A Development Process for Requirements Based Service Choreography

Marco Autili, Davide Di Ruscio, Paola Inverardi, James Lockerbie, Massimo Tivoli

A Development Process for Requirements Based Service Choreography

Marco Autili*, Davide Di Ruscio*, Paola Inverardi*, James Lockerbie† and Massimo Tivoli*
* *Università degli Studi di L'Aquila, Italy - {marco.autili,davide.diruscio,paola.inverardi,massimo.tivoli}@univaq.it*
† *City University London, UK - James.Lockerbie.1@city.ac.uk*

***Abstract*—The Future Internet envisions a ubiquitous world where available services can be easily discovered and coordinated so to fit users' requirements and needs. Service choreographies will play a central role in this vision as an effective means to allow heterogeneous services to suitably collaborate. This paper describes our experience of choreography development within the CHOReOS project.**

I. Introduction

The near future in software production envisions an ubiquitous world of available services that can be easily discovered and coordinated to fit users' needs. The Internet of Services paradigm [7] emerges from the convergence of the Future Internet (FI) and the service-oriented software paradigm. Services play a central role in this vision as effective means to achieve interoperability between heterogeneous parties of a business process and independence from the underlying infrastructure. At the same time they offer an open platform to build new value added service-based systems as a *choreography* of available services discovered within the FI.

Choreography is put forward as a generic abstraction of any possible collaboration among multiple services, and integrates previously established views on service composition, among which service orchestration. In principle, any possible view on collaboration and previously established view on composition among multiple services (e.g., service orchestration) can be abstracted as a choreography. Three often overlapping viewpoints and related terminologies can be distinguished [3], [8]: (1) *Choreography* captures collaborative processes involving multiple services and their interactions from a global perspective; (2) *Behavioral interface* captures the behavior of a single service that participates in the choreography (i.e., the signature and the interaction protocol with the environment that the service supports); and (3) *Orchestration* deals with the description of the interactions in which a set of services can be engaged, as well as the internal steps between these interactions.

In the FI, choreographies must be reliable and meet users' requirements. Additionally, they should be adaptive to context changes concerning both user-centric and resource-centric data, and quality of service (QoS). New methodologies are needed to address the specification, adaptation, synthesis, scalability, validation, simulation, dynamic execution, and re-configuration aspects of a choreography. The whole choreography development process is affected by the high dynamicity and evolution degree of a highly-scalable choreography. Thus, certain phases of the development process together with their proper artefacts, need to become pervasive, accompanying the choreography implementation at run-time.

This paper describes our experience in this direction and presents our approach to choreography development within the CHOReOS project (www.choreos.eu). The core objective of CHOReOS is to leverage, as far as possible, relevant SOA standards, while making choreography development a fully dynamic, scalable, and domain-expert-centric process so as to face the challenges posed by the Ultra Large Scale (ULS) FI.

Towards the definition of the choreography dynamic development process, CHOReOS focuses on the following (macro-)activities: (i) a domain-expert requirements specification framework for adaptable QoS-aware highly-scalable choreographies; (ii) methods and mechanisms for large scale service base management; (iii) choreography synthesis; (iv) service-oriented middleware for the FI to support choreography execution; and (v) analysis of choreography scalability, governance and V&V.

Although we provide an overview of the entire development process model (Section III), in this work we focus on the first and third activity only (Sections IV and V). Related state of the art is discussed in Section II. Conclusions and future work are given in Section VI.

II. Related Work

An example of a dynamic development process model has been proposed within the FP6 IST EU PLASTIC project (www.ist-plastic.org/). The PLASTIC development process relies on Model Driven Development (MDD) [17] solutions to build adaptable context-aware service-oriented applications. The CHOReOS dynamic development process (and its related tools), is inspired by the PLASTIC development process model and will be based on the models@runtime approach [4]. We will follow a conceptual-model-centric strategy, as in PLASTIC, and we will exploit model transformation technologies in order to make models available, and synchronized, at each phase of the process life-cycle from

This work is supported by the European Community's Seventh Framework Programme FP7/2007-2013 under grant agreement number 257178 (project CHOReOS - Large Scale Choreographies for the Future Internet - www.choreos.eu).

design- to run-time, as in the models@runtime approach. Existing research on models@runtime mainly focuses on models representing software structures and procedural representations of them such as architectural models. As pointed out in [4], the models@runtime vision should go further by raising the level of run-time model abstraction to that of requirements. For instance, a research challenge concerns the development of methods and tools for observing the systems requirements and goals during execution. To this aim, the domain-expert specification framework proposed by CHOReOS (see Section IV) represents a step forward.

In general, requirements engineering research has not recognized the importance of open systems and service-oriented computing, considering it to be downstream from requirements activities, although some work has been undertaken (e.g., [9]). However, most of this work has not directly addressed recognized challenges in service-oriented computing. Some of the most relevant work in service-oriented computing has been in QoS ontologies and measurement [16], however this work focuses more on service-level qualities and SLA, but not the originating requirements and their expression. In [13], the authors propose a general framework for automatic Web service composition. Waldinger [18] elaborates an idea for service synthesis by theorem proving. Most end-users, however, will lack the education and/or training to be able to express requirements using even semi-formal notations supported by UML or first-order languages. Most functional requirements will be expressed in, at best, structured natural language. Zachos et al. [19], for example, present techniques to match semi-structured natural language QoS requirements to service qualities during service selection using ontologies.

Current approaches for synthesizing service choreographies, involve having a central unit responsible for selecting and composing the system [5], [12], [15] (just to cite some). The centralization is a clear drawback when considering the expected Ultra Large Scale of the Future Internet. Dealing with the synthesis phase of decentralized choreographers is among the main objectives of the CHOReOS dynamic development process. We define choreography patterns by considering as baseline our previous work described in [2], so that the choreography may be distributed in a set of wrappers. We will introduce model transformation techniques [11] to refine the domain-expert choreography requirements specification into a peer-style specification of the choreography. Thus, the choreography synthesis process shall result in producing the decentralized model of an abstract *coordination delegate* that, accounting for service functional and non-functional abstractions, enforces the composition of the discovered services to adhere to the specified choreography.

III. CHOReOS DYNAMIC DEVELOPMENT MODEL

The evolutionary nature of choreographies in the Future Internet makes unfeasible a standard development process and all the artefacts and models might be exploited also by the deployment- and run-time activities hence leading to a dynamic development process model. Therefore, central to the development process is the notion of models and models@runtime techniques. The CHOReOS Development Process Model, decomposes into:

- First, CHOReOS intends to support the systematic development of choreographies from their design to their actual enactment, i.e., execution. CHOReOS uses two orthogonal model-transformation approaches: (i) top-down and (ii) cross-cutting. The former will serve to refine the natural language-based domain-expert requirements specification into analysis-technique-specific models derived from a peer-style specification. The latter will serve to integrate the different modeling/reasoning technologies by passing from a technique-specific model to a different technique-specific model.
- A distributed (large scale) service base will further manage information about available services offered by service providers. The service base organizes available services into functional and non-functional views. Each view is characterized by corresponding abstractions (functional/non-functional) and a set of available services that are represented by each abstraction.
- The approach envisioned for the choreography synthesis then starts from (*i*) the peer-style specification of the choreography and from (*ii*) the set of services discovered from the large scale service base. Input (*i*) is based on the functional and non-functional abstractions, and it comes from the refinement of the domain expert specification obtained by means of the top-down transformational process. Input (*ii*) comes from the exploitation of the service base management mechanisms. That is, the synthesis process assumes that the services into the registry/base have been discovered so that they satisfy the local (to the service) functional and non-functional requirements that have been specified for the choreography and, hence, can be considered as potential candidates to participate in the global choreography process. The choreography synthesis produces the abstract coordination delegates that, on top of the CHOReOS middleware, support the enactment of choreographies in a distributed way.
- Finally, the process accounts for complex analysis and validation steps at run-time.

IV. REQUIREMENTS SPECIFICATION

As neither the role of domain expert or service consumer can assume requirements knowledge, a core challenge to address is how quality requirements on future services can be expressed as simply as possible. The most straightforward form of expression of requirements from these two roles is natural language. However, some degree of structure to free-text expression is needed to capture the service quality

needs of the stakeholders effectively. To do this we use qualifiers [1], which transform the functional root of a requirement into a non-functional requirement (NFR). Rather than asking stakeholders to express requirements using typical NFR types our approach prompts them to express qualities on the service through 5 simple questions: "How quick?", "How secure?", "How precise?", "How robust?", and "How simple to use?". These questions were selected based on a model of qualities and measures developed for the Cloud Commons Consortium (www.cloudcommons.com) and revised for the CHOReOS project. Figure 1 presents a simple representation of the model, and shows how the qualifier questions map onto service-based application qualities and QoS on services aggregated in these applications. It would not be realistic for domain experts and service consumers to express requirements on qualities such as scalability, flexibility and serviceability as these areas are the concern of service developers and providers. However, through associations in the model these can be implied, for example, high performance may require a particular level scalability and flexibility of the services.

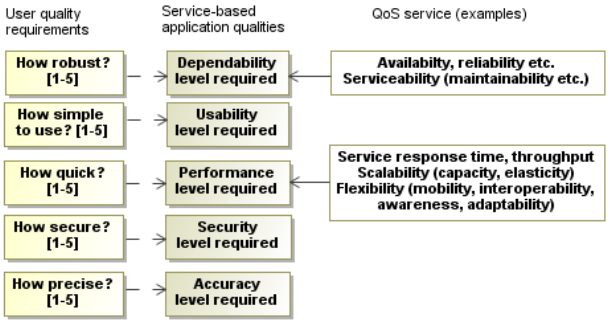

Figure 1. Simplified overview of the CHOReOS quality model

Our approach also uses the concept of customer satisfaction from the VOLERE specification [14] to measure how happy the user will be if the requirement is implemented and how unhappy they will be if it is not. All user responses on quality and satisfaction are prioritized relative to each other using a sliding scale recorded notionally from 1 to 5. Prioritization of requirements is particularly useful when considering scalability and the potentially hundreds of requirements for domain experts to manage.

The specification process (see Figure 2) begins with the service consumer who expresses requirements on their service needs. The domain expert then receives consumer requirements through the evolving requirements specification and can also act as a surrogate for individual service consumers by expressing their own domain specific requirements. The domain expert can constantly review the latest requirements/queries to extract the overall requirements on the services. Domain expert analysis helps to identify emerging patterns or similarities based on functionality, quality and user type or preference. The resulting set of prioritized

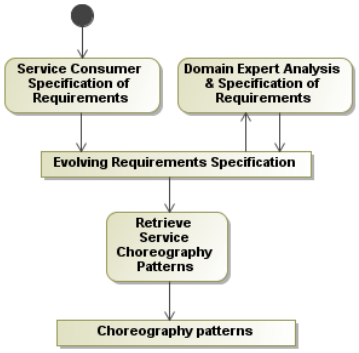

Figure 2. Goal and requirements specification

quality-based requirements is then associated with choreography strategies, which are expressed in the form of patterns.

The service choreography patterns encapsulate different complex choreography decisions made in the presence of user requirements, by linking codified domain knowledge of user tasks to classes of service solutions. The user requirements are matched to user task models that, in turn, are used to reformulate service queries with terms describing classes of services in the task model. Each pattern considers the qualities of the services to be invoked, the qualities that different choreographies offer and how, when combined, these service and choreography qualities deliver service-oriented applications with different qualities.

V. Choreography Synthesis

A choreography can be seen as a collaborative process among interacting participants. The message exchange globally defined by the choreography has to be projected (in a peer-style fashion) among different participants according to the "local" roles they play to fulfill the global choreography. Choreography synthesis concerns the realization of distributed coordination delegates that, supported by the CHOReOS middleware, cooperatively work to support the enactment and execution of the choreography. By relying on the CHOReOS service base management mechanisms, *Model-to-Model Transformation* techniques [10] are used to refine the *Choreography Model* together with the service *Behavioral Models* into a *Peer-style Specification* (see Figure 3).

Service behavioral models are automatically synthesized by the *Synthesis of Behavior Protocol* activity starting from the descriptions of services that have been discovered within the *Service Base*. The synthesis process is able to understand how to actually coordinate the discovered services to suitably realize the specified choreography. For instance, it might be the case that the discovered services, although potentially suitable in isolation, when interacting together can lead to, e.g., concurrency and interaction mismatches such as deadlocks, or safety and timeliness violations. Thus, although a given set of services, if coordinated in the right way, can be used to achieve the specified choreography, they can completely miss the choreography's goal if coordinated in a different way. Further applying *Model-to-Model*

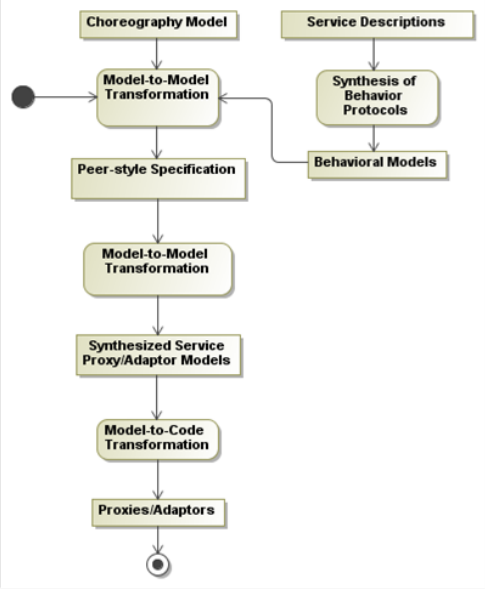

Figure 3. Choreography synthesis

Transformation, the synthesis method will produce abstract coordination delegate models. Such models, accounting for service functional and non-functional abstractions, force the collaboration of the discovered services to guarantee the specified choreography. To actually realize the choreography, the abstract coordination delegate models are then concretized into actual software artifacts by means of *Model-to-Code Transformation*s. The generated coordination delegates are deployed on the CHOReOS choreography engine and, by means of the CHOReOS service-oriented middleware, they suitably access and coordinate the discovered services.

In the CHOReOS deliverable D3.1 [6], we discuss an example concerning the development of a choreography-based travel agency system that can be realized by choreographing four services: a `Booking Agency` service, two `Flight Booking` services, and a `Hotel Booking` service. The example shows how coordination delegates correctly and distributively interact, at run-time, with each other, hence ensuring the behavior globally specified by the *Choreography Model*. For space reasons, it is not possible to describe this example here and, hence, for it we entirely refer to [6].

VI. Conclusions and Future work

In this paper we have briefly described our initial experience of choreography development within the CHOReOS project (www.choreos.eu). The core objective of CHOReOS is to leverage relevant SOA standards, while making choreography development a fully dynamic, scalable, and domain-expert-centric process so as to face the challenges posed by the FI. After providing an overview of the entire development process model, we have focused on requirements specification and choreography synthesis.

As future work we plan to instantiate the process model and apply it to the case studies we have promised to target in CHOReOS.

References

[1] I. Alexander and R. Stevens. *Writing Better Requirements*. Addison-Wesley, 2002.

[2] M. Autili, L. Mostarda, A. Navarra, and M. Tivoli. Synthesis of decentralized and concurrent adaptors for correctly assembling distributed component-based systems. *JSS*, 81(12), 2008.

[3] A. Barros, M. Dumas, and P. Oaks. Standards for web service choreography and orchestration: Status and perspectives. In C. Bussler and A. Haller, editors, *Business Process Management Workshops*, volume 3812 of *Lecture Notes in Computer Science*, pages 61–74. Springer Berlin / Heidelberg, 2006.

[4] G. Blair, N. Bencomo, and R. B. France. Models@run.time. *Computer journal*, 42, 2009.

[5] A. Brogi and R. Popescu. Automated generation of bpel adapters. In *CIbSE'07*, 2007.

[6] CHOReOS Project Team. Deliverable D3.1: Initial Architectural Style for CHOReOS Choreographies, September 2011.

[7] CORDIS ICT Programme. Service and Software Architectures, Infrastractures and Engineering (SSAI).

[8] R. Dijkman and M. Dumas. Service-oriented design: A multi-viewpoint approach. *International Journal of Cooperative Information Systems*, 13:337–368, 2004.

[9] S. Fickas and M. Feather. Requirements monitoring in dynamic environments. *Proc. of RE'95*, 1995.

[10] G. Huang, H. Mei, and F.-Q. Yang. Runtime recovery and manipulation of software architecture of component-based systems. *ASE journal*, 13, 2006.

[11] H. Mei, G. Huang, L. Lan, and J. Li. A software architecture centric self-adaptation approach for internetware. *Science in China Series F: Information Sciences*, 51(6), 2008.

[12] T. Melliti, P. Poizat, and S. B. Mokhtar. Distributed behavioural adaptation for the automatic composition of semantic services. In *Proc. of FASE'08/ETAPS'08*, 2008.

[13] J. Rao and X. Su. A survey of automated web service composition methods. In *Proc. of SWSWPC'04*, 2004.

[14] S. Robertson and J. Robertson. *Mastering the Requirements Process*. Addison-Wesley, 1999.

[15] G. Salaün. Generation of service wrapper protocols from choreography specifications. In *Proc. of SEFM'08*, 2008.

[16] P. Sawyer, J. Hutchinson, J. Walkerdine, and I. Sommerville. Faceted service specification. In *Proc. of SOCCER'05*.

[17] B. Selic. The pragmatics of model-driven development. *IEEE Softw.*, 20, 2003.

[18] R. Waldinger. Web agents cooperating deductively. In *Formal Approaches to Agent-Based Systems*, LNCS1871. 2001.

[19] K. Zachos, G. Dobson, and P. Sawyer. Ontology-aided translation in the comparison of candidate service quality. In *Proc. of SOCCER '08*, 2008.