leverages information contained within a specific 'space' to adapt a service process using appropriate adaptation algorithms. A space is considered to be the process's environment which is open to other processes and systems e.g. Service-Oriented systems or Agent based systems, for information exchange. For the process to be able to exploit the available information, appropriate extensions are pre-configured and embedded within the process specification.

illustrates the proposed platform with the comprising components which are:

- A **Semantic Context Space Engine** responsible for the collection of contextual information,
- A **Process Optimizer** responsible for the adaptation of service processes based on the collected information, and
- A **Service Orchestration Engine** responsible for the execution of heterogeneous service processes.

The SCS Engine provides an open space where one may place relevant information. From a functional point of view the SCS Engine leverages one to *i) Write and Retrieve* information within the process's environment and to ii) *Logically Group* information of interest to a specific domain and *Specify associations* among logical information groups, which contain information elements from related/depending domains.

The Process Optimizer component implements an AI planner that facilitates the discovery of process plans that control the execution and adaptation of service processes. Such plans are usually modelled as conditional plans which contain branching control structures i.e. if-then-else that decide which execution path will be followed based on the value of a condition. The problem of data-driven adaptation can be modelled as a non-deterministic, partial observability planning problem, where Model Checking techniques (Nau, Ghallab & Traverso, 2004) are extensively applied lately.

The Service Orchestration engine provides a BPEL-based engine that facilitates the execution of heterogeneous service orchestrations (e.g. Web, Grid and P2P service orchestrations). One of the core features of the orchestration engine is the support for the monitoring and reconfiguration of process state according to the suggestions made by the Process Optimizer. To facilitate the execution and the adaptation of a process state, the orchestration engine exchanges information with the SCS Engine.

Details on the comprising components are presented next.

3 SEMANTIC CONTEXT SPACE

The Tuplespace model has been extensively applied in the coordination of distributed and parallel systems (Rossi, Cabri & Denti, 2001). Yet, its utilization by the Service-Oriented Computing paradigm is an emerging trend that has been accompanied by proprietary extensions (Nixon et al., 2008) (Zeng, Lei & Chandramouli, 2005). Although services provide a layer of abstraction that facilitates the interoperation of systems over the web, the merits of the Tuplespace paradigm can further enhance the Service Oriented model. These merits include the i) decoupling of process components, the ii) associative based addressing (i.e. data is referenced by its content and not by its address) and the support for the provision of iii) synchronous and asynchronous communication patterns (Rossi, Cabri & Denti 2001). More to that, such properties will foster the formation of new types of collaboration schemes among Service-Oriented systems and other existing or emerging types of systems such as Agent based systems, Sensor and Grid applications.

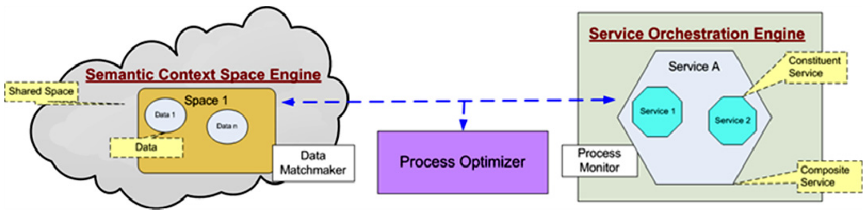

Figure 1: Platform high level architecture.

The SCS Engine incorporates appropriate semantic annotations to the Tuplespace information model and has been structured so as to accommodate the following functional needs:

- Support for multiple meta-information models along with associated querying mechanisms for the discovery of contained information elements (i.e. registered pieces of information within the SCS Engine). Such models could range from OWL based annotations to simple key-value pairs.
- Use of 'scope' delimiters to facilitate the logical organization of registered information elements in 'information islands' with related information.
- Support for the specification of affiliations among scopes that may spawn across dispersed engines, facilitating the formation of logical associations among related 'information islands.'
- Use of a Web-based reference mechanism for uniquely addressing elements and scopes.
- Handling of both Java and XML based information elements.

In order to facilitate the abovementioned features we decided to use the JavaSpace service (Freeman, Hupfer & Arnold, 1999) of the Jini framework (Waldo & Team, 2000) as a basis for the implementation of the SCS Engine. The selection of the JavaSpace service instead of an open source Tuplespace e.g. TSpace (Fontoura et al., 2003) or an XML based Tuplespace such as XMLSpace (Tolksdorf, Liebsch & Nguyen, 2004), was primarily driven by our need for flexibility over the supported meta-information models and the related query engines, as well as the need for supporting Java and XML based information elements. Nonetheless, one may argue that the SCS Engine could be implemented on top of a database management system e.g. an RDBMS such as Oracle, or MySQL etc. but the rigidness of their information models and their complex implementation does not render them a first class choice.

The interface of the SCS Engine is an extended version of the Linda model catering for: the writing, reading (discovery) and retrieval (taking) of information elements, either from the whole space or from a specific scope; the persistent querying (continuous querying) for information elements; and the management of scopes such as the creation, discovery, and removal of a scope, the creation and/or removal of affiliations among scopes. As one may easily note the provided operations accommodate a basic set of functionality. This was decided so as facilitate the extensibility of the SCS engine upon which additional higher level operations and mechanisms can be built.

All these operations are accessible either through a Java based interface or via the use of a Web service based interface (currently only the Java based interface has been implemented).

4 PROCESS OPTIMIZATION

As it has been pointed out in other research efforts too (Pistore et al., 2004), automated service composition can be mapped to a non-deterministic, partially observable planning problem. This is because, similar to a partially observable planning problem, automated service composition has to confront concerns such as the uncertainty raised by the interaction with external services (and partners) and the partial knowledge of the composition state accruing from the lack of information on the internals of each constituent service (i.e. interacting partner). The construction of task plans in non-deterministic and partially observable domains has received considerable investigation by the AI planning community, e.g. (Kuter et al., 2007) (Bryce, 2006). Solutions to such problems are in the form of conditional plans; conditional plans contain branching control structures i.e. if-then-else structures that decide which path will be followed based on the value of a condition.

In this frame, a formal representation of the problem domain (D) is a tuple $\langle S, A, R, O, X \rangle$, where:

- S: is the finite set of states of the associated state transition system,
- A: is the finite set of actions $A = \{a_i | i \leq l\}$,
- $R \subseteq S \times A \times S$ is the transition relation,
- O: is a finite state of observation variables $O = \{o_i \mid i \leq n\}$
- $X_o: S \times \{\bot, \top\}$ is the relation for the evaluation of observation variables $o \in O$ on each state. Within our context the value of an observation variable is independent of the action that may have preceded

Contrary to what stands in a deterministic problem domain (Nau, Ghallab & Traverso, 2004) the transition relation R can map the execution of an action $a \in A$, on a state $s \in S$ (assuming that α is applicable on s) to more than one successor states i.e. $S' \subseteq S$, $|S'| \geq 1$. An action α is applicable on a state $s \in S$ iff there exists a state $s' \in S$ such that $R(s, a, s')$ stands. The set O contains the finite set of observation variables o_i whose values are evaluated at runtime. The value of each observation variable at each state is defined by the observation relation X. A simplification normally introduced to avoid the unnecessary complexities is to consider observation variables as boolean variables whose values could be either true or false (i.e. $\{\top, \bot\}$). Therefore if $X_o(s, \top)$ holds at a state $s \in S$ then the value of variable o at state s is *True*. The dual holds in cases where variable o is *False*. In cases where both $X_o(s, \top)$ and $X_o(s, \bot)$ hold variable o has an undefined value.

Even though the data-driven process adaptation problem can be mapped to the non-deterministic, partially observable planning problem a thorough look into the requirements of the proposed approach unveils additional concerns that are not properly handled by the classic representation of the planning problem (D) and the associated solutions e.g. (Pistore et al., 2004)(Pistore et al., 2005). These concerns are related to the *'origin'* and *'validity-time'* property of observed information, as well as to the need to consider additional information (i.e. partially matching information) to the one that is captured by the pre-specified set of observation variables.

Most of the existing approaches, adhering to D, utilize observation variables to grasp information stemming from either the controlling process or partner services. Nonetheless, this assumption hinders the interaction of the controlling process with external (i.e. with respect to the process) systems and information sources. Observation variables are tuned for collecting information from pre-specified sources i.e. constituent services or the controlling process, thus neglecting information that may stem from uncontrolled sources. Within a controlled environment observation variables are always conforming to specific constraints expressed in terms of format and semantic meaning but, in a pervasive computing environment varying format and semantic meaning is the norm rather than the exception. An additional important information trait is that of the associated validity-time; validity-time dictates the period of time within which information may be safely consumed. Existing approaches adhering to D, avoid considering this property when dealing with the automated composition of services.

To cater for these concerns we propose a set of appropriate extensions and supporting mechanisms. These comprise: a) a mechanism facilitating the valuation of observations (i.e. observation variables) based on queries executed over an open set of information elements, b) an observation 'interpolation' mechanism that facilitates the inclusion of additional information elements, and c) the use of validity-time property for each information element and appropriate management features.

Details on the proposed extensions and their ramifications on the planning domain are presented next.

4.1 Observation Considerations

We can safely assume that an observation variable (o_n) of the planning domain D is defined in a finite set of *information elements i.e.* $o_n \in OD_n$ *(*see (Pistore et al., 2004)). The assessment of an observation variable at runtime should always return a value within the specified set OD_n. In the frame of our approach the valuation of an observation variable (o_n) is mapped to the assessment of a query (q_n) performed over an information source, which corresponds to the system environment. However, due to its open nature, such an information source may comprise information elements that are irrelevant to an executing process. Thus, the set of queries ($Q=\{q_1,q_2,..,q_n\}$) performed over a source should be properly structured so as to avoid the retrieval of erroneous information elements. To accommodate this concern we provide a semantic-based information discovery mechanism. Queries (q_i) are extracted out of the semantic and syntactic details of the associated observation variables (o_i). The SCS Engine, executes these queries, and matching results are returned back to the running process.

Our approach towards the use of similar information elements is based on the interpolation of observations with related ones in a manner controlled by the utilized ontology. The underpinning assumption is that instead of considering partially matching results to the performed observations we may as well look for exact matches to 'partially matching' observations. Hence, the set of observations *O* linked to a process is expanded by the introduction of related observations. To facilitate this process we introduce two additional features i.e. an expansion operator (*oExp*) and an expansion ratio property (*sD*). *sD* dictates the minimum (i.e. the infimum) similarity distance among the concepts of an ontology so as to consider them part of the same set (i.e. expansion set). Given an ontology, a concept (*co*) of that ontology and an expansion ration (*sD*) the *oExp* operator returns all terms of the provided ontology whose similarity to *co* is equal to or greater than *sD*.

The original set of observations *O* associated to a specific process can therefore be expanded to a set *O´* with the application of the *oExp* operator over *O* for a specific *sD* value and ontology Vc. To ensure that the extended set (O') contains observations that can be handled by our system we need to prune it so as to remove observations that may lead to unacceptable states. The pruning process ensures that O' contains observations leading to states where actions (α) belonging to *A* can be applied.

Finally, to accommodate the concern related to the information validity time, every information element registered to the SCS Engine is attributed with a specific Time To Live (TTL) property which dictates its validity period. The SCS Engine is responsible for cleaning up the space and removing obsolete information elements. Therefore, executing queries for the discovery of information are ensured to return related and valid information elements.

5 PROCESS EXECUTION

The outcome of the process optimization procedure is a conditional plan, containing possible adaptation cases, that can be transformed into an enhanced WS-BPEL description. The pre-defined adaptation cases incorporated with the enhanced service specification control the adjustment of a process based on collected information. The provided specification can be seamlessly executed by an existing BPEL based orchestration engine.

Nonetheless, the need to accommodate the use of several types of services poses significant concerns. This requirement has been exemplified in the SODIUM project (Tsalgatidou, et al., 2008); our approach towards the accommodation of heterogeneous services has been influenced by the outcomes of this project. Appropriate extensions have to be introduced both to the Service Orchestration Engine and to the BPEL specification to accommodate the invocation of services such as Grid and P2P services. Nevertheless, as these types of services i.e. P2P and Grid services, are described in WSDL-based languages i.e. WSRF (Czajkowski, et al., 2004) specification and PSDL (Tsalgatidou, et al., 2008), the required extensions can be easily accommodated.

PSDL along with the associated middleware facilitating the description and invocation of P2P services respectively, can be seamlessly utilized by existing BPEL orchestration engines to leverage the use of P2P services. Appropriate plug-ins can be provided to support the binding of such services. A crucial aspect for the utilization of Grid services (or similarly WSRF services) is the identification of WS-Resources (i.e. the combination of Web Services and Resources). An indirect approach such as the one employed in (Ezenweoye et al. 2007), catering for the specification of the correct address, i.e. based on the WS-Addressing protocol, can be used. The WS-Resource endpoint can be provided either via a variable or as the outcome of a WSRF Factory service.

Another core feature of the adopted orchestration engine, rooted to the need for data-driven adaptation, is the inherent support for the continuous monitoring of executing process instances. Upon the discovery of related information by the SCS Engine a process monitoring component adapts the execution path according to the suggestions embedded in the process specification by the Process Optimizer.

Interactions among the Service Orchestration Engine and the SCS Engine components are bidirectional. The Service Orchestration Engine provides information to the SCS Engine that is used for refining the specified information retrieval queries, whereas the SCS Engine upon the discovery of matching information elements pushes them to the Service Orchestration Engine. Based on certain criteria, e.g. if the discovered information can be used for a meaningful adaptation of the running process, the Process Monitor component of the Service Orchestration Engine utilizes this information to adapt t he executing process accordingly.

An engine that can easily serve as the basis for the provision of such features is the Apache ODE engine. Apache ODE is an open source, Java-based implementation of a BPEL-based orchestration engine facilitating the execution of WS-BPEL v2.0 specifications. It accommodates appropriate extensions catering for the interaction of the runtime with external data sources e.g. databases. This feature can be further extended in order to facilitate the interactions of the Orchestration Engine and the SCS Engine.

6 RELATED WORK

The work presented in this paper lies across the fields of several research communities. The Context Aware Computing community has spent considerable efforts on adaptable service compositions. Contrary to our approach, which facilitates heterogeneous service processes, quite a lot of efforts such as the one by Lirong et al. (Lirong, Zhongzhi & Fen, 2006) have focused on the provision of adaptable Web service compositions. Other approaches such as the one employed by Vukovic, et al. (Vukovic & Robinson, 2004) utilize contextual information for the elicitation of specific properties e.g. location, network connectivity, etc. and consequently adapt web service compositions in a (semi-) automatic manner.

AI and workflow based techniques have been also applied for the provision of adaptable service compositions. Most of these techniques as it has been also pointed out by Jinghai et al. (Jinghai & Xiaomeng, 2004), have focused on processes which comprise Web services or semantically-enhanced Web services. Further to that, most of these approaches neglect the importance of contextual information. Service compositions are usually modeled as deterministic state transition systems and the resulting composition problems are modeled as planning problems within deterministic and fully observable domains (Nau, Ghallab & Traverso, 2004).

Similar to the approach ensued by Pistore et.al (Pistore et al., 2005) which caters for the provision of automated compositions of web services, our approach considers the provision of composite services as a non-deterministic, partially observable planning problem. Nonetheless, we provide for the use of additional types of services and the management of contextual information e.g. the consideration of partially relevant information and the use of the TTL property, as well as for the use of other types of service beyond Web services. Along the lines of the ALLOW project (Marconi et al., 2009.) our mechanism is also based on the incorporation of context adaptation mechanisms in the process specification. Nonetheless, ALLOW accommodates a constrain based mechanism to facilitate the evaluation of design-time specific contextual properties, whereas our approach adheres to an automatically configured data-based adaptation strategy.

With respect to existing approaches catering for the provision of semantically enhanced Tuplespaces, the SCS Engine provides a flexible and extendable architecture. Most of the contemporary semantic Tuplespaces can be considered as Knowledge Bases, which utilize RDF for the representation of semantic information. Contrary to these efforts, the SCS Engine provides a simpler, yet extensible model, which can support various meta-information schemes for the annotation of Information elements. Additional high level functions e.g. automated inference mechanisms can be built on top of the proposed mechanism. Further to that, the SCS Engine accommodates an enhanced logical grouping mechanism i.e. scopes, which differs considerably from the ones employed by other contemporary approaches. Contrary to the scoping mechanism used by Merrick (Merrick & Wood, 2000) or the one used in the Semantic Web Spaces (Tolksdorf, et al., 2005), the provided mechanism accommodates a flexible affiliation scheme that facilitates the specification of various types of relationships among scopes e.g. hierarchical relationships or other user defined associations.

7 CONCLUSIONS

The need for the provision of data-driven adaptable service processes has been highlighted in several application domains, e.g. the crisis management or the environmental services domain (e.g. Envision EU project www.envision-project.eu). In this paper we presented an approach and an enabling architecture addressing this problem. Core within the specified approach is a set of components, which facilitate the collection of information elements, the adaptation of service processes and their respective execution.

The proposed approach along with the related platform constitutes a departure from the current state of the art. On the one hand it accommodates the provision of data-driven adaptable heterogeneous service processes, something that, to the best of our knowledge, has not been addressed so far whilst, on the other hand it fosters the opening of service processes to external systems and the emergence of additional collaboration schemes among them. This form of cooperation further promotes the decoupling among collaborating parties which is an important prerequisite for the provision of adaptable systems. Our future work plans include the finalization of the implementation of the platform components that are currently in an early development stage i.e. the Service Orchestration Engine and the Process Optimizer. In addition, with respect to the Process Optimizer, we plan to utilize and evaluate contemporary algorithms proposed for the respective planning problem domain e.g. MBP (Pistore et al., 2004), POND (Bryce, 2006).

ACKNOWLEDGEMENTS

This work has been partially funded by ELKE (contract 70/3/10294) and the European Commission (contract ICT-FP7-249120 ENVISION project)

REFERENCES

Alves, A., et al. 2007, Web Services Business Process Execution Language Version 2.0. OASIS Standard, Apr. 2007, OASIS WSBPEL TC.

Botts, M., et al., 2008, 'OGC Sensor Web Enablement: Overview and High Level Architecture', White Paper, Open Geospatial Consortium INC, (OGC 07-165).

Bryce, D. 2006, 'POND: The Partially-Observable and Non-Deterministic Planner', International Conference on Automated Planning & Scheduling 2006, Jun. 2006, Cumbria, UK.

Czajkowski, K., et al., 2004, 'The WS-Resource Framework', Whitepaper, Globus Alliance.

Ezenweoye, O., et al. 2007, 'Grid service composition in BPEL for scientific applications', International Conference on Grid computing, high-performAnce and Distributed Applications, 2007, Vilamoura, Algarve, Portugal.

Fontoura, M., et al. 2003, 'TSpaces services suite: Automating the development and management of Web Services', 12th International World Wide Web Conference, 2003, Budapest, HUNGARY.

Freeman, E, Hupfer, S & Arnold, K 1999, JavaSpaces Principles, Patterns, and Practice: Principles, Patterns, and Practice, Addison-Wesley.

Jinghai, R. & Xiaomeng, S. 2004, 'A Survey of Automated Web Service Composition Methods', First International Workshop on Semantic Web Services and Web Process Composition (SWSWPC 2004), Jul. 2004, San Diego, CA, USA, pp. 43-54.

Kuter, U., et al., 2007, 'Conditionalization: Adapting forward-chaining planners to partially observable environments', Workshop on Planning and Execution for Real-World Systems -- Principles and Practices for Planning in Execution, Sept. 2007, Rhode Island, U.S.A.

Lirong, Q, Zhongzhi, S & Fen, L, 2006, 'Context Optimization of AI planning for Services Composition', IEEE International Conference on e-Business Engineering (ICEBE 2006), Oct. 2006, IEEE Computer Society, Shanghai, China.

Marconi, A., et al., 'Enabling Adaptation of Pervasive Flows: Built-in Contextual Adaptation.', 7th International Joint Conference on Service Oriented Computing (ICSOC- 2009), Nov. 2009, Stockholm, Sweden

Merrick, I. & Wood, A 2000, 'Coordination with scopes', ACM Symposium on Applied Computing-Volume 1, Mar. 2000, Como, Italy.

Nau, D., Ghallab, M. & Traverso, P. 2004, Automated Planning: Theory & Practice, Morgan Kaufmann Publishers Inc., San Francisco, C. A, USA.

Nixon, L., et al., 2008, 'Tuplespace-based computing for the semantic web: A survey of the state-of-the-art', The Knowledge Engineering Review (2008), Cambridge University Press, vol 23, no. 2, pp. 181--212, .

Pistore, M., et al. 2004, 'Planning and Monitoring Web Service Composition', Artificial Intelligence: Methodology, Systems, and Applications, 11th International Conference (AIMSA 2004), Sept. 2004, Varna, Bulgaria.

Pistore, M, et al.2005, 'Automated Synthesis of Composite BPEL4WS Web Services', IEEE International Conference on Web Services (ICWS05), IEEE Computer Society, Jul. 2005, Orlando, Florida, USA.

Rossi, D., Cabri, G. & Denti, E 2001, 'Tuple-based technologies for coordination', Coordination of Internet agents: models, technologies, and applications, pp. 83-109, Springer-Verlag, London, U.K.

Tolksdorf, R, Liebsch, F. & Nguyen, L., 2004, 'XMLSpaces.NET: An Extensible Tuplespace as XML Middleware', 2nd International Workshop on.NET Technologies,.NET Technologies'2004, May-Jun. 2004, Plzen, Czech Republic.

Tolksdorf, R., et al. 2005, 'Towards a tuplespace-based middleware for the SemanticWeb', IEEE/WIC/ACM International Conference on Web Intelligence (WI2005), Sept. 2005, Compiegne University of Technology, France.

Tsalgatidou, A., et al. 2008, 'Unified Discovery and Composition of Heterogeneous Services: The SODIUM Approach', in, At your service: An overview of results of projects in the field of service engineering of the IST programme, MIT Press Series on Information Systems.

Vukovic, M. & Robinson, P. 2004, 'Adaptive, planning-based, Web service composition for context awareness', International Conference on Pervasive Computing (Pervasive 2004), Apr. 2004, Vienna, Austria.

Waldo, J. & Team, J. T. 2000, The Jini TM Specification, 2nd Edition, Addison-Wesley Professional.

Zeng, L., Lei, H & Chandramouli, B 2005, 'Semantic Tuplespace', 3rd Internation Conference on Service Orienteed Computing, (ICSOC-2005), Dec. 2005, Amsterdam, Netherlands.