The FRESCO Framework: An Overview

Giacomo Piccinelli, Christian Zirpins and Winfried Lamersdorf

***Abstract*—The dynamic composition of existing services into new services is at the core of service-oriented computing. The objective of FRESCO (Foundational Research on Service Composition) is to develop a framework that service providers can use in order to model, develop, and execute composite services. The FRESCO framework will include conceptual tools, such as models for service composition and aggregation. The framework will also include technology elements, such as an integrated development environment and specific components of the infrastructure for service execution. A methodology will be given for the use of the FRESCO framework in the development of composite service solutions. In this paper, we describe the main aspects of the approach to service composition adopted in FRESCO.**

***Index Terms*—Electronic Business, Service Composition**

I. Introduction

IN the economic theory, a *service* is an immaterial economic good that is provided by a legal entity (referred to as the provider) to satisfy the needs of another entity (referred to as the client) [1]. The client focuses on the use of the capabilities provided by the service. The provider focuses on the resources and operational knowledge that substantiate such capabilities.

Service-oriented computing can be defined as the conceptual and technology framework for the electronic creation and provision of services. Beyond electronic data exchange, service-oriented computing targets the basic infrastructure of clients and providers. Services become the unit of modularisation for business capabilities. A uniform service model applies to different types of service. The distinction between domains becomes just one aspect of the model. Apart from the logic of specific business rules, there should be no obstacles to the immediate incorporation of an external capability into the business processes of a company. Fundamental aspect of service-oriented computing is the capability to compose services into new services.

Manuscript received October 16, 2002. This work was supported by the Hewlett Packard Laboratories.

Giacomo Piccinelli is a reserach assistant at the Department of Computer Science, University College London, Gower Street, London, WC1E 6BT, UK (e-mail: g.piccinelli@cs.ucl.ac.uk). He also acts as external collaborator for the HP Labs, Bristol, UK.

Christian Zirpins a reserach assistant at the VSIS Group, Computer Science Department, University of Hamburg, 22527 Hamburg, Germany (e-mail: zirpins@informatik.uni-hamburg.de).

Winfried Lamersdorf is head of the VSIS Group, Computer Science Department, University of Hamburg, 22527 Hamburg, Germany (e-mail: lamersd@informatik.uni-hamburg.de).

In FRESCO (Foundational Research on Service Composition), we envision an integrated approach to service creation and provision based on composition, aggregation, and coordination of pre-existing services. Our focus is on the dynamic nature of composite services. Capitalising on the experience of research projects such as COSMOS [2], DynamiCS [3] and DySCo [4], our goal is to develop a conceptual and technology framework for dynamic service composition. The intended users of such framework are technical developers as well as business developers. We believe that the successful realisation of composite services depends on the convergence of business and technical expertise.

In this document, we present a preliminary description of the FRESCO framework. Section two contains an outline of the business scenarios that are used in FRESCO. Such scenarios are used for reference as well as validation purposes. Sections three and four cover the core concepts of the service and service composition model envisioned in FRESCO. Section five contains an outline of the technology components of the framework. In section six, we discuss related works in the service composition area. Section seven contains some observations and remarks.

II. Scenarios

Building on the experience of FreightMixer [5], the application scenarios we consider for FRESCO are in the context of the transport industry. The focus is on a FRESCO-enabled service provider (FreightMixer), in terms of both service offer and execution capabilities. Distinctive features of FreightMixer are rich service offer and efficient cost structure.

From a service-offer perspective, FreightMixer maintains a wide portfolio of predefined services. For special customers, both ad-hoc customisation of existing services and on-demand definition of new services are also supported. The main constraint for new services is that the content must stay within the scope of the domain knowledge of FreightMixer. For example, the transport of particular goods may be arranged in a way that half of the goods follows one route and the other half follows a different route. Having the goods subjected to processing (e.g. testing, labelling) might instead be outside the scope of FreightMixer.

In terms of execution capabilities, the infrastructure of FreightMixer is kept to a minimum. The business model of FreightMixer is based on dynamic composition of services and aggregation of service providers. The FRESCO framework provides to FreightMixer the methodology and conceptual tools required for modelling composite service-oriented solutions. FreightMixer adds the business knowledge of the transport industry. The framework also includes service modelling tools and core components of the execution infrastructure. FreightMixer adds a selection of components that complete the execution capabilities of the system.

As in the initial version of FreightMixer, there are two common elements for all the scenarios. On the conceptual level, the focus is on services and service composition. On the practical level, a business-level interface that shows the various aspects of customer interaction and service delivery is used as main communication instrument.

III. Service Model in FRESCO

The service model adopted in FRESCO is service-centric. A service is based on a set of core capabilities that a service provider turns into new capabilities. We refer to the new capability as *service content*. Integral part of the service is the provision logic for the service content. The provision logic is enforced by specific capabilities of the service. We refer to such capabilities as *service shell.* The type of a service is defined based on both service content and provision logic. Similar considerations apply to the definition of a service instance.

The capabilities involved in the service shell fall into five categories: interaction, publication, negotiation, contract management, and billing. Interaction relates to the information exchange between service and users (e.g. ontology and process logic). Publication relates to the communication of information about a service to potential users (e.g. advertising). Negotiation relates to the agreement over terms and conditions associated to the provision of the service (e.g. price or optional features). Contract management relates to the compliance of the service with the terms and conditions agreed with the user (e.g. service level agreements). Billing relates to the enforcement of the charging model for the service usage (e.g. request tracking). For all the categories, the attention is on the information flows as well as on the procedural logic associated to service provision.

The main roles involved in the service model are service provider, service user, and intermediaries. Specialisations of the main roles appear in various parts of the service shell. Delegation of roles is transparent to the model. For example, FreightMixer can relay on a third party to fulfil the duties involved in an insurer role. Nevertheless, FreightMixer retains full responsibility for the level of insurance agreed with the customer. Similarly, the customer can relay on a bank to fulfil the role of payer. While orthogonal to the service model, role management is a fundamental aspect of FRESCO.

IV. Service Composition in FRESCO

Similarly to the FRESCO service model, the composition model envisioned in FRESCO is service-centric. The main assumption in FRESCO is that the capabilities of a service S are based entirely on other services (SC_1, SC_2, ...). We refer to S as the *composite service*, and to SC_i as the *service components*. Composite service and service components are based on the same service model. A composite service can act as service component for other composite services.

Service composition in FRESCO is about creating new services out of existing services. The content of a composite service depends on the content of the related service components. For example, the transport capabilities of FreightMixer depend on the transport capabilities of its suppliers. Similar dependencies apply to the provision logic of a composite service and its components. For example, the service level agreements (SLAs) offered by FreightMixer reflect the SLAs offered by its suppliers.

In the remaining part of this section, we describe the main characteristics of the service composition model for FRESCO. Aggregation and coordination of service components are highlighted as specific aspects of the overall model.

A. Service Composition

Given $\mathcal{S}$ the set of all possible service types, the FRESCO model for service composition can be described as the definition framework for functions f of type $\mathcal{S}^* \rightarrow \mathcal{S}^*$. We refer to f as a *composition function*. The definition of a composition function captures the use of service components and of specific elements of the composition framework for the enforcement of new services. Both domain and co-domain of a composition function can include multiple instances of a service type; the rationale is that multiple instances of a service may be used as well as produced. Excluding the capabilities provided by the composition model, all the capabilities involved in the enforcement of new services are modelled as services.

Time and the interdependencies between services are fundamental dimensions for services and service composition. For example, FreightMixer may offer a transport service whereby the customer decides on some of the features (e.g. level of insurance) at different stages during the execution of the service. Depending on the choices of the customer and on when the choices are made, some service components may or may not be required. Temporal and causal relations between service components are fundamental aspects of the logic of enforcement for a composite service. The dynamic nature of such relations is central to the service composition model proposed in FRESCO.

Intermediation plays an important role for the composition of services. In FRESCO, an *intermediary* is defined as an entity intervening in the relations between two or more services. Intermediation can be the sole purpose of specific services (e.g. brokers), as well as the result of specific roles played by a service component in a composite service. The FRESCO model is based on distributed intelligence; intermediation and delegation provide the glue between service components. Adaptability is another aspect of the model.

B. *Service Coordination*

A composite service defines a network of relations connecting service components. The coordination model covers the aspects of operation and interaction in service provision related to the cooperation between different service components.

Reflecting the requirements coming from the composition model, the coordination model envisioned for FRESCO supports dynamic changes of the cooperation logic between service components. Following from the example in section 4.1, the introduction of a new insurance service may require the coordination logic for the other services to be adjusted. Similarly to the composition intelligence, the coordination intelligence is distributed. Intermediation, delegation and adaptability are directly supported in the model.

C. *Service Aggregation*

Service aggregation covers the acquisition of instances of service components by the provider of the composite service. The aggregation of service components has a direct impact on the provision of the composite service. In the example in section 4.1, FreightMixer may commit to a customer request before having acquired all the service instances required. On the one side, FreightMixer can reduce costs by acquiring only what is actually required to satisfy the customer. On the other side, fluctuations in the service offer may leave FreightMixer incapable to fulfil contractual promises.

The aggregation model envisioned for FRESCO supports dynamic acquisition of service components. The model captures the interdependencies between services, as well as the notion of surrogate. Overall, the model is based on the definition and optimisation of risk and utility functions. Intermediation, delegation and adaptability are fundamental aspects of the model. The aggregation intelligence is distributed.

V. The FRESCO Technology Framework

The objective of the overall FRESCO framework is to support service providers in the creation and provision of composite services. To put FRESCO concepts into practice, the composition model will be supported by concrete technology in the form of tools and infrastructure components. Such technology augments the technical capabilities of service providers related to the support of the lifecycle of composite services. We refer to the technology-related part of the FRESCO framework as *FRESCO technology framework*.

In the remaining part of this section, we first outline the model envisioned in FRESCO for service providers. We then present an overview of the technology framework we will contribute to the realisation of such model.

A. *Provider Model*

Although the notion of service is central for service providers, the *provider model* is provider-centric. A provider offers a set of services. The external view of a service is captured in a service type, which reflects the content and the provision logic for the service. For each service, the provider defines a specific implementation logic based on the FRESCO composition model. We refer to the implementation logic of a service as *service definition*. Service definitions represent a key asset for a provider, and they are not visible to service users.

The realisation of the service content can be based on internal as well as external capabilities. The only requirement is that all capabilities are modelled as services. The internal capabilities of a provider can derive from a wide range of resources. For example, FreightMixer may have a process management system, a knowledge base, and a customer relationship management system. We refer to the overall set of internal and external capabilities as the *execution platform* of a provider. As an observation, we envision technology trends such as grid computing and data centres having a potentially big impact on the execution platform of many service providers; hence on the service offer of such providers.

Like for the service content, also the realisation of the service shell is based on a set of capabilities that can be internal or external to the service provider. The main difference is that there is no requirement for such capabilities to be modelled as services. We refer to the capabilities that realise the service shell as *service platform*. The service platform is divided into five parts, each related to one of the aspects of the service shell indicated in the FRESCO service model (interaction, publication, negotiation, contract management, and billing). A provider also maintains knowledge of clients and service usage in accordance with specific models. For example, the client model may capture information then used for service customisation.

B. *Technology Framework*

The main elements of the FRESCO technology framework are the integrated development environment (IDE) and an integrated runtime environment (IRE). The FRESCO IDE is intended to support providers in the engineering phase of a composite service. For example, graphical tools will be provided for the specification, verification, and simulation of composite services. Service specification encompasses the composition logic, the aggregation logic, and coordination logic for the service. Verification encompasses formal proofs of service properties, such as behaviour and results produced. For example, the provider may need to verify that the service can reply to specific user requests. Simulation supports the investigation of aspects of a service for which formal proofs are not feasible. For example, simulation can be applied to investigate the reaction of the service to different loads of requests.

The FRESCO IRE includes mechanisms dedicated to the enforcement of composite services. The IRE encompasses the composition, coordination, and aggregation of the instances of service components involved in an instance of a composite service. The enforcement of the basic service model (service shell in particular) is outside the scope of the IRE. Nevertheless, specific indications and requirements will be given in order to ensure seamless integration between service-related and composition-related aspects of service enforcement. As an observation, the IRE will be built on established open technologies and standards (e.g. workflow management systems, databases, middleware platforms, Componentware). Principles such as encapsulation, loose coupling, and late binding will be driving the implementation of the IRE.

An overall methodology encompassing the use of the FRESCO technology framework will be provided. Particular attention will be devoted to the integration with technologies that are likely to be already available to service providers.

VI. Related Works

The notion of service is one of the most overloaded in the information technology arena. Focus on capabilities and abstraction from embodiment emerge as common elements in the different interpretations of a service. Still, service-oriented computing covers anything from the retrieval of an object reference to the booking of a flight. From a modelling perspective, the notion of service emerges as a natural evolution of the notion of object. Around the middle of the past decade, the reference model for open distributed processing (RM-ODP) [6] captured one of the first formalisations of the service notion. Previous research efforts such as Trade [7] converged into the RM-ODP.

The next big step for service-oriented computing was in 1998, with the HP proposal of the e-service vision [8]. The e-service vision set a consistent framework of business requirements on technical solutions centred on the service notion. The momentum created in the IT industry and in the research community is mainly reflected in the web service standard and technology stack [9]. The main limitation of the current web-service stack is that the service model is almost indistinguishable from a basic object model. For example, WSDL (Web Service Description Language) [10] covers the equivalent of method invocations. Still, current proposals such as WSCL [11] (Web Service Conversation Language) promise to raise the level of abstraction. Initiatives such as UDDI [12] (Universal Description Discovery and Integration) and the Semantic Web [13] are also targeting richer service models.

Composition is a central concept in the e-service vision. In terms of web-service stack, initiatives such as WSFL (Web Service Flow Language) [14] and XLANG [15] explicitly target service composition. The main issue is that the web-service component model constrains the scope of service composition. Independent from the web-service stack, recent works on service composition include PICCOLA [16], ICARIS [17], DynamiCS [3], E-Flow [18], and DySCo [4]. In relation to FRESCO, the main limitation we identify in such works lays in the service model assumed. Generality in the assumptions positively reflects on the generality of the results. The problem is that generality often prevents for direct applicability.

VII. Conclusions

Service composition is central to service-oriented computing. In FRESCO, we prose a service model that reflects the requirements deriving form HP e-service vision. Based on such model, we concentrate on the definition of a conceptual and technology framework that supports the creation and enforcement of composite services. In particular, we target the dynamic service composition of services. Fundamental characteristic of our proposal is the separation between composition, coordination, and aggregation logic for a composite service.

References

[1] AA.VV., *Gabler Wirtschafts-Lexikon*: Gabler, Wiesbaden, 1988.

[2] F. Griffel, M. Boger, H. Weinreich, M. Merz, and W. Lamersdorf, "Electronic Contracting with COSMOS - How to Establish, Negotiate and Execute Electronic Contracts on the Internet," in *2nd Int. Enterprise Distributed Object Computing Workshop (EDOC '98)*, Z. Milosevic, Ed.: IEEE, 1998, pp. 10.

[3] M. Tu, C. Seebode, F. Griffel, and W. Lamersdorf, "DynamiCS: An Actor-based Framework for Negotiating Mobile Agents," *Journal on Electronic Commerce Reserach, Special Issue on ,Agents In Electronic Commerce*, 2000.

[4] G. Piccinelli and L. Mokrushin, "Dynamic e-service composition in DySCo," in *Proc. Int. Workshop on Distributed Dynamic Multiservice Architecture, part of the 21st IEEE International Conference on Distributed Computing Systems (ICDCS-21), Phoenix, USA.*, 2001.

[5] G. Piccinelli, G. D. Vitantonio, and L. Mokrushin, "Dynamic service aggregation in electronic marketplaces," *Special Issue on Electronic Business Systems, Computer Networks Journal, Elsevier Science*, 2001.

[6] ISO/IEC-JTC1/SC21, "Basic Reference Model of Open Distributed Processing, Part 1: Overview. Draft International Standard 10746-1," International Organisation for Standardization 1995 1995.

[7] W. Lamersdorf, M. Merz, and K. Müller-Jones, "Middleware Support for Open Distributed Applications," in *Proceedings of the first International Workshop on High Speed Networks and Open Distributed Platforms, St. Petersburg*, 1995.

[8] H. Kuno, "Surveying the E-Services Technical Landscape," in *Proc. Second International Workshop on Advance Issues of E-Commerce and Web-Based Information Systems (WECWIS)*. Milpitas, California, USA: IEEE Computer Society, 2000.

[9] W3C, "Web Service Activity", http://www.w3.org/2002/ws/, 26.1.2002

[10] E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana, "Web Services Description Language (WSDL) 1.1, W3C Note", http://www.w3.org/TR/wsdl, 26.1.2002

[11] A. Banerji, "Web Services Conversation Language (WSCL) 1.0" W3C Note," 2002.

[12] UDDI, "Universal Description, Discovery and Integration", http://www.uddi.org, 26.1.2002

[13] A. Ankolekar, M. Burstein, J. R. Hobbs, O. Lassila, D. L. Martin, S. A. McIlraith, S. Narayanan, M. Paolucci, T. Payne, K. Sycara, and H. Zeng, "DAML-S: Semantic Markup For Web Services," in *Proceedings of the International Semantic Web Working Symposium (SWWS)*, 2001.

[14] F. Leymann, "Web Services Flow Language (WSFL 1.0)", http://www.ibm.com/software/solutions/webservices/pdf/WSFL.pdf, 26.1.2002

[15] S. Thatte, "XLANG - Web Services for Business Process Design", http://www.gotdotnet.com/team/xml_wsspecs/xlang-c/default.htm, 26.1.2002

[16] F. Achermann and O. Nierstrasz, "Applications = Components + Scripts - A Tour of Piccola," in *Software Architectures and Component Technology*, M. Aksit, Ed.: Kluwer, 2001, pp. 261-292.

[17] V. Tosic, B. Pagurek, and K. Patel, "E-Business Service Components with Multiple Classes of Service and Dynamic Adaptability Mechanisms", in," in *Proc. CITO Knowledge Network Conference 2001, Nepean, Canada*, 2001.

[18] F. Casati and M. C. Shan, "Dynamic and Adaptive Composition of E-Services," *Information Systems*, vol. 6, 2001.