Type-based Validation and Management of Business Service Interoperability

Toni Ruokolainen

Dept. of Computer Science
P.O. Box 68 (Gustaf Hällströmin katu 2b)
FI-00014 UNIVERSITY OF HELSINKI, FINLAND
Toni.Ruokolainen@cs.Helsinki.FI

Abstract. To attain interoperable business service collaborations, the nature and concepts of service-oriented inter-enterprise environments must be carefully analysed and formalized. Moreover, development tools and runtime infrastructure services are needed for establishing a sustainable business service ecosystem that provides interoperable business service delivery. The research proposal introduced in this paper addresses these issues by developing metamodels and methods for validating and maintaining business service interoperability in the context of inter-enterprise collaborations.

1 Introduction

Modern networked enterprises require flexibility and openness from collaboration facilities to tolerate changes in the ever-changing technology and business domains, and to gain a competitive edge. However, heterogeneity, autonomy, and dynamism inherent in inter-enterprise computing environments present severe interoperability problems when loosely coupled collaborations should be established. To attain interoperable business service collaborations, the nature and concepts of service-oriented inter-enterprise environments must be carefully analysed and formalized. More over, development tools and runtime infrastructure services are needed for establishing a sustainable business service ecosystem that provides interoperable business service delivery.

The research proposal introduced in this paper addresses the interoperability problem by developing methods for validating and maintaining business service interoperability in the context of inter-enterprise collaborations. The research is thematically located within the areas of service-oriented computing [1] and modern middleware research [2], model-driven engineering (MDE) [3], as well as formal methods [4–6]. The contributions of the research are focussed mainly on the areas of service-oriented computing and integrated formal methods while MDE is adopted as a supporting framework and software engineering discipline.

The structure of this paper is as follows. Section 2 discusses the related research areas and technologies, and identifies challenges to be addressed. The research theme and approach is introduced in Section 3. Finally, the expected results and contributions are reflected upon in Section 4.

2 Background and Significance

The openness of collaborations and agility of organizations stressed by modern electronic business necessitates very flexible interaction relationships between individual enterprise information systems. The degree of flexibility can be increased using the *service-oriented computing* (SOC) approach which is claimed to attain a very loosely coupled model of distributed computing and to provide means for the "open service markets" [1]. From the author's perspective, the SOC field still lacks support for three essential elements that can truly bring the ideal of open service markets closer. These missing elements and challenges to address are *1)* a feasible trading mechanism for interoperable business service delivery, *2)* mechanisms for attaining loosely coupled service relationships, and *3)* a methodology for service-oriented software engineering (SOSE).

The research is motivated by the preceding challenges. The approach taken is based on a notion of *service typing* [7] that gives formal rigour to such concepts as business service behaviour, interoperability and substitutability. Metamodels are provided for service definition and description with theoretical foundations for verifying interoperability between service definitions, and validating the correspondence of a service description (business service offer) with respect to the claimed service definition (service type). Such validation and verification functionalities play an important role when considering service trading in truly open environments. In addition to a feasible service trading infrastructure such concepts as separation of concerns, runtime binding of business services, and dynamic configurability of service capabilities are needed to attain loose coupling in service-oriented systems. These concepts are addressed by the research and reflected in the metamodels, as for example in the form of metamodels for non-functional aspects [8]. Finally, the notion of service typing and corresponding toolchain is believed to fertilize a SOSE methodology; this hypothesis and the feasibility of the methodology is to be validated during the research.

3 Research Theme and Approach

The overall theme of my research can be characterised as type-based validation and management of business service interoperability in the context of inter-enterprise collaborations. The research follows a constructive research methodology with phases of conceptualization, formalization, artefact construction, and evaluation of the approach. The elementary contents of the constructive research phases are discussed in the following.

3.1 Conceptualizing service-oriented collaborations

In the conceptualization phase, the target concepts from the domain of inter-enterprise collaborative systems are identified and formalized into a set of metamodels. The resulting modeling framework follows the principles of MDE [3] and comprises a hierarchy that is illustrated in Figure 1. The highest layer consists of the MOF constructs [9] applicable for defining metamodels. The *collaborative systems metamodel* provides target concepts for specific kinds of collaborative systems. The set of concepts include

notions for collaboration, interaction, service-oriented computing etc. The concepts defined at the *federated service communities metamodel* describe such notions as service types [7], business network models [2], and electronic contracts. In addition for defining modelling constructs for creating the knowledge required, the metamodels prescribe the essential facilities for an abstract service-oriented computing platform.

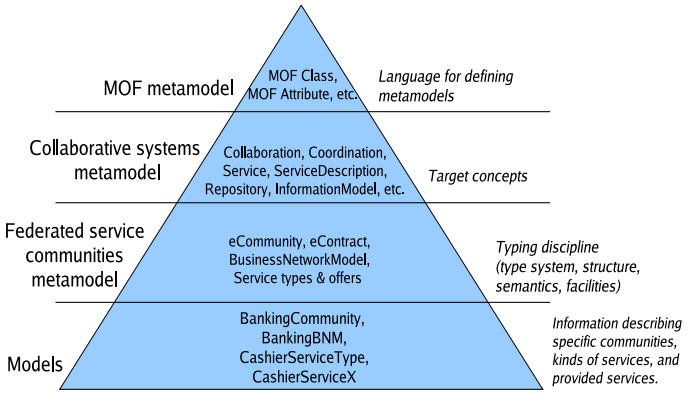

Fig. 1. A four-level modelling hierarchy for describing federated service communities.

Preliminary versions of the metamodels have been introduced in [10]. The complete versions of the metamodels are defined in the author's Licentiate's thesis. The metamodels are illustrated using UML class diagrams. The Object Constraint Language (OCL) is used for further refining the semantics of the metamodels by defining refinement relationships and inter-dependencies between different metamodels and their concepts. Currently the set of metamodels contains over 15 metamodels (packages) with over 80 classes. The metamodels are concretely developed over the Eclipse Modeling Framework (EMF) [1].

3.2 Formalizing business service interoperability and collaborations

The formalization involves construction of a typing discipline that involves three essential type structures, namely 1) service types, 2) binding types, and 3) collaboration types. While service types consider the behaviour of individual business services, the two latter prescribe consistency rules for bilateral and multilateral service interactions (i.e. service compositions and choreographies), respectively. The abstract syntax for all the preceding type structures is given by the federated service communities metamodel depicted in Figure 1.

A *service type* is an abstract description of service capabilities, behaviour and structure [7]. It is a unit of design and composition which represents a bilateral business

[1] http://www.eclipse.org/emf

service interface. Service typing is a scheme of behavioural typing where service descriptions are considered as terms that have to be well-typed with respect to corresponding service types. The service typing discipline is based on session typing [5], pi-calculus [4], and research conducted on the areas of XML-typing [6] and process algebras with XML-typing such as the PiDuce [11]. For service typing purposes the original session typing concept is extended with constructs for handling XML-like structured data (see e.g. [6]) and the tagged braching constructs (see [5]) that are reminiscents of the RPC-paradigm are replaced with constructs compatible with the message passing semantics of SOC. Service typing as the foundation for business service interoperability has been discussed in [7] with an initial discussion about the type management infrastructure and its applicability during the business service development process.

A *binding type* characterizes a bilateral business service connection. It provides the semantics for the connection by defining business protocols used at the connection endpoints and, when necessary, a set of transformations to establish compatibility between the endpoints. Behavioural compatibility between business protocols and thus consistency of the binding type is validated by methods based on the session typing discipline [5]. In addition to business protocols, a binding type may define non-functional properties for the connection (e.g. "*secured interaction*" or "*non-repudiable service*") and business protocol transformations. A preliminary formalization of the non-functional properties has already been conducted [8]. The business protocol transformations may involve mappings between business document structures or even transformations between behavioural patterns.

Service composition and choreographies are managed by so-called *collaboration types* that define the well-formedness and consistency rules for such service collaborations. There are two major research challenges to be addressed: 1) giving type theoretic interpretations for service collaborations, and 2) developing the consistency criteria and their validation methods for the collaboration types. Collaboration types for service compositions and choreographies are constructed as compositions between a set of service types and corresponding coordination structures. A coordination structure formalizes the inter-dependencies between service types; feasibility of so-called event structures [12] as the logical framework for collaboration types will be evaluated. The consistency criteria for collaboration types includes the absence of circular relationships between causal dependencies and the behaviour induced by the coordination structure and service type behaviours, since such a relationship would result in a deadlock. Moreover, satisfaction of coordination obligations prescribed in coordination structures by the service types has to be validated; a static analysis framework, possibly based on the flow logics [13], has to be developed for this purpose.

The semantics of business processes and collaboration choreographies will be founded on pi-calculus [4]-based process calculus. The labelled selection constructs used in the process calculi of session typing disciplines such as described in [5] are replaced with pattern matching constructs. In the process calculus, sessions are created over explicit session creation channels as usual [5]; however, the use of session creation channels shall be governed by a linear usage-typing discipline that provides means for prescribing obligations or prohibitions with respect to the usage of corresponding communication channels. The process calculus must also incorporate XML-based messag-

ing. XML-based messaging has previously been used in process calculi such as the PiDuce [11], but the session typing has not been used in the calculi the author is aware of.

3.3 Constructing the facilities for service-oriented computing

In the constructive part of the research a set of meta-information repositories and software engineering tools are developed. The repositories and tools are based on the metamodels and formalisms developed during the previous research efforts. All of the routine management code is provided by the metamodels, and the code generation facilities provided by the Eclipse environment. Most of the actual work expected in this phase considers the implementation of the verification and static analysis algorithms developed during the formalization phase.

The type management infrastructure is a public and distributed meta-information management system which maintains service type information and relationships between service types. Type management infrastructure consists of public type repositories and name registries. Type repositories implement type checking and type matching functionalities which are needed for interoperation validation during collaboration establishment. Name registries are used for name-based resolution of meta-information such as service types and business network models in the Pilarcos interoperability middleware [2]. The role of name registries is not a foundational part of my research but their existence is necessary for delivering a business service trading environment.

The development tools to be constructed include modelling tools for service types, collaboration types and business network models for example. The development tools utilize the meta-information repositories discussed above. For the purpose of integrating the different development tools and meta-information repositories the Eclipse MDDi framework [2] can be utilized.

4 Expected Results and Contribution

The expected results elaborate the concepts of service-oriented computing and especially the role of formal service definitions as part of service-oriented architectures and software engineering processes. Moreover, research and development of facilities to attain loosely coupled business service collaborations will be strongly represented in this research. The foundations for these are laid by the conceptualization work in the form of metamodels and typing disciplines.

The incredients for a SOSE methodology are addressed by the research proposal. First of all, the service typing discipline provides support for interoperable service discovery and delivery while the collaboration types address service composition issues. Secondly, the semantics of business processes and corresponding type-based interoperability validation procedures can be attached to SOSE design and development tools. Using the results stemming from the previous research areas, a tool-chain and lightweight methodology for SOSE can ultimately be constructed. Consequently, this re-

[2] http://www.eclipse.org/mddi

search also contributes to the work on integrating formal methods in software engineering processes.

Feasibility of the service typing discipline, metamodels and corresponding development and SOA facilities are validated using case studies. The feasibility of the service type concept must be validated from a practical perspective: is it valuable for business service developers, and are the type checking and other analysis procedures implementable in the development tools and meta-information repositories?

References

1. Papazoglou, M.P., Georgakopoulos, D.: Special issue on Service-Oriented Computing. Commun. ACM **46** (2003)
2. Kutvonen, L., Ruokolainen, T., Metso, J.: Interoperability middleware for federated business services in web-Pilarcos. International Journal in Enterprise Information Systems **3** (2007) 1–21
3. Schmidt, D.C.: Model-Driven Engineering. Computer **39** (2006) 25–31
4. Milner, R., Parrow, J., Walker, D.: A calculus of mobile processes, part I/II. Journal of Information and Computation **100** (1992) 1–77
5. Vallecillo, A., Vasconcelos, V.T., Ravara, A.: Typing the behavior of objects and components using session types. Electronic Notes in Theoretical Computer Science **68** (2003) Presented at FOCLASA'02.
6. Hosoya, H., Pierce, B.C.: XDuce: A statically typed XML processing language. ACM Trans. Inter. Tech. **3** (2003) 117–148
7. Ruokolainen, T., Kutvonen, L.: Service Typing in Collaborative Systems. In Doumeingts, G., Müller, J., Morel, G., Vallespir, B., eds.: Enterprise Interoperability: New Challenges and Approaches, Springer (2007) 343–354
8. Ruokolainen, T., Kutvonen, L.: Characterizing Non-Functional Aspects of Services in Federated Service Communities. Manuscript. (2007)
9. Object Management Group: Meta Object Facility (MOF) Core Specification. 2.0 edn. (2006) OMG Available Specification – formal/06-01-01.
10. Ruokolainen, T., Kutvonen, L., Metso, J.: Ontology for Federated Management of Business Networks. In: 2nd International Workshop on Interoperability Solutions to Trust, Security, Policies and QoS for Enhanced Enterprise Systems. (2007)
11. Brown, A., Laneve, C., Meredith, G.: PiDuce: A process calculus with native XML datatypes. In: 2nd International Workshop on Web Services and Formal Methods (WS-FM 2005), Versailles, France (2005)
12. Winskel, G., Nielsen, M.: Models for concurrency. In Abramsky, S., Gabbay, D.M., Maibaum, T.S.E., eds.: Semantic Modelling. Volume 4 of Handbook of Logic in Computer Science. Clarendon Press (1995)
13. Nielson, H.R., Nielson, F.: Flow Logic: a multi-paradigmatic approach to static analysis. In T. Mogensen et al., ed.: The Essence of Computation: Complexity, Analysis, Transformation. Essays dedicated to Neil D. Jones. Volume 2566 of Lecture Notes in Computer Science. Springer Verlag (2002) 223–244