



Large Scale Choreographies for the Future Internet

ICT IP Project

Deliverable D1.1

CHOReOS State of the Art, Baseline, and Beyond

<http://www.choreos.eu>

THALES



INSTITUT FÜR KUNST- UND
KULTURWISSENSCHAFTEN
AN DER UNIVERSITÄT DUISBURG
ESSEN



Project Number	:	FP7-257178
Project Title	:	CHOReOS Large Scale Choreographies for the Future Internet

Deliverable Number	:	D1.1
Title of Deliverable	:	CHOReOS State of the Art, Baseline, and Beyond
Nature of Deliverable	:	Report
Dissemination level	:	Public
Licence	:	Creative Commons Attribution 3.0 License
Version	:	3.0
Contractual Delivery Date	:	01/01/11
Actual Delivery Date	:	21/01/11
Contributing WP	:	WP1
Author(s)	:	Valérie Issarny (INRIA) Antonia Bertolino, Guglielmo De Angelis (CNR) Amira Ben Amida, Jean-Pierre Lorré (EBM) Nikolaos Georgantas, Valérie Issarny, Animesh Pathak (INRIA) James Lockerbie, Neil Maden (CITY) Marco Autili, Davide Di Ruscio, Massimo Tivoli (UDA) Dionysis Athanasopoulou, Panos Vassiliadis, Apostolos Zarras (UOI) Alfredo Goldman, Marco Gerosa, Fabio Kon (USP) Teodoro De Giorgio, Gianmarco Panza, Maurilio Zuccalà (CEFRIEL) Andrea Polini (UNICAM)
Reviewer(s)	:	Alfredo Goldman, Fabio Kon (USP) Marco Autili (UDA)

Abstract

The D1.1 deliverable clarifies baseline, progress, and state of the art that CHOReOS will address.

For each of the first four CHOReOS work packages, WP1 to WP4, this deliverable gives a precise definition of the state of the art, an indication of the envisaged progress beyond the state of the art by CHOReOS and the baseline for its research.

Keyword list

Future Internet, service-oriented computing and architecture, choreography, service composition, conceptual model, meta-modeling, software architecture, model-driven engineering, process synthesis, BPMN2, middleware, ESB, grid and cloud computing, pervasive computing, Internet of smart things, verification and validation.

Document History

Version	Changes	Author(s)
V1.0	First version of outline and integration of material from the DoW	Valérie Issarny, INRIA
V1.1	Individual contributions to sections	All authors
V2.0	First integrated version	Valérie Issarny, INRIA
V2.1	Revised section by authors following internal review	All authors
V2.2	Second integrated version	Valérie Issarny, INRIA
V3.0	Final integrated version	Valérie Issarny, INRIA

Document Review

Review	Date	Ver.	Reviewers	Comments
Outline	14/10/10	1.0	All authors	OK by all
Draft	22/12/10	2.0	M. Autili, A. Goldman, V. Issarny, F. Kon	Complete version released for final review
QA	14/01/11	3	A. Bertolino, V. Issarny, F. Kon, H. Vincent,	Editorial comments
PTC	21/01/11	A	PTC	

Glossary, acronyms & abbreviations

Item	Description
a.k.a.	Also known as
API	Application Programming Interface
BPEL	Business Process Execution Language
BPEL4SWS	Business Process Execution Language for Semantic Web Services
BPMI	Business Process Management Initiative
BPMN	Business Process Modeling Notation
BPSS	Business Process Specification Schema
CA	Consortium Agreement
CBDI	Component Based Development and Integration
CBDI-SAE	CBDI Service Architecture and Engineering
DAML	DARPA Agent Markup Language
DAML-S	DARPA Agent Markup Language for Services
DCV&V	Decentralized Collaborative Validation and Verification
DL	Deliverable Leader
DoDAF	Department of Defense Architecture Framework
DOW / DoW	Description of Work
DSB	Distributed ESB
EAI	Enterprise Application Integration
EDA	Event-Driven Architecture
EFSM	Extended Finite State Machine
ESB	Enterprise Service Bus
ETP	European Technology Platform
FI	Future Internet
FIA	Future Internet Assembly
GRL	Goal Representation Language
HTTP	HyperText Transfer Protocol
IaaS	Infrastructure as a Service
IAC	Industrial Advisory Committee
IDE	Integrated Development Environment
IDRE	Integrated Development and Runtime Environment
IT	Information Technology
JB1	Java Business Integration
MBT	Model-Based Testing
MDD	Model Driven Development
MODAF	Ministry of Defence Architecture Framework
MST	Management support team
M2M	Model-to-Model
M2T	Model-to-Text
OMG	Object Management Group

Item	Description
OSGi	Open Services Gateway Initiative
OSS	Open Source Software
OWL	Ontology Web Language
OWL-S	Ontology Web Language for Services
PaaS	Platform as a Service
PL	Project Leader
PMT	Project Management Committee
PO	Project Officer
PTC	Project Technical Committee
QoS	Quality of Service
R&D	Research & Development
RDF	Resource Description Framework
REST	REpresentational State Transfer
RTD	Research and Technology Development
SaaS	Software as a Service
S-APL	Semantic-Agent Programming Language
SAWSDL	Semantically Annotated Web Service Description Language
SBPM	Semantic Business Process Management
SL	Scientific Leader
SLA	Service Level Agreement
SLS	Service Level Specification
SOA	Service-Oriented Architecture
SoaML	Service-oriented architecture Modeling Language
SOAP	Simple Object Access Protocol
SOC	Service-Oriented Computing
SoTA / SOTA	State of The Art
SQL	Structured Query Language
TCC	Test Collaboration Contracts
TDD	Test-Driven Development
TSC	Testing Service Contract
UDDI	Universal Description Discovery and Integration
ULS	Ultra-Large-Scale
ULS-FI	Ultra-Large-Scale Future Internet
UML	Unified Modeling Language
UMM	UN/CEFACT Modeling Methodology
UPDM	Unified Profile for DoDAF and MODAF
V&V	Verification & Validation
W3C	World Wide Web Consortium
WP	Work Package
WPL	Work Package Leader

Item	Description
WS	Web Service
WSA	Web Service Architecture
WS-CDL	Web Services Choreography Description Language
WSCE	Web Service Crawler Engine
WSCI	Web Service Choreography Interface
WSDL	Web Service Description Language
WSDL-S	Web Service Description Language with Semantics
WSDM	Web Services Distributed Management
XML	eXtensible Markup Language
XSLT	XML Stylesheet Language for Transformations

Table of Contents

1. Introduction	1
2. WP1: Architectural Style for Choreography-based Future Internet.....	3
2.1. <i>CHOReOS Conceptual Model for Choreography-based Future Internet</i>	3
2.1.1. <i>State of the art</i>	3
2.1.2. <i>Baseline</i>	5
2.1.3. <i>Progress.....</i>	6
2.2. <i>CHOReOS Architectural Style for Choreographies in the Future Internet.....</i>	7
2.2.1. <i>State of the art</i>	8
2.2.2. <i>Baseline</i>	10
2.2.3. <i>Progress.....</i>	11
3. WP2: Dynamic Development of Adaptable, QoS-aware ULS Choreographies	13
3.1. <i>Dynamic Development Process & Tools.....</i>	13
3.1.1. <i>State of the art</i>	13
3.1.2. <i>Baseline</i>	14
3.1.3. <i>Progress.....</i>	15
3.2. <i>Large Scale Service Base Management.....</i>	16
3.2.1. <i>State of the art</i>	16
3.2.2. <i>Baseline</i>	18
3.2.3. <i>Progress.....</i>	19
3.3. <i>Domain Expert Requirements Specification and Choreography Modelling.....</i>	20
3.3.1. <i>State of the art</i>	20
3.3.2. <i>Baseline</i>	21
3.3.3. <i>Progress.....</i>	22
3.4. <i>Adaptable Choreography Synthesis.....</i>	23
3.4.1. <i>State of the art</i>	23
3.4.2. <i>Baseline</i>	24
3.4.3. <i>Progress.....</i>	25
3.5. <i>A Model for Choreographies: Analyzing and Predicting Choreography Scalability</i>	25
3.5.1. <i>State of the art</i>	25
3.5.2. <i>Baseline</i>	26
3.5.3. <i>Progress.....</i>	26
4. WP3: Service-Oriented Middleware for the Future Internet.....	27
4.1. <i>Service-oriented Middleware</i>	27
4.1.1. <i>State of the art</i>	27
4.1.2. <i>Baseline</i>	28
4.1.3. <i>Progress.....</i>	29
4.2. <i>Distributed Service Bus.....</i>	29
4.2.1. <i>State of the art</i>	30
4.2.2. <i>Baseline</i>	31
4.2.3. <i>Progress.....</i>	31
4.3. <i>Middleware Solutions for the Internet of Smart Things</i>	32
4.3.1. <i>State of the art</i>	32
4.3.2. <i>Baseline</i>	34
4.3.3. <i>Progress.....</i>	34
4.4. <i>Scalable Service Provisioning using Grid and Cloud Computing Technology</i>	35

4.4.1.	<i>State of the art</i>	35
4.4.2.	<i>Baseline</i>	36
4.4.3.	<i>Progress</i>	37
4.5.	<i>Choreography Adaptation</i>	37
4.5.1.	<i>State of the art</i>	37
4.5.2.	<i>Baseline</i>	40
4.5.3.	<i>Progress</i>	40
5.	WP4: Governance and V&V Support for Choreographies for the Future Internet.	43
5.1.	<i>Governance</i>	43
5.1.1.	<i>State of the art</i>	43
5.1.2.	<i>Baseline</i>	46
5.1.3.	<i>Progress</i>	47
5.2.	<i>Verification & Validation</i>	50
5.2.1.	<i>State of the art</i>	50
5.2.2.	<i>Baseline</i>	52
5.2.3.	<i>Progress</i>	53
6.	References	57

1. Introduction

As detailed in the project's Description of Work (DoW), 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 Future Internet (ULS-FI).

Towards the above objective, CHOReOS revisits the concept of choreography-based service-oriented systems, to introduce a dynamic development process and associated methods, tools, and middleware – referred to as the CHOReOS Integrated Development and Runtime Environment (IDRE) – for coordinating services in the ULS-FI.

Work to be undertaken within CHOReOS includes the study of the following elements of the IDRE, which are the focus of dedicated Work Packages (WPs):

- Architectural style for choreography-based Future Internet, investigated within WP1;
- Support for the dynamic development of adaptable QoS-aware ULS choreographies addressed by WP2;
- Service-oriented middleware for the Future Internet developed within WP3; and
- Governance, and Verification and Validation (V&V) support for choreographies for the Future Internet examined within WP4.

Actual development of the IDRE, based on the integration of the above core elements, will be the focus of WP5. The CHOReOS IDRE will additionally build upon well known IDEs (Integrated Development Environments) such as Eclipse to deal with low-level programming (e.g., implementing the components and Web services in Java) and Web-based environments for higher-level specifications and operations (e.g., defining and enacting choreographies, and monitoring the execution of their services).

The purpose of this document is to clarify the baseline, progress, and state-of-the-art that CHOReOS will address, concentrating on the aforementioned core RTD WPs, i.e., WP1 to WP4. As such, the deliverable provides an updated version of the content provided in Section B1.2 (“Progress beyond state of the art”) of the project’s DoW.

This document is structured in 4 sections, which respectively concentrate on WP1 to WP4. Each section is further decomposed into a number of subsections, one for each of the main scientific and technological areas investigated (as identified by the WP tasks) for which we set: (i) State of The Art (SoTA), (ii) baseline built upon and (iii) progress to be made by CHOReOS.

2. WP1: Architectural Style for Choreography-based Future Internet

The goal of WP1 is to unambiguously characterize the paradigms underlying the Future Internet according to the CHOReOS vision, with a special emphasis on enabling choreographies in such Ultra-Large-Scale (ULS) networking environments.

A main objective of WP1 is then to conceptually characterize, in terms of a *conceptual reference model* (Section 2.1), the many facets of the choreography-based Future Internet, i.e., service-oriented systems realized as (decentralized) collaborations of services discovered in ULS networking environments. Further, based on this conceptual model, the goal is to define a *reference architectural style* (Section 2.2) for the design, development, and validation of choreography-based large-scale services in the Future Internet.

Thus, the conceptual model will serve as input to the development of the RTD work packages WP2, WP3, and WP4. The definition of the conceptual model will conversely be refined from the work conducted in these three work packages, hence benefiting from the lessons learned from concrete instantiation as part of the CHOReOS technological development.

2.1. CHOReOS Conceptual Model for Choreography-based Future Internet

During the last few years, the concept of choreography has received a growing interest as a means for implementing complex service-oriented systems. This interest has led to the proliferation of different interpretations. In principle, any possible view on collaboration and previously established composition among multiple services (e.g., service orchestration) can be abstracted as a choreography. As already pointed out in the CHOReOS DoW, three often overlapping viewpoints and related terminologies can be distinguished [DD04, BDO06]: (1) *Choreography* captures collaborative processes involving multiple services and their interactions from a global perspective; (2) *Behavioural* interface captures the behaviour 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 given service can engage with other services, as well as the internal steps between these interactions.

The CHOReOS conceptual reference model, which will capture the relevant entities, concepts, and the relationships among them, will characterize the choreography-based Future Internet. This conceptual model will provide a high-level common ground, which will support the common understanding among all the project partners (and also end-users) and will make communication easier and unambiguous. The model will also provide the fundamentals for the reference architectural style.

Towards this direction, our SoTA analysis has focused on work related to: (i) the Future Internet definition, (ii) the conceptual characterization of service-based systems, and (iii) the languages underpinning choreography specifications.

2.1.1. State of the art

The Future Internet, as a particular case of a ULS system, constitutes a futuristic vision of a yet-to-come Internet, in which “scale changes everything” [ULS06].

The Future Internet vision

In the context of the European Economic Recovery Plan, significant attention is being paid to Future Internet R&D activities. It is worth mentioning the Future Internet Assembly (FIA), which groups 100 projects to coordinate their R&D activities to foster a strong European

footprint on Future Internet research. The main objective is to develop open, standardized, cross-sector service platforms. From a European policy perspective, sectors such as healthcare, mobility, environment, and energy management are prime candidates to benefit from novel, “smart”, Internet-empowered infrastructures, which will facilitate the rapid take-up and adoption of services by millions of users and consumers.

The work described in the Cross-ETP Vision Document [Cross-ETP] illustrates the overall objectives and ambitions underlying the trend towards the Future Internet. In particular, the authors of [Cross-ETP] devise a strategy and action plan that will make the Future Internet an industrial, economic, and societal success in Europe.

Potential problems and risks of the Future Internet as well as potential opportunities and benefits with a new networking approach are discussed in [FIA-MANA]. Some concrete usage scenarios of the Future Internet are proposed in [FIAFCN07,FIA-FISO,FIA-RWI,FIRE09,FIA-FISE]. For each scenario, the authors give description, functional requirements, potential barriers and problems, R&D challenges, as well as potentials for business innovation. Concrete research challenges are identified also in particular scenarios where trust and identity aspects have to be taken into account [FIA-TI].

Conceptual models for service-oriented systems

The implications of building large-scale choreographies for Future Internet scenarios in line with the CHOReOS vision need to be understood. Thus, their key characteristics need to be devised by taking into account the state of the art conceptual models and languages for service-oriented systems at large and service-oriented choreography in particular, in relation with the key features of the Future Internet.

A number of conceptual characterizations have been proposed in the literature for modelling and describing service-based systems. Roughly speaking, these approaches strive towards the same goal but at different levels of abstraction and with different specific purposes. The most recent approach is the NEXOF Reference Model [NEXOFRM]. Specifically, considering Future Internet as an application domain, the NEXOF Reference Model was born by considering an integrated comprehensive view of all the previous approaches. In particular, it can be seen as a holistic model that accounts for the Web Service Architecture [BHMN+07], the CBDI Model [CBDI], the SeCSE Model [CDDD+05,SeCSE] (to which the CHOReOS partner CITY contributed), the OASIS Reference Model [MLMB+06], the EGA Reference Model [EGA1,EGA2], the TrustCoM Conceptual Models [SWHA+05], the Service Component Architecture [BBBE+07], the Service-oriented architecture Modelling Language [SoaML], and PLASTIC [PlasticWS,Plastic08] (to which the CHOReOS partners INRIA, CNR-ISTI, and UDA contributed).

Languages for choreography specification

As already reported in the DoW, several languages that underpin the specification of service compositions and collaborations have been proposed, like the W3C Web Services Choreography Description Language (WS-CDL) [WSCDL], and the OASIS and UN/CEFACT ebXML Business Process Specification Schema (BPSS) [BPSS]. WS-CDL has been a W3C candidate recommendation since 2005, but its Working Group has closed in 2009 due to the lack of implementations of the standard. Still, the Web Services Business Process Execution Language, BPEL, [WS-BPEL], which is the major widely accepted industry standard for modelling service composition, overtaken by OASIS, can be conveniently exploited in the context of choreography modelling to specify abstract processes for behavioural interfaces. BPEL4Chor [DKLW07] further extends BPEL for defining choreographies by introducing an interconnection layer on top of abstract BPEL processes, thus leading to interconnected behavioural interface descriptions. As BPEL itself is used unchanged, the BPEL4Chor extensions support a seamless integration between service choreographies and orchestrations. Let's Dance [ZBDH06] is a visual choreography language derived from

workflow and architecture description languages and targeted at business analysts. Acknowledging the need to give unambiguous semantics to proposed choreography models and languages, there have been several research approaches on defining the formal semantics of choreographies with a special focus on WS-CDL [LHZIP07, CZ08, ZLMZ+08, YZCQ08], although other languages also received attention, such as WSCI [HXWX+07].

Formal models and languages have been introduced with the aim of verifying conformance between the global choreography and local behavioural interfaces; in this way, the ability of services to take part in a given choreography can be verified. Related research approaches have addressed: consistency checking between ebXML BPSS choreography and BPEL orchestration based on CSP [Yeu08]; reducing the complexity of conformance verification based on an extension to the pi-calculus for enabling resource-constrained mobile devices to participate in a choreography [STDD07]; and verifying WS-CDL choreographies against service behaviours expressed in BPEL based on Finite State Processes and Labelled Transition Systems [FUMK06].

From the methodological point of view, the OASIS and UN/CEFACT Modelling Methodology (UMM) [UMM,HHLS+06] is based on UML (it is defined as a UML profile) and addresses the analysis of the business environment, the requirements of each partner, and the requirements for an inter-organizational collaboration. The OMG Business Process Management Initiative (BPML) further considers the transition from the business process design (supported by the OMG Business Process Modelling Notation (BPMN) [BPMNv1.2]) to the business process implementation. Thus, the mapping from BPMN to XML-based executable business processes (e.g., BPEL) is part of the standard. As part of the related OMG continuous standardization process, BPMN 2.0 will be the next version of BPMN and is currently under working status [BPMNv2.0]. This update aims to enrich BPMN with a comprehensive embedded meta-model and related graphical notation and interchange format. This will improve the capability for business analysts to develop, communicate, and understand business process models, and also related tool development. The Service oriented architecture Modelling Language (SoaML) specification [SoaML] provides a meta-model and a UML profile for the specification and design of services within a service-oriented architecture. In particular, SoaML focuses on the basic service modelling concepts and the intention is to use this as a foundation for further extensions related to the integration with other OMG meta-models like BPMN 2.0.

Last but not least, a variety of transformations between models and languages have been proposed for enabling top-down, model-driven generation of behavioural interfaces from choreographies. In [HH08], UMM is projected to a local orchestration model developed as a UML profile; then, this model is transformed to an executable orchestration expressed in BPEL. The work described in [REMPD07] integrates QoS aspects in a top-down choreography development process, annotating WS-CDL choreographies with SLAs and further generating a BPEL orchestration for each partner along with WS-QoS policies that can be enforced during the orchestration execution.

2.1.2. Baseline

CHOReOS will build upon the latest SoTA research and technologies in the area of Future Internet understanding and definition, service-based system conceptual characterizations, and choreography modelling and specifications. In particular, CHOReOS will build upon the aforementioned BPMN 2.0 whilst accounting for the prevalence of the BPEL standard and its relevance for specifying the respective behaviour of choreographed nodes. Still, the modelling of choreographies should abide by the requirements imposed by the ULS Future Internet. Thus, the conceptual model shall identify the relevant Future Internet entities and concepts as well as the relationships among them while rigorously developing highly scalable choreographies.

The definition of the conceptual model for the choreography-based Future Internet will build upon previous efforts towards the conceptual characterization of service-oriented systems. Indeed, the corner stone for the CHOReOS Conceptual Model will be the NEXOF Reference Model [NEXOFRM]. This choice is motivated by the fact that the NEXOF Reference Model constitutes a comprehensive and integrated view of all the previous approaches further refined and enriched by also considering Future Internet concerns.

The conceptual model will be described by using the UML-like notation adopted to define the NEXOF Reference Model. The conceptual model will also be described textually to allow a better understanding of the choreography-based Future Internet domain and of the challenges that must be faced while defining a dynamic development process for adaptable QoS-aware scalable choreographies. In fact, as further argued in Section 3.1, the notion of *models* built upon concepts precisely collected in the conceptual model is central to the CHOReOS development process. Thus, taking as input the UML-like-based description, the conceptual model can be concretely realized as, e.g., MOF-compliant (meta-)models. For instance, the part of the conceptual model that is related to the specification of service choreographies will be constructed by extending the existing BPMN 2.0 language. In particular, we will extend it to add new modelling constructs in the existing choreography package to support specific aspects of the Future Internet, which are currently neglected.

2.1.3. Progress

Towards a significant progress with respect to the above discussed SoTA and baseline, CHOReOS revisits the paradigms of choreography-based service-oriented systems to face the challenges of the ULS Future Internet. Beyond dealing with the ULS issue, that is a novelty *per se*, such a goal includes introducing, into the conceptual model, dedicated key concepts that comprise user-centrism, decentralization, compositionality, adaptation, and QoS-awareness, which are not comprehensively accounted for by the SoTA and baseline work. Based on the conceptual model, CHOReOS shall provide an architectural style that identifies reference components, connectors, and coordination patterns specific for the highly scalable choreographies of the Future Internet. This work, which will be undertaken as part of the work package WP1, will then serve as a basis for the CHOReOS development process and supporting IDRE, spanning methods, tools, and middleware specifically conceived for sustaining choreographies in the ULS Future Internet.

Concerning the specification of service choreographies, CHOReOS will provide a modeling environment based on an extension of the BPMN 2.0 meta-model. In particular, BPMN was designed as a simple language covering a single view of business process modeling. While it is a perfect tool for that purpose, it lacks the capabilities to integrate the processes with the other aspects of Enterprise Architectures such as information model, organization roles, and software systems that provide automation for some of the business tasks. Most of these Enterprise Architectures aspects can be captured in UML or its profiles oriented to specific frameworks, such as the Unified Profile for DoDAF and MODAF (UPDM) [UPDM]. In order to integrate BPMN models as a part of such Enterprise Architectures models in UML tools, it is necessary to support BPMN 2.0 as a UML profile. CHOReOS intends to define a UML profile for BPMN 2.0 and contribute to OMG for standardizing a UML Profile for BPMN Processes.

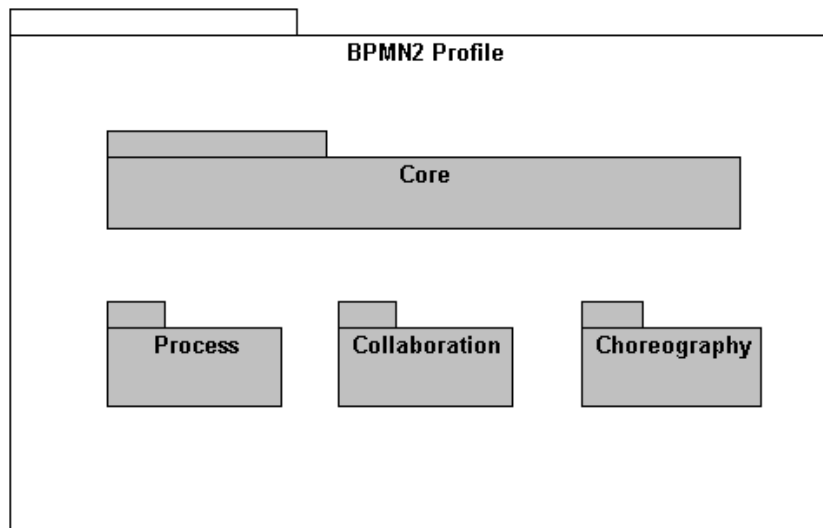


Figure 1: BPMN2 first-level package overview

As shown in Figure 1, the BPMN meta-model consists of four “first level” packages:

- The *Core package* provides a set of basic elements identified as core elements of the specification.
- The *Process package* defines how to model processes - a sequence or flow of activities in an organization with the objective to accomplish a task. In BPMN, a process is depicted as a graph of flow elements, which are a set of activities, events, gateways, and sequence flows that allow for defining the execution semantics of processes.
- The *Collaboration package* shows how to depict the interactions between two or more business entities. Collaboration modelling focuses on the participants shown as pools and their interactions.
- The *Choreography package* focuses on modelling interactions between business participants. A choreography is a type of process, but differs in purpose and behaviour from a standard BPMN process. A standard process defines the flow of activities of a specific partner entity or organization. In contrast, a choreography formalizes the way business participants coordinate their interactions. The focus is not on orchestrations of the work performed within these participants, but rather on the external exchange of information (messages) among them.

Starting from the existing meta-model, we intend to extend the packages summarized above by introducing new modelling constructs that are required to support user-centrism, decentralization, compositionality and incrementality, adaptation, and QoS awareness of choreographies. The extension will then support a fully dynamic, scalable, and domain-expert-centric choreography development process.

2.2. CHOReOS Architectural Style for Choreographies in the Future Internet

The Future Internet is expected to offer to the developers of service-oriented software a plenitude of design choices, concerning the services that will be available, the ways to interact with these services and the ways to compose them. Coping with the complexity introduced by the availability of such a plenitude of design choices is a main challenge of CHOReOS. To deal with this challenge, the CHOReOS conceptual model shall specify from a conceptual and generic view the main entities involved in a service-oriented system of the Future Internet. On the other hand, the CHOReOS architectural style shall extend/refine the aforementioned generic view.

2.2.1. State of the art

Since the early 90s the software engineering community recognized the importance of investigating the high level organization of large scale software systems, a.k.a. *software architecture*, towards facilitating the systems' analysis, design, implementation, reuse, testing and evolution [GS96]. Specifically, a software architecture models a system in terms of high level *abstractions* such as *components*, *connectors* and *configurations*. Components represent the basic computational elements of the system, while connectors represent interaction protocols between these elements. Configurations are high level views of the system, consisting of components and connectors.

The particular research field that emerged as a result of the aforementioned focus of attention brought into light several valuable results, which include:

- *Software architecture description languages* [MT00] that provide means for the formal specification of components, connectors and configurations.
- *Software architectural styles* [GS93] that define families of software architectures (i.e., instances of a style) [GS93] through the specification of:
 - a vocabulary of components and connectors, and
 - constraints on the way that these elements can be combined to construct instances of the style.

Specification of components/connectors/configurations

Concerning the formal specification of components, connectors and configurations various approaches have been proposed. The interested reader may refer to [MT00,Zar00] for detailed surveys of the prominent features of early approaches; briefly, in these approaches, the components' interfaces are specified in various ways such as provided/required operations in ASTER, entry/exit points in Conic/Darwin, top/bottom ports in C2, players in Unicon, input/output ports in Wright, events in Rapide. The components behaviours are specified in terms of various formalisms such as temporal logic, CSP (Communicating Sequential Processes), CCS (Calculus of Communicating Systems) and partially ordered sets of events.

Connectors are specified in similar manners. However, not all of the early architecture description languages considered connectors are first-class modelling concepts (e.g., Conic, Darwin, Rapide) [BI03]. Concerning the modelling of connectors and interaction protocols in general, there have also been several interesting independent efforts (i.e., not in the context of specific architecture description languages). The most recent approaches include [BS08] that allows hierarchical specifications, [ACMM+09] that undertakes a stochastic approach, and [ACMM07] that focuses on QoS-related attributes.

Configurations (or topologies, structures) specify relative arrangements of component and connector instances. Therefore, configurations are typically described in terms of bindings between points of interaction. Several approaches either assume or provide means to describe stylistic constraints. The constraints may simply describe restrictions on the way components are bounded. Moreover, the constraints may be on the behaviour of the overall configuration (e.g., requirements for security and dependability, conditions for reconfiguration).

The main concepts introduced by the aforementioned approaches are nowadays included in UML 2.0, which is considered as the main industrial standard for specifying the structure and behaviour of software [MRRR02]. In particular, UML 2.0 provides means for specifying components and connectors. A component may provide and/or require a number of interfaces. An interface specification consists of a set of operations. Two different kinds of general purpose connectors are supported: (1) delegation connectors link the external interface of a component to the internal parts (i.e., other more primitive components, or

classes) that realize the component's behaviour; (2) assembly connectors link required interfaces to provided interfaces. The behaviour of components and connectors can be specified using various standard means such as state machines, interaction diagrams, use case diagrams, etc.

More recently, software architecture description is examined from a broader perspective that goes beyond the view that has been adopted by the foundation architecture description languages. Specifically, the recent approaches concentrate on modelling multiple stakeholders' concerns and design decisions [ISOSA,KLV06,TMD09,MMPT10]. To achieve this goal, the provided modelling means should be extensible. An interesting approach to this direction is ByADL (Build Your ADL) [DMMP+10], a framework proposed for developing next generation ADLs, with respect to particular domains and stakeholders' concerns. ByADL provides means enabling software architects to create their own ADL by customizing or extending existing ADLs. ByADL is fully integrated and built on top of DUALy [MMPT10,EMMP+10], a framework that enables interoperability among ADLs, as well as UML. In the same spirit, the Charmy framework [PIM09] assists the architects in the assessment of architectural choices, *via* iterative modelling, simulation and model checking.

Architectural styles

Regarding architectural styles, the prominent work of Garlan & Shaw [GS96] constitutes the foundation of this field. In particular, in [GS96] the authors documented a set of widely used architectural styles along with the pros and the cons that characterize them. The aforementioned fundamental architectural styles constitute the basis of Internet-related architectural styles that emerged in the 2000's [FT02,WSA]. In this context, two different architectural styles emerged, namely RESTful services [FT02] and WS* services [WSA].

The RESTful services style is derived from the fundamental client-server architectural style [GS96]. Specifically, according to the RESTful services style, *the notion of components* corresponds to services [PZL08]. A service provides access to a set of resources, through a unified interface that defines a fixed set of four operations, namely PUT, GET, POST and DELETE. The PUT operation is used for creating a new resource; the DELETE operation is used for deleting a resource; the GET operation allows reading the current state of a resource; finally, the POST operation enables changing the current state of a resource. A resource is identified by a corresponding URI and its state may be provided in a variety of formats such as HTML, XML, JPEG, PDF. In the RESTful services style, *the notion of connectors* corresponds to the interaction protocols that can be used to access the resources that are offered by RESTful services. In particular, 2 different types of connectors are provided HTTP-based and Waka [FT02,PZL08]. Finally, the notion of configurations in the RESTful services style corresponds to mashups [PZL08].

The WS* style is also derived from the fundamental client-server architectural style [GS96]. As in the case of the RESTful style, *the notion of components* corresponds to services. The WS* style does not assume a unified interface for all services [PZL08]. Specifically, a service provides access to a set of resources through a set of interfaces. An interface consists of a set of operations that can be invoked. The input of an operation invocation is a request message that is sent towards the service and the output is a response message that is returned back to the invoker, after the execution of the requested operation. An interface is typically specified using standard languages such as WSDL, SAWSDL or WSDL-S. The latter two languages allow specifying additional semantic information concerning the interface operations. The formats of the exchanged messages are specified in terms of XML schemas, which are further mapped into lower level messages that conform to the standard SOAP format.

The notion of connectors in the WS* style corresponds to the protocols that can be used to interact with WS* services. As opposed to the case of the RESTful style that assumes HTTP

or Waka, in the WS* style services can be accessed *via* a variety of protocols such as [PZL08] HTTP, SMTP, IIOP, TCP, JMS, MQ, BEEP.

Finally, *the notion of configuration* in the WS* style corresponds to service compositions that may come in two different flavours. In the simplest case, WS* services are configured in the form of an *orchestration*, which defines a process consisting of simple and complex activities. A simple activity may correspond to the reception of a request message that initiates the process, the sending of a reply message that comprises the results produced by the process, the invocation of a service operation, etc. A complex activity consists of constituent (complex or simple) activities and control/data flow dependencies among these activities. Typical complex activities involve the sequential, concurrent, conditional, iterative execution of their constituent activities. Several standard process description notations have been proposed to support the specification of service orchestrations. Among the prominent ones we have WS-BPEL which originates from WSFL and XLANG. In a sense, service orchestrations are similar to mashups. In the most advanced case, WS* software is configured in the form of a *choreography* that describes a collaboration amongst a set of constituent services, which does not involve a single point of control. A choreography can be seen as *a composition of independently developed orchestrations*, specifying how a constituent service collaborates with other constituent services. Typical standard notations that allow specifying service choreographies are WSCDL, WSCI and more recently BPMN2.

Further Internet-related styles that emerged recently are cloud computing [Hay08, Vou08, BYVB+09, AFGJ+10] and the Internet of Things [WW10]. The cloud computing style is layered. In the first layer the main components are servers that are especially configured towards the provisioning of cloud services. The layer on top of the servers layer consists of infrastructure services that can be exploited by customers so as to purchase resources offered by the servers layer. The layer that lays over the infrastructure services consists of platform services that facilitate the deployment of customer applications. Finally, the customer applications consist of application-related services. The cloud services at all layers may conform to the RESTful or to the WS* architectural styles. A typical example is the case of the Amazon cloud services¹ that come in both flavours. On the other hand, in the Internet of Things, the main components are entities with limited resources and computing capabilities that are identified by URIs. These elements may provide lightweight services that can be accessed through connectors, which correspond to a variety of wireless communication protocols. In this style, configurations may scale up to a very large number of things and are mainly *ad hoc*.

2.2.2. Baseline

As already mentioned in Section 2.1, the CHOReOS architectural style shall refine the main concepts of the CHOReOS conceptual model, which in turn will be developed upon the NEXOF reference architecture [NEXOFRM] that provides a unified view of the state of the art service-oriented architectural styles. As in our conceptual model, UML will be employed to model the main entities of the CHOReOS style.

The NEXOF reference architecture distinguishes a set of concerns that should be covered by platforms that support the development of service-oriented software:

- the service concern relates to the creation of services;
- the message concern relates to the messages and the interaction protocols between services;
- the discovery concern focuses on the registration and the retrieval of services;
- the composition concern addresses the construction of software processes;

¹ <http://aws.amazon.com/>

- the presentation concern considers the interaction between the users of a service-oriented development platform and the platform itself;
- the management concern relates to the deployment, the monitoring and the adaptation issues;
- the security concern addresses security issues in service access;
- finally, the resource concern considers the necessary resources for the execution of services.

The CHOReOS platform particularly focuses on the development of service-oriented software for the Future Internet. Consequently, the CHOReOS style must consider the aforementioned NEXOF concerns and further extend/adapt them to the specificities of the Future Internet and particularly, the constantly growing amount of alternative design options that become available all over the Web.

2.2.3. Progress

Up to now, a large amount of RESTful and/or WS* services can be discovered all over the Web (see ServiceFinder², WebServiceList³, RemoteMethods⁴, WSCE⁵, ProgrammableWeb⁶). Several services provide similar functionalities through different interfaces and/or interaction protocols. In ServiceFinder, for instance, there are already more than 25,000 registered services. Amongst these services, more than 1800 are for communication (900 email services, 850 Fax services, 40 SMS services and 40 instant messaging services). Moreover, there are more than 8000 services for shopping, more than 1700 services for scientific calculations, etc.

In the Future Internet, this diversity of design choices is expected to dramatically grow. This expectation becomes evident by looking at the growth of the amount of available services in the past few years. In the WSCE, for instance, the annual growth of the number of available services is more than 130%. Given this situation, the good news is that the developers will be able to exploit this plentitude of alternative options, towards developing the Future Internet applications. However, the bad news is that the complexity of exploring and taking advantage of such a plentitude of alternative design options shall grow with the cardinality of these options. In principle, dealing with the complexity introduced by the availability of a large amount of available design options amounts to defining corresponding abstractions that represent the alternative design options, whilst hiding the specificities of each one of them. Defining such abstractions is a main objective in CHOReOS.

In particular, the CHOReOS style shall go beyond the state of the art and baseline on Internet-related architectural styles, by providing novel abstractions (i.e., components, connectors, configurations) to facilitate the exploitation of the large variety of available design options that emerge in the Future Internet, towards the development of highly scalable choreographies. Doing so, involves revisiting and extending the NEXOF service concern. From a functional perspective, the CHOReOS abstractions shall allow grouping services that provide similar functionalities through different interfaces and/or interaction protocols. From a non-functional perspective, the CHOReOS abstractions shall enable grouping services that provide similar non-functional properties such as performance, availability, reputation, and cost. In a sense, the CHOReOS abstractions will play the role of abstract types, while the represented services will play the role of concrete subtypes.

² <http://www.service-finder.eu/>

³ <http://www.webservicelist.com/>

⁴ <http://www.remotemethods.com/>

⁵ <http://www.uoguelph.ca/~qmahmoud/WSCE/index.html>

⁶ <http://www.programmableweb.com>

Consequently, both the CHOReOS abstractions and the represented concrete services must adhere to some notion of behavioural subtyping [LW94].

The CHOReOS abstractions shall be the core concepts that will facilitate the organization of available services in an abstraction-based hierarchical structure, which will enable in turn the efficient retrieval, composition and adaptation of large scale Future Internet applications. Consequently, the NEXOF discovery concern shall be extended with respect to this idea. Moreover, the NEXOF composition concern will be revisited and enhanced, so as to consider means for the specification of abstract configurations that represent choreography-based computations, and the gradual refinement into concrete choreographies. Further support shall be provided towards the evaluation and prediction of the choreographies non-functional properties, through analytic and simulation techniques that will be tailored to the CHOReOS architectural style.

The CHOReOS architectural style will also revisit the NEXOF presentation concern towards dealing with the interaction between the CHOReOS users and CHOReOS itself. Regarding the management concern, the CHOReOS style will provide extensions that reflect the monitoring, V&V, and adaptation of large scale Future Internet applications. Finally, the NEXOF resource concern shall be enhanced to account for execution conditions, derived from diverse styles such as cloud computing and the Internet of Things.

3. WP2: Dynamic Development of Adaptable, QoS-aware ULS Choreographies

The Future Internet demands a dynamic and user-centric development process from design-to run-time, spanning all the software life cycle. WP2 is about defining such a process and devising its related methods and tools to enable the dynamic development of adaptable, QoS-aware, highly scalable choreographies. WP2 then studies:

- A dynamic development process for adaptable QoS-aware highly-scalable choreographies (Section 3.1);
- Methods and mechanisms for large scale service base management (Section 3.2);
- A domain-expert specification framework for adaptable QoS-aware highly-scalable choreographies (Section 3.3);
- Adaptable choreography synthesis (Section 3.4);
- Analysis of choreography scalability (Section 3.5).

3.1. Dynamic Development Process & Tools

The decentralization and pervasiveness of adaptable, QoS-aware highly-scalable choreographies demand new service composition and coordination paradigms that span the entire choreography life-cycle including all the phases of the choreography development process, from development to deployment and execution. In particular, CHOReOS choreographies must be reliable and meet user's requirements and needs. Additionally, they should be adaptive to context changes concerning both user-centric and resource-centric data, and QoS. New methodologies are needed to address the specification, adaptation, synthesis, scalability, validation, simulation, dynamic execution, and re-configuration aspects of a CHOReOS choreography. The whole choreography development process is affected by the high dynamicity and evolution degree of a highly-scalable choreography. As described in the CHOReOS DoW, certain phases of the development process, like analysis and validation, together with their proper artefacts, need to become pervasive, accompanying the choreography implementation at run-time.

3.1.1. State of the art

Broadly speaking, a “standard” development process focuses on activities that are traditionally divided into development-, deployment-, and run-time activities [Som10]. The evolutionary nature of choreographies in the Future Internet makes unfeasible a standard development process since, e.g., dealing with adaptation and QoS-awareness would require predicting functional and non-functional properties of the choreography (with respect to virtually any possible change) before its execution. Whenever a change occurs, if choreography evolution has to be supported by means of adaptation, all the artefacts and models might be exploited also by the deployment- and run-time activities hence leading to a “non-standard” development process view [Inv07], i.e., a dynamic development process model. Therefore, central to the development process is the notion of models built upon concepts precisely collected in the conceptual model (see Section 2.1). Models are abstract views of systems, suitable for reasoning, developing, and validating a real system. Models can be functional and non-functional and can represent different level of abstractions of the real system, from requirements to code. Model Driven Development (MDD) [Sel03,CD07,DP05] and, in particular, Model to Model (M2M) transformations enable the shift among various levels of abstractions and across several domains, possibly in an automatic way. The models@runtime approach [BBF09,BBFJ09] seeks to extend the applicability of models produced in MDD approaches to the run-time environment, hence promoting ever more the realization of the dynamic development process view.

An example of a dynamic development process model has been proposed within the FP6 IST EU PLASTIC project [PlasticWS, Plastic08]. The PLASTIC development process [ABCD+07,PlasticD2.2], which instantiates the proposed process model [PlasticD2.1], relies on model-based solutions to build adaptable context-aware service-oriented applications. It encompasses methodologies and software tools to generate QoS models and adaptable application code from UML-based specifications [PlasticD2.3]. All these methodologies and tools are supported by an integrated framework, which is based on a UML profile of the PLASTIC conceptual model [PlasticD1.2]. That is, in PLASTIC, all the development process activities originate from the PLASTIC conceptual model and exploit as much as possible M2M and Model to Text (M2T) transformations. A service model that is conforming to the conceptual model is specified. Out of this model, M2M transformation is performed in order to derive models for different kinds of analysis. M2T transformation is used to generate the adaptive code (with respect to the context and user needs) of the service. The PLASTIC development process represents a specific runtime approach where models are used at run-time in order to both drive adaptation and monitor the service execution so as to provide feedbacks to the analysis, hence refining the service model according to possible changes in the environment.

Ultra Large Scale (ULS) factors such as decentralization, data heterogeneity, and uncertainty represent challenges that can be addressed by adopting a models@runtime approach [Che09]. Modelling techniques coupled with MDD capabilities such as model transformation and code generation provide viable means to enable system monitoring, model analysis and adaptation at run-time [ZC06,GC08]. In [Mao09], model-based execution traces are proposed as a specific models@runtime artifact. Design-level models (e.g., sequence diagrams and statecharts) are made available at run-time through MDD technologies so that the program execution generates high-level execution traces that are based on the events extracted from the design-level models, hence providing support to monitoring and run-time adaptation. In [CGFP09], design variability models are reused at run-time to support self-reconfiguration of systems when triggered by changes in the environment. In [MBJF+09], run-time models of a system are used to reduce the number of configuration and reconfigurations that has to be considered when planning adaptations of the application. Dealing with ULS, analogous optimizations become a primary concern. In [GHT09], the use of architectural models as run-time artefacts to enable adaptation is proposed. The use of configuration graphs is investigated as a means for monitoring and recording information about the system adaptations. In [GIWO09], an advanced run-time model has been defined. It is based on the novel concept of H-graph and the study carried on several projects has shown that H-graphs can be more understandable from a user's perspective than more traditional architectural models. We recall that user-centrality is another key ULS factor. As discussed in [LBTA10], meta-models allowing the definition of models where design- and run-time concepts are combined are another key aspect for the creation and exploitation of effective run-time models. A further important aspect concerns what are the requirements of the run-time models in order to enable the exploitation of existing MDD technologies, such as M2M and M2T transformations. The work described in [SHCS10] shows promising preliminary results in this direction.

3.1.2. Baseline

While devising the CHOReOS dynamic development process and its related tools, our work will be inspired by the PLASTIC development process model and will be based on the models@runtime approach. 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 design- to run-time, as in the models@runtime approach.

Our choice of being inspired by PLASTIC is based on three considerations: (i) as the whole choreography model specification will be supported by the CHOReOS IDRE (Integrated Development and Runtime Environment), few errors can be introduced in the functional and non-functional specification of a choreography; (ii) QoS embedded within a choreography model better supports the M2M transformations towards analysis models and, on the way back, better supports the feedback of the analysis; and (iii) in the path to code generation, for a choreography, being QoS-aware suitably drives the choreography adaptation strategies. However, the model transformation techniques exploited in PLASTIC are standard and, hence, no support to their automated verification is provided. Note that this aspect is critical when specifying M2M transformations between models defined at different levels of abstraction. Thus, CHOReOS represents a challenging domain that requires both extending the formalisms, techniques, and tools of the PLASTIC development process and, due to the peculiarities of the Future Internet, adding new ones, as discussed in Section 3.1.3.

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 [BBF09], 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 system's requirements and goals during execution. To this aim, the domain-expert specification framework proposed by CHOReOS (see Section 3.3) and its model-transformation-based exploitation during the whole development process life-cycle concern issues whose solution will represent a step forward with respect to the models@run-time approaches discussed in the SoTA.

3.1.3. Progress

As a suitable adaptation of the models@runtime approaches discussed in the SoTA, and as a novel instantiation of the PLASTIC development process model, the dynamic CHOReOS development process poses new challenges whose solution represents a significant progress with respect to the discussed SoTA and baseline.

In the following sections (from Section 3.2 to Section 3.5), we discuss progresses specific to the main artefacts and activities of the CHOReOS dynamic development process. Here, we discuss “process-level” progress concerning the transformational approaches that we use to integrate the various artefacts and activities, and the progress that we make with respect to the MDD and transformation techniques in the SoTA.

As discussed in the CHOReOS DoW, differently from what is traditionally done, it should be possible for domain experts, even when they do not have a technical background, to specify the desired choreography with respect to their domain (business) goals and expectations. For this reason, within the CHOReOS development process, a novel key artefact for modelling is represented by the domain-expert specification framework based on a suitable extension of the i^* notation (see Section 3.3). The choreography domain-expert model results in a very high-level specification that although, on the one side, allows domain experts to effectively specify the choreography, on the other side, makes it hard to drive automated reasoning directly from it. To deal with this issue, M2M transformation techniques are developed to translate the domain-expert specification into a more technical form of choreography specification, an extended BPMN2 model [BPMNv2.0]. Due to the unavoidable gap between the abstraction levels of the two models, specifying these transformations often results in an error-prone process. Thus, novel techniques for supporting the automated verification of these transformations represent a main progress with respect to the standard MDD approach used in the SoTA. As a further novelty, these techniques should be general enough to be applied also to the M2M transformations required for deriving the choreography models suitable for synthesis purposes (see Section 3.4), and for analysis and scalability prediction purposes (see Section 3.5). Another progress is represented by the definition of a “cross-cutting” transformational approach, which serves

to achieve interoperability among different technique-specific models. Differently from what is done in the SoTA, this integrates the different modelling and reasoning technologies, hence bridging the gap between the various models that have to be used for choreography analysis, validation, synthesis, and implementation purposes.

3.2. Large Scale Service Base Management

Service registry and repository are core concepts in service-oriented architectural styles that facilitate the registration of provided services and the retrieval of required services. Nevertheless, these concepts should be revisited in the context of the Future Internet where the rich plentitude of available services is expected to pose additional scalability, efficiency, precision, and recall requirements concerning the main functionalities offered by service registries and repositories. Consequently, a main goal in CHOReOS is to deal with these requirements by providing a large-scale service base management infrastructure.

3.2.1. State of the art

Alexander Mintchev in his ICWS 2008 paper [Min08] repeats an interesting definition of service registries, originally coming from a Gartner report: “A *service registry* is defined as ‘a specific type of repository that allows companies to catalog and reference the resources required to support the deployment and execution of services’. While a registry is a storage containing references to resources, a repository is storage, containing the resources themselves“. Both the research and the industrial community have mainly worked on registries; thus, in the following discussion we focus on registries.

We organize the description of the state of the art according to the following issues:

- 1) The content (or data model) of a service registry, and
- 2) The degree of decentralization, including centralized repositories, decentralized (practically P2P) and hybrid (including super-peers in the architecture). The discussion on centralized repositories also covers issues of internal organization for registries, as well as the problem of discovering services that are relevant to a user’s query.
- 3) Finally, we discuss Web service search engines that constitute an alternative approach to service registries.

For the interested reader who would like to probe further, there are a couple of excellent surveys worth mentioning:

- Dustdar and Treiber in their DAPD 2005 paper [DT05] try to pose criteria for registry evaluation and organize the discussion from two viewpoints, the viewpoint of the system and the viewpoint of humans. Therefore, they present an insightful description of QoS aspects around a Web service registry. Specifically, the main requirements (and thus, selection criteria) for Web service registries are: Interoperability, Reliability, Integration with the Web, Security, Fault tolerance, Scalability, Availability, Expressiveness of query language, Expressiveness of Web service description, Extensibility of data model, Transient Web services, and Management and provisioning facilities. Then, the authors move on to discuss different architectures and data models for Web service registries. We will follow this organization in our discussion, too.
- A more recent survey, by Rambold *et al.* published at SCC 2009 [RKL09], provides a more fresh and detailed discussion of the activity in the later years around the structuring of Web service registries. The main emphasis is given to two dimensions and, specifically: (a) the degree of distribution (which we also follow in the sequel), and (b) the degree that a matchmaking process in the registry employs semantic information for Web services.

Finally, we refer the reader to Mintchev's report [Min08] on problems concerning the interoperability of UDDI registries.

Registry's content (also known as data model)

Typically, most of the works adopt the UDDI model to structure the contents of the registry. The Universal Description Discovery and Integration (UDDI) specification [UDDI] provides a registry that allows the advertisement and retrieval of Web service descriptions. A UDDI registry is organized as a collection of: (a) white pages where business entities advertise the offering of business services, (b) yellow pages where this advertisement is based upon a registry-based taxonomy, and (c) green pages, where pointers to Web service descriptions are also provided. To allow the description of the technical characteristics of Web services, UDDI proposes the usage of tModel's ("technical models"). tModels are placeholders for the technical description of Web services. The standard does not specify neither a language, nor a structure, and, naturally, no semantics for this description; nevertheless, there is a separate set of guidelines, for publishing WSDL descriptions in UDDI (but not as part of the standard). A set of APIs for the advertisement of services, the inquiry of the registry on the basis of user-specified criteria and miscellaneous other operations (such as registry synchronization) is also provided. Summarizing, we can say that at this time, UDDI is a human-oriented standard, with the aim: (a) to help knowledge workers decide on possible collaborations at a business level, and (b) to help developers find out the specifications over which to base the development of client applications that exploit advertised services. However, the latter is performed *via* references to WSDL descriptions of the published services, which are practically out of the scope of the UDDI standard (and the developers, or developer-supporting tools are supposed to jump from the tModel out of the UDDI specification and to the WSDL document of a service in order to get the technical details).

The UDDI standard specifies an API that can be used by front-end tools to browse or pose queries to the registry's contents. The API is mainly a keyword (or, value) lookup API with calls of the form: `find_binding()`, `find_business()`, `find_tModel()`, etc. The entire process is mainly aimed at enabling business users find related business entities, rather than technical ones. Unfortunately, this issue also involves the linking to WSDL documents. Fundamentally, the linkage between WSDL and UDDI is performed according to UDDI Best Practice Report that practically instructs a WSDL port Type to be mapped to a homonymous UDDI tModel. When a query is performed, the tModel overview URL contains the link to the homonymous WSDL port type and its name can possibly contain the name of the WSDL service - and this is all that the registry reports to the user.

Other data models include ebXML and WSIL but they have less impact in the state of the art - in any case, see Dustdar & Treiber [DT05] for a discussion.

Layers of decentralization & structure

Three main categories of registry structure are identified in the literature:

- 1) *The centralized approach* stores all the registry information in a single repository. This provides consistency (single version of the truth) and fast local retrieval under normal circumstances.

As expected, the main issues in a centralized setting mainly involve how the retrieval of service descriptions is conducted. To facilitate the discovery process, the structuring of these service descriptions within the registry is also important. From the early days of service bases, the research community has worked towards enhancing the simple contents of a UDDI registry by semantically rich metadata [PKPS02] [SVSM03] [LMKK+06] based on DAML-S and OWL-S languages. Since this annotation requires extra data entry effort, later efforts also pursued the path of clustering the contents of the registry in order to "extract" the semantically rich information in an automated way [NL07] [LL08] [EHM10] [LSYW+10]. As an add-on to these efforts, more elaborate organizations of the registry contents have also been explored in order to improve the recall and response time of the registry [LL07].

- 2) *The decentralized approach* allows peers, participating in a network of cooperating sites to store locally their own service registry. Then, service retrieval is facilitated by distributed querying services that span all these local registries to compile and present answers to users. Naturally, the distributed setting provides a richer wealth of answers, significant chances of better scalability in terms of both the user load and the available data to index, without the risk of a single point of failure that the centralized solution suffers. This comes at the cost of supporting a framework that allows each peer to know which fellow peers to contact for servicing a user request along with the necessary communication overhead whenever a query to the decentralized "virtual" registry is posed (e.g., see [SVS04]).
- 3) *Hybrid architectures* involve a more elaborate structure with super-peers that act as yellow pages and/or heads of sub-structures within the P2P network. Supper peers facilitate the efficiency and effectiveness in the servicing of user requests (i.e., by achieving a reduction to the messaging overhead in the P2P network and better response time, without sacrificing the wealth of the answer) (e.g., see [LZSK+05] [DHL05] [KEMS07]).

Alternatives approaches to service registries

Is UDDI the reference solution for service bases? Is this the only way to discover services? We would like to clarify here that all the works we have reviewed so far, assume that service descriptions are published *via* a UDDI registry. However, despite the fact that UDDI is the unquestioned choice as a registry standard, there is a quite significant discussion on whether a registry *per se* is the best possible way to follow in order to discover service descriptions. The main reason for criticism is empirical, coming from observations that publicly available UDDI registries contain incomplete or out-of-date descriptions (mainly in terms of the WSDL description of the advertised services). To this end, there are several suggestions to implement the discovery process *via* (either specific- or general- purpose) search engines that crawl the Web and index service descriptions that they locate.

Most notably, Al-Masri and Mamoud, in their WWW 2008 paper [AM08], have performed a crawling experiment in which they constructed a crawler that collected approximately 5000 service descriptions for the Internet (specifically, 1405 from registries and 3672 from search engines). Along with the crawler, the system involved the verification of the existence of the WSDL file, the validation of the described service interface and the eventual analysis of the retrieved specification. The processing of the retrieved service descriptions revealed that only 47% of the registries would perform as expected as opposed to 91% of services retrieved *via* search engines. The authors attribute this to staleness, as search engines continuously update the information they index, whereas registries do not. Also, in [SCMK07] the authors experiment with different ways of advertising Web service descriptions for retrieval *via* popular, general-purpose, search engines. The results show that the best possible way is to publicize the WSDL description of the services by embedding them into Web pages.

3.2.2. Baseline

Since CHOReOS will overall rely on the NEXOF reference architecture [NEXOFRA10], the NEXOF service discovery concern shall serve as a main baseline for the CHOReOS service base. Concerning the distribution architecture, there is a large variety of architectures and platforms from which one might choose. Naturally, most of the registry approaches consider scalability and efficiency. Nevertheless, the constantly growing amount of services in the Future Internet calls for further investigation towards achieving these qualities. A significant issue that has been overlooked in the design of the existing approaches, and presents an impediment to the scalable and efficient retrieval of services, is the fact that the precision of the results of a query is usually very poor [SS10]. In other words, it is generally hard to find the services that can actually fulfil given functional and non-functional requirements.

Consequently, since precision is inherently poor, service registries should provide means for service retrieval that maximize the recall. Maximizing recall implies that the whole search space of service registries should be explored towards answering a query. This search space is continuously scaling up and in the context of the Future Internet, its size is expected to dramatically affect the performance of service retrieval.

So far, only two approaches consider this issue. These approaches shall constitute the baseline for the organization of the CHOReOS service base and its accompanying service registration and querying engines. Specifically, in [ZB09] the authors propose an approach for the efficient retrieval of all possible interesting service compositions from an ontology-based registry. The proposed approach aims at reducing the overall search space *via* the definition of a given search context. The search context consists of a number of Web service domains of interest and locale attributes. After pruning the search space, the proposed approach employs rather typical service matching facilities. UOI and INRIA also worked towards an approach for efficient service retrieval that maximizes recall [AZI09]. In particular, the proposed approach organizes existing services in groups of equivalent services based on a corresponding equivalence relation and theorem. The service groups are characterized by functional abstractions, called profiles. Moreover, groups of services that provide: (a) fewer and/or (b) more generic functionalities are associated with groups of services that offer: (a) more and/or (b) more specific functionalities, *via* a corresponding specialization relation and theorem. Then, the complexity of retrieving services that match given functional requirements scales up with the number of available groups, instead of scaling up with the number of available services.

3.2.3. Progress

The previously mentioned baseline approaches and, in particular, the work proposed by UOI and INRIA [AZI09] suggest that the abstraction-based organization of service registry is a promising direction towards coping with the Future Internet scale. CHOReOS plans to evolve this baseline approach towards devising a hierarchically structured service base organization of service abstractions. Moreover, we plan to consider both functional and non-functional abstractions leading to different views of the vast amount of available services in the Future Internet. Specifically, our contribution concerning the CHOReOS service base consists of:

- Methods and tools for the recovery of functional and non-functional abstractions out of the vast amount of available Future Internet services.
- A distributed service base that reflects different functional and non-functional hierarchical views of available Future Internet services.
- A service registration engine that allows for enriching the contents of the service base.
- A service query engine that allows for exploiting the contents of the service base; the query engine in particular should provide means for the retrieval of services that facilitate the work of users whose experience and roles may be quite diverse (e.g., domain experts, designers, developers).

Concerning the recovery of service abstractions and the corresponding organization of service descriptions in the CHOReOS service base, our approach shall rely on the observation that the operation structure is the key for the clustering of similar services. So, we will cluster services by their operation signatures and exploit this clustering later, at retrieval-time, when the developer is looking for similar services that meet given requirements. Our method will use an agglomerative clustering approach as a starting point (unlike the method of Nayak & Lee [NL07], we shall exploit the similarity of operation structures instead of text terms); we anticipate a limited scalability potential by agglomerative algorithms, so we will also explore the possibilities offered by alternative hierarchical clustering schemes. Similarly to the above service clustering, we will also cluster service with respect to their non-functional properties, such as availability, price, reputation, etc. We will

start our investigation using a similar approach to the functional characteristics, mainly based on our preliminary findings concerning the handling of scale. We will leave ontological aspects out of our investigations, mainly because there is too little benefit from them: as currently understood, ontologically-aware approaches more or less map WSDL to a practically similar structure in OWL-S, without being able to extract an ontology out of the corpus of service descriptions. We follow a syntactic-based approach and we will pursue the construction of such an organization, in the form of clusters that will correspond (approximately, of course) to an intuitive user-level organization of services.

3.3. Domain Expert Requirements Specification and Choreography Modelling

Requirements specification is a fundamental activity in software development. The distinctive feature of the Future Internet that primarily affects requirements specification is the active role of domain-expert users who tend to take the place of requirements analysts. Therefore, providing domain-expert centric requirements specification methods and tools that guide the synthesis of choreographies is amongst the main challenges of the CHOReOS dynamic development process.

3.3.1. State of the art

There has been little crossover between research in requirements engineering and in service-oriented computing, and limited investigation of requirements-based service discovery and composition in the service-centric system communities. 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. FF95, WRGM08]. 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 quality-of-service ontologies and measurement (e.g., [SHWS05]), however little research or practice traces service qualities back to the originating quality requirements on the systems. One serious problem to overcome is the granularity mismatch between services and systems [ZM08]. As a consequence, most QoS research focuses more on service-level qualities and SLA, but not the originating requirements and their expression.

In [RS04], the authors propose a general framework for automatic Web service composition, where one of the phases involves generating the composition process based on service consumer requirements. Users may express their requirements in the form of a service specification, effectively describing the one service that, if available, there would be no need to compose an equivalent from atomic services. Waldinger [Wal00] elaborates an idea for service synthesis by theorem proving. The approach is based on automated deduction and program where available services and user requirements are described in a first-order language, related to classical logic.

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. Natural language can be decoded both by the end-user and by the analyst, and can assure the communication between them. The current literature suggests that an overwhelming majority of requirements specifications are written in natural language, often enhanced by information in other notations, such as formulae and diagrams [BKK03]. The FP6 SeCSE Integrated Project reported an iterative and incremental requirements process with software tools support for service-centric systems [SeCSE]. Analysts form queries from a requirements specification to discover services that are related to the requirements in some form. Descriptions of these discovered services are retrieved and explained to stakeholders,

then used to refine and complete the requirements specification to enable more accurate service discovery, and so on.

In addition, outside of structured English, few methods for specifying QoS requirements have been established. A notable exception is Planguage [Gil05] – a keyword-driven language that allows measurable, testable QoS requirements to be written. Planguage has many benefits – it is flexible, compact, extensible, and prevents omissions by providing a consistent set of parameters for QoS requirements. Also, Zachos *et al.* [ZDS08] present techniques to match semi-structured natural language QoS requirements to service qualities during service selection using ontologies.

Within the last couple of years, the research community identified a number of different ways to define composition approaches taking QoS requirements into account. For instance, Brandic *et al.* [BPB08] describe Amadeus, a QoS-aware Grid workflow system that supports a comprehensive set of QoS requirements where each workflow element may be specified with different properties that indicate the user's QoS requirements. Moreover, current semantic Web service composition languages such as BPEL4SWS [NLKL07] and OWL-S [MBHL+04] use ontologies in order to reach a higher degree of automation in service discovery and service composition. DAML-QoS [ZCL04] and QoSOnt [DLS05] are just two examples of ontologies for QoS in service-centric systems that provide openly available, extensible OWL-S ontologies, allowing complex and varying QoS metrics to be defined. In contrast to DAML-QoS, QoSOnt does not just provide a schema for QoS in Web services, but uses the power of knowledge representation in OWL-S to allow a certain degree of intelligence to be applied by agents and applications (e.g. conversion of units, inference of composite metric values, and inference of the QoS of composite services). Future research challenges will have to focus on semi-automated techniques that allow users to specify what a composition should do - in a declarative manner - and what QoS guarantees are required and should be delivered at runtime. CHOReOS will support the decomposition and refinement of these requirements to match to choreography strategies that will increase the likelihood that the run-time service-based application will meet the requirements.

Another way to specify what a system should do is clarify how it should react to some situations written up by the system designer or system architect. One way to facilitate this is to design a specific high-level language to express requirements that is later on refined into a language more closely to the business process. This is what the work in [KPR04] does: the graphical modelling language Tropos makes it possible to express the requirements a business process has to meet, and to perform model checking directly on the business requirement level. Tropos is based on i^* [YM94] – an established process and graphical modelling technique that enables organizations to model socio-technical systems as a network of dependent and interacting heterogeneous actors. It supports the representation of actor goals and how the attainment and achievement of these goals contribute to the satisfaction and attainment of other goals. As such, i^* is suitable as a graphical technique for enabling end-users to express desired quality trade-offs in the behaviour of the service-based application.

3.3.2. Baseline

Requirements driven automated service composition is still an open research problem, since none of the existing approaches have been universally accepted as a standard method for automatically composing Web services. In the context of the Future Internet, the issue becomes even more challenging given that the goal is to allow domain-experts, instead of requirements analysts, to guide the systematic synthesis of ultra-large scale choreographies. First and foremost, steps should be taken towards a standard way of describing domain-expert goals and requirements. This description should be able to capture functional requirements as well as non-functional properties, including QoS aspects. At the same time, it should be easy for domain-experts to express their requirements using such descriptions.

Finally, conventional methods for requirements elicitation, analysis, and specification should be adapted to the dynamic nature of the Future Internet. To achieve these goals in CHOREOS, we will integrate research on qualities of service-oriented systems from projects, including SeCSE, with agent-oriented goal modeling techniques, such as the *i** modeling technique and model propagation algorithms.

Results from SeCSE that will be built upon include a natural language-based Expansion and Disambiguation Discovery Engine (EDDiE) and a software module called AnTiQue (Analogy Tracker in Queries) that retrieves Web services in domains analogical to a problem [ZMJ07, ZM08]. EDDiE discovers Web services from requirements expressed using structured natural language by extending information retrieval techniques to overcome incompleteness and ambiguity. The WordNet online lexicon is used to add different terms with similar meanings to the query in order to increase the number of Web services retrieved from the registries. AnTiQue retrieves designs and implementations of Web services that service providers designed for domains that are analogical to the current requirement problem. It matches incomplete and ambiguous natural language descriptions of requirements and Web services from different parties using different lexical terms, and computes complex analogical matches between descriptions without a priori classification of the described domains.

*i** and its underlying GRL (Goal Representation Language) will provide the scheme for representing actors' – domain experts' – requirements and goals. More established methods and notations such as the UML have no explicit representative of requirements and goals, so *i** was selected from available goal modeling techniques. *i** was selected because it represents a system as a network of heterogeneous actors – humans and software services – that depend upon each other to achieve goals. Previously the technique had been used to model component-based software systems and assist reasoning about selecting combinations of software components that best achieve a small set of goals of a system. In CHOREOS, we use *i** to describe actor goals independent of the behavior and qualities of systems and services implemented to achieve these goals, to maximize the space of opportunities in which we can implement different service choreography strategies. We do not use it to provide formalism for describing and reasoning about services or their choreographies.

3.3.3. Progress

As part of the work carried out within WP2, CHOREOS will specifically extend results of SeCSE with software tools to support the discovery of services with which to inform requirements expression during development-time and run-time activities, and will enable domain-expert end-users, rather than requirements analysts to express functional and quality requirements on services and service-based applications.

We will also develop a new quality ontology for expressing quality requirements on service-based applications and QoS of services aggregated in these applications. One distinguishing characteristic of this ontology will be the use of natural language expressions of different qualities designed to make the ontology usable to end-users. To develop this ontology, we will integrate quality of service ontologies, such as QoSOnt with requirements expression techniques, such as Planguage, and tailor the experimental ontology empirically to selected domains.

In CHOREOS we will exploit semantic matching mechanisms rather than formal reasoning techniques with *i**. We will extend *i** with a broad Quality-of-Service ontology that will enable CHOREOS to match and compare domain expert quality requirements, service qualities and qualities associated with different choreography strategies. We will extend the Quality-of-Service ontology with simple mechanisms to aggregate qualities during this matching process based on the concept of satisfaction arguments developed in requirements engineering research [Jac95]. We will also extend requirements-service matching techniques

from the FP6 SeCSE Integrated Project to match domain expert functional requirements with software services and choreography strategies. These strategies act as patterns that contextualize more formal reasoning techniques used in other tasks.

One advantage of service-oriented architectures is that new designs can be deployed and tested quickly to generate data with which to refine predictions. To reconcile the static view produced at development-time with the dynamic view that is created at run-time, we will take design-time techniques for software systems engineering – e.g., satisfaction arguments [Jac95] – and develop template argument types associated with different choreography run-time patterns. The main innovation here is capturing data to support the validity of domain and system assumptions that need to be true for services of different qualities to lead to satisfaction of quality requirements expressed by domain-expert end-users.

3.4. Adaptable Choreography Synthesis

Current approaches for synthesizing service choreographies, involve having a central unit responsible for selecting and composing the system. The centralization is a clear drawback when considering the expected Ultra Large Scale of the Future Internet. Dealing with the synthesis phase of decentralized coordination is among the main objectives of the CHOReOS dynamic development process.

The adaptation of synthesized choreographies is another challenge in ULS Future Internet and thus in CHOReOS (see Section 4.5). Recent efforts in the field range from dynamically substituting services to dynamically re-synthesizing parts of the choreography. Service substitution can be based either on wrappers (adapters) or higher level abstraction mechanisms. On the other hand, the re-synthesis of choreographies is related to the conventional dynamic reconfiguration problem.

Given the multidimensional and ever-changing context of the ULS Future Internet, an adaptable choreography implementation faces several challenges, like an ultra large service base, highly scalable choreographies, wide distribution and high heterogeneity in terms of domain policies and resources.

3.4.1. State of the art

Pioneer approaches concerning coordinator-based synthesis first appeared in the 1990s in the control theory domain [RW89]. The purpose of these approaches was to automatically synthesize a controller aiming at restricting the behavior of the system, satisfying the given specification. Thereafter, these approaches have been revised to fit the domain of embedded software systems [AMP94, CDFL+05, BCDF+07, BGLB+04, KS05]. As a common limitation, these approaches focus on mismatch prevention rather than on their resolution. The idea of embedding the interaction protocol into the components by means of adapters has been introduced by Yellin & Strom [YS97] to solve incompatibilities between component interfaces. However, this approach is not automatic.

In [AH01, PD02, CPS06], Labeled Transition Systems (LTSs) are used to model the I/O behavior of the components and automatically synthesize a set of constraints on the components' environment that allows interaction while enforcing failure avoidance (e.g., avoidance of deadlocks). A limitation of this approach is that the adapter code has to be manually written, exploiting the synthesized constraints. Later approaches from CHOReOS partners [IT03, IT04, AIN07, TII08] show how to automatically derive the actual implementation of the adapter from a specification of the interaction among components and of the requirements that the composed system must fulfill. The adapter allows both prevention and resolution of incompatible interactions. However, these approaches do not take into account non-functional properties. In this context, CHOReOS partners proposed, in cooperation with others, an approach to the automatic adapter synthesis for real-time

components [TFGG07]. Although this approach deals with both functional and non-functional properties, it is limited as it synthesizes only a model of the adapter.

Concerning coordinator synthesis in the W3C point of view of the SOA style, the key element that corresponds to a coordinator is the service choreographer. Many approaches have been proposed in the literature aiming at automatically composing services by means of BPEL-, WSCI-, or the latest W3C choreography pattern candidate, WS-CDL -choreographers [BP06, CDLM+08, MPT08, MPM08, MGI07, PBLH08, Sal08, SBFZ07]. The common idea underlying these approaches is to assume a high-level specification of the requirements that the choreography has to fulfill and a behavioral specification of the services participating in the choreography. From these two assumptions, by applying data- and control-flow analysis, the BPEL-, WSCI- or WSCDL- description of a centralized choreographer specification is automatically derived. This description is derived in order to satisfy the specified choreography requirements. In particular, Su *et al.* [SBFZ07] propose an approach to automatically derive service implementations from a choreography specification. The authors of [FT04] and [SHP03] present different approaches to semi-automatic services composition based on abstract functional blocks and semantic service descriptions, respectively. Ponnekanti & Fox [PF02] propose an automatic approach for service composition, using AI planning algorithms. Salaun [Sal08] strives for the same goal, however assuming that some services are reused. The proposed approach exploits wrappers to make the reused services match the specified choreography. Some studies [BTRR04, RBK05, CH06, ZPPN+07] investigate dependency management in a dynamic service composition scenario. Employed techniques include service and middleware instrumentation, use of self-healing rules, and establishment of a dependency-aware service-oriented architecture.

A limitation of these approaches is that some of them are specific to Web services or make off-line planning. Therefore, none of them addresses the multidimensional and ever-changing context of the ULS Future Internet.

3.4.2. Baseline

Regarding the CHOReOS objectives, scalability is a crucial aspect of the synthesis process. Scalability can be achieved by following a compositional approach, e.g., through assume-guarantee techniques [GL94, Pnu84]. To support compositional synthesis, we will define choreography patterns by considering as baseline the UDA work described in [AINT07], so that the choreography may be distributed in a set of wrappers. We intend to extend this approach by taking into account non-functional properties of the networked services. By relying on suitable mechanisms such as the ones proposed by INRIA and UOI to organize hierarchically the vast amount of available services [AZI09] (see Section 3.2.3) and by considering the coordinator synthesis approach described in [TII08] as a baseline, we will introduce model transformation techniques [MHZJ06, HMY06] 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* coordinator that, accounting for service functional and non-functional abstractions, enforces the composition of the discovered services to adhere to the specified choreography.

As in the case of the CHOReOS conceptual model and architectural style we shall further rely on the NEXOF reference architecture and specifically the composition concern that specifies the basic means that should be offered by a service-oriented development platform for the creation and management of service compositions. This particular concern should be adapted in the context of CHOReOS with a specific focus on large scale service choreographies.

3.4.3. Progress

The fact that the state of the art approaches in choreography synthesis rely on centralized choreographer models clearly points out that the existing approaches cannot deal with the multidimensional scale up of the Future Internet. Within WP2, CHOReOS will investigate this issue to come up with novel methods and tools for choreography synthesis allowing for an efficient decentralization of the choreography.

We will consider automata (Labeled Transition Systems) as the formal model, where each transition is labeled with, e.g., the service operation name (plus I/O data). For addressing QoS, the transitions are annotated with QoS attributes expressed in a way that allows their combination with QoS attributes from other services during the synthesis process. Current approaches do not address QoS of the synthesis process. In the baseline work, a suitable service abstraction is assumed, predicating on both functional and non-functional (so far, only real-time attributes) characteristics of the service interaction behavior. Then, by exploiting this abstraction, the synthesis process automatically derives a centralized coordinator that not only guarantees, if possible, the specified functional interaction, but also end-to-end timeliness (e.g., latency, WCET, etc.).

For CHOReOS, we plan to extend the baseline work to produce a distributed choreographer that guarantees non-functional properties beyond end-to-end timeliness, e.g., performance and reliability. The proposed approach shall rely on the CHOReOS hierarchical service base (Section 3.2) where services are grouped into classes of functional and non-functional abstractions, to overcome the ultra-large number of services. Furthermore, compositional reasoning, relying on assume-guarantee techniques, will be used to refine abstract choreographies in order to map to the actual networked services that can be possibly used and adapted by the choreographer. We will also consider and adapt object-oriented tools, models, and metrics to deal with choreographies, connection intermittency, scalability, platform, and location variability.

3.5. A Model for Choreographies: Analyzing and Predicting Choreography Scalability

Scalability has been studied in many works on parallel computation. A consensual definition is the possibility for an application to keep a good speedup as either its work or number of resources increases [KG94, JW00]. Our approach is to start from this initial knowledge to propose a model for scalability of choreographies.

This section is related to Sections 3.1 and 3.2. In Section 3.1, the notion of models is extensively developed given that CHOReOS adopts an MDD-based development process. Then, the specific model for scalability that we will propose will form an integral part of our MDD approach. In Section 3.2, there is also the notion of scalability, however, in the scope of service base management while we focus here on the scalability of choreographies.

3.5.1. State of the art

Analysis of computation scalability on distributed environments has been carried out in many studies. Most of these works, however, are based on a classical parallel computing paradigm [KG94, JW00] or on a grid or cloud computing paradigm [VPB09, FZRL09]. An interesting framework for scalability analysis was proposed in [DRW07]. One of its main interests is that it may be adapted to a broad class of computation paradigms.

On the SOC paradigm, there are works on scalability [GT06, Eng05] and performance modelling of Web services and Web servers [Dil06, GSCL+04, AW97, UPSS+07]. These works do not focus directly on choreographies. Other works studied other aspects of choreographies such as enforceability and realizability. In these works different abstract representations are proposed for choreographies, mainly: Petri Nets [DW07, Dec09, Wu09],

Open Nets [Dec09], Automata [SBFZ07] and Pi-Calculus [PW05], or more elaborated languages for choreography representation [QZCY07, BK06, DDO08, Gro07].

3.5.2. Baseline

We will consider four main stages in analyzing scalability of choreographies. The first stage is the elaboration of a model for scalability and QoS of choreographies. For this, we adopt the scalability model proposed in [DRW07]. The framework proposes to consider a model of scalability as composed in an abstract point of view by two elements: scaling dimensions and dependent variables. Scaling dimensions are variables that may vary during the execution of an application.

The second stage concerns the development of a mathematical model for representing choreographies. We will consider mainly two representations there: the usage of a graph model [Fer04, FGM10] or the utilization of a Petri net [DW07, Dec09, Wu09].

The third stage concerns the development of a theoretical model for predicting scalability and QoS of choreographies. This model exploits the two previous ones. For deriving it, we may use a learning process or stochastic automata networks (SANs) [PA91].

The fourth stage concerns the elaboration of an experimental framework for evaluating the theoretical model developed for QoS. To assess the prediction model proposed for choreographies, we will generate and execute choreography representations. The results obtained will be used to estimate the quality of the proposed model. One difficulty here is obtaining an experimental environment. We will proceed in using a simulator like SimGrid [CLQ08], Ns-2 [NS2], OMNET++ [OMNET], or the SMART package [Smart].

3.5.3. Progress

We just started the proposition of a scalability model. This includes the definition of the scaling dimensions, the definition of QoS parameters, and a model for scalability. With this definition, one can judge the scalability of an application by analyzing the behavior of dependent variables when varying scaling dimensions.

The scalability model then comprises three steps:

- 1) The identification of scaling dimensions for choreographies on the Future Internet;
- 2) The definition of dependent variables;
- 3) The definition of a measure for scalability as the effect of change in a scaling dimension on dependent variables.

The first step consists in determining an inventory of elements that may change the QoS of enacted choreographies. We can already propose some of these elements such as number of nodes, number of user requests, number of node faults in a period, latency as perceived by the final user, used bandwidth, etc. Similarly to the first step, the second one consists also in creating an inventory. Some variables that can be measured here are the mean time to fulfil a request, the number of requests executed in a period, the speedup in the computation, etc. In the third step, we have to provide an idea of a scalable choreography given dependent variables and scaling dimensions. At least two suggestions can be considered here. The first is to consider that given different values of scaling dimensions, one would obtain a result on dependent variables. The second is to use vectors whose components are scaling dimensions and dependent variable values represent the scalability.

The modelling of scalability and QoS is the first stage of our project. For it, we propose to adopt the approach proposed in [DRW07]. Our task will then consist to adapt the framework proposed there to choreography enactment on the ULS Future Internet. At the end of this stage, we will have a mathematical model for scalability analysis. We will then study how it can be applied in practice for choreography design, implementation, and dynamic reconfiguration.

4. WP3: Service-Oriented Middleware for the Future Internet

The execution of choreographies within the Future Internet heavily relies on adequate middleware support that must: (i) leverage the diversity of networks and networked resources that get federated, and (ii) implement the service-oriented abstractions subsumed by the CHOReOS dynamic development process and associated methods and tools. Providing such a support forms the core objective of WP3, which relates to:

- Service-oriented middleware (Section 4.1);
- Distributed service bus (Section 4.2);
- Middleware solutions for the Internet of smart things (Section 4.3);
- Scalable service provisioning using Grid and Cloud computing technologies (Section 4.4); and
- Choreography adaptation (Section 4.5).

4.1. Service-oriented Middleware

The Ultra-Large-Scale of the Future Internet significantly affects the middleware support that is needed for the enactment of Future Internet choreographies. The ultra-large number of users and services, and the high degree of heterogeneity of choreographed services, whose hosting platforms may range from that of resource-rich, fixed hosts to wireless, resource-constrained devices, challenge the middleware for choreography enactment. These challenges call for considerable advances to the state of the art of the Service Oriented Computing (SOC) paradigm, adopted by CHOReOS. This section introduces our overall viewpoint of Service-oriented Middleware for the Future Internet, while specific aspects are further detailed in Sections 4.2 to 4.5.

4.1.1. State of the art

Enacting Future Internet choreographies first of all raises the issue of choreography execution. While execution has been well addressed for business orchestrations based on BPEL with a multitude of execution engines, often free and open source [ODE, ORC, ACTBP, PXE, DFK04, Hac06, ORACBP], choreography is mostly considered as a design artifact rather than an implementation artifact. Nevertheless, the authors of [KWH07] propose a (probably unique) WS-CDL execution engine aimed at enabling testing and evaluating the properties of WS-CDL with practical use cases. The engine enables simulating choreographies on a single machine, which facilitates testing and debugging WS-CDL documents. This work additionally extends WS-CDL to WS-CDL+ to resolve some usability weaknesses. We further introduce the CHOReOS view on choreography execution in Section 4.2. In addition, executing choreographies in a highly dynamic, Ultra-Large-Scale environment as the Future Internet, raises the issue of choreography adaptation; we discuss this issue in Section 4.5.

Regarding the high degree of heterogeneity of Future Internet choreographed services, ESB platforms constitute the widely accepted emerging solution. ESBs cover a wide variety of capabilities including service management, service orchestration, messaging, and QoS provisioning. The state of the art on ESB platforms comprises open source ones, e.g., OpenESB [OPESB], Apache ServiceMix [SMIX], FUSE ESB [FUSE], Bostech ChainBuilder ESB [BTECH], Mule [MULE] and OW2 PETALS [PETLS], as well as closed source ones, e.g., Progress Sonic ESB [PSONC], IBM Websphere ESB [WBSPH], and Oracle ESB [ORESBS]. In the highly decentralized Future Internet environment, distribution of ESBs is of key importance; we develop our view of Distributed Service Bus (DSB) in Section 4.2.

Given the prevalence of mobile networking environments and powerful user handhelds, CHOReOS considers resource constrained devices and Things, although focused on smart

things, as first-class entities of the Future Internet. The CHOReOS view of pervasive computing and the Internet of Things is detailed in Section 4.3.

Concerning the requirement of supporting the ultra-large number of users of Future Internet services/choreographies, Grid and Cloud technologies appear as a promising solution, since their inherent characteristic is the ability of handling very large load. Computational Grids have received great attention from both academia and industry [FK03, CGCK06]. Using middleware and virtualization technologies [Kro09], it becomes possible to instantiate homogeneous software platforms, isolate the resources that each grid application is allowed to access, and determine the amount of resources provided to each application. Cloud computing further emerged as a new paradigm for providing Utility Computing [Hay08]. In the Cloud Computing model, enterprises provide computational resources (such as CPU cycles and storage space) on-demand and charge the customers for the amount of resources used. We further detail our view to scalable service provisioning based on Grid and Cloud technologies in Section 4.4.

Last but not least, the heterogeneity and ultra-large scale of the Future Internet have a direct impact on coordination among interacting entities. Our choice of choreography as global coordination style among services should further be underpinned by support for and interoperability between heterogeneous coordination/interaction models, such as message-driven, event-driven, and data-driven models. Different coordination models apply to different needs, for instance, asynchronous, event-based publish/subscribe is more appropriate for highly dynamic environments with frequent disconnections of involved entities. Enabling interoperability between such models is imperative in the very heterogeneous Future Internet integrating Services, People and Things. Interoperability efforts are traditionally based on, e.g., bridging communication protocols, where the dominant position is held by ESBs, wrapping systems behind standard technology interfaces [AG09, LLJC08], and/or providing common API abstractions [GBS05, PEKS07]. However, such efforts mostly concern a single coordination model and thus do not or only poorly address coordination model interoperability. Efforts combining together diverse coordination models include: implementing the LIME tuple space middleware on top of a publish/subscribe substrate [CMP08]; enabling Web services SOAP-based interactions over a tuple space binding [WML08]; and providing ESB implementations based on the tuple space paradigm [BKMTS05, Mar06, MKS10].

4.1.2. Baseline

The CHOReOS middleware in general shall incarnate functionalities for the enactment of large-scale service choreographies. These functionalities shall conform to the different concerns, prescribed in the NEXOF reference architecture and refined/extended in the CHOReOS conceptual model and architectural style (see Sections 2.1 and 2.2). Specifically, the NEXOF service concern is related to the CHOReOS middleware since it refers to middleware functionalities for the execution of services; the message concern relates to middleware functionalities for interaction and interoperation; the discovery concern considers middleware functionalities for the discovery of services; the composition concern focuses on middleware engines for the execution of composite services; finally the management and the resource concerns refer to middleware functionalities for, deployment, configuration, QoS management, monitoring, resource allocation, and tuning.

The development of the CHOReOS middleware, from design to implementation, will extensively build on the expertise of CHOReOS partners, in particular EBM, INRIA, UOI, and USP, on all of the technologies that appear as appealing solutions to cope with the various scalability issues introduced by the Future Internet upon service-oriented middleware. This rich background will serve as an overall baseline for the CHOReOS service-oriented middleware.

More specifically, EBM has developed the PEtALS Enterprise Service Bus. The main PEtALS feature is that it extends the Java Business Integration (JBI) standard specification by providing a distributed version of the JBI platform. Still, PEtALS is not a simple JBI container; it also provides various frameworks, components, and tools for extension, service integration, management, and monitoring purposes (see Section 4.2).

INRIA has long expertise on service-oriented middleware for pervasive computing environments that covers from lower-level cross-layer networking to higher-level semantics of services, as well as transversal concerns such as context and privacy. In the context of the INRIA-UOI collaboration, work has been done in the field of context aware-middleware [AZIPV08], as well as pervasive service-based systems adaptation [ZFGI06]. The produced research results are further discussed in Sections 4.3 and 4.5.

USP has led the development of InteGrade, an open-source Grid Computing middleware infrastructure for leveraging the idle time of existing computers in organizations [GKGFC04, CFK09]. USP further carries out research to extend the scope of InteGrade to cope with the Cloud Computing model (see Section 4.4).

Finally, INRIA has been working on providing solutions to the coordination model interoperability and integration problem in the context of service orchestrations as part of the ANR ITeMIS project [ITEM]. The introduced approach is based on multi-level model abstractions enabling, first, to map diverse middleware platforms to their respective coordination models, and then, to map such models to a single generic application coordination/programming model. Mappings are performed at design-time and allow deploying orchestrations that are agnostic to the underlying middleware platforms of the constituent heterogeneous systems.

4.1.3. Progress

Existing ESB, Grid/Cloud, and pervasive middleware technologies emerged independently to cope with different scalability issues. Nevertheless, the Future Internet calls for an integrated solution. To this end, CHOReOS will build upon the aforementioned baseline of the individual CHOReOS partners to refine individual solutions so that they meet the challenges of the Future Internet. The goal is indeed to develop a unified middleware infrastructure that enables: (1) service provisioning for the ultra-large number of Future Internet users based on available Grid and Cloud technologies, (2) networking an ultra-large number of heterogeneous services *via* ESB-based middleware, (3) networking services from the Internet of (smart) Things based on middleware for pervasive networks, and (4) execution and adaptation of dynamic service choreographies, adopting the models@runtime approach promoted by the CHOReOS MDD-based development process. Our foreseen progress in each one of these aspects is detailed in the following sections. In particular, to address interoperability of different coordination models, we will extend our current solution, linked to the orchestration paradigm, to dynamic, ultra-large scale choreographies. This requires decentralizing our approach and enabling runtime model mappings and related transformations. The work outlined here and further detailed in the following sections will be undertaken within the dedicated work package WP3.

4.2. Distributed Service Bus

The Enterprise Service Bus (ESB) concept is a follow-up of the Enterprise Application Integration (EAI) framework. The main goal of the EAI solutions is to provide integration of heterogeneous applications. ESB technology leverages best practices from EAI mechanisms and service-oriented architectures (SOA). Remote applications integration is based on sophisticated mechanisms such as brokering, message transformation and routing, quality of service support, service orchestration, etc. Moreover, ESB offers a backbone implementation for service-oriented architectures by connecting distributed services consumers with providers [SHLP05]. The main functionalities of an ESB are supported by components that

can be both centralized or distributed over a network. Besides, ESB technology allows several ESB to be connected over a large network, achieving a pervasive integration, which is strongly required in wide and dynamic systems. Other key characteristics are listed in Chappell's book [Cha04] namely, standardization, service discovery, high distribution, and reliability. They make ESB a promising technological choice for future systems. Finally, standard-based ESB concepts are strongly attracting industrials.

4.2.1. State of the art

Existing proprietary and open source ESB approaches provide powerful integration and orchestration solutions for distributed services. For instance, the BizTalk server [Biztalk] is a proprietary Microsoft ESB based on an extended server. A toolkit offering a collection of tools and libraries supporting a loosely coupled and dynamic messaging architecture is provided. It functions as a middleware that provides tools for mediation between services and their consumers. Open ESB [OPESB] is Java-based open source enterprise service bus providing application integration. It supports open standards such as SOAP, WS-*, XML standards, and a NetBeans-based IDE. Both BizTalk server and Open ESB rely on centralized architectures.

However, even if this generation of ESB provides powerful means of integration and services orchestration, it is still not adapted to highly distributed environments. Therefore, a more flexible and dynamic vision is needed to cope with scalable distribution and particularly scalable choreography. Actually, widely distributed systems are all about the large size of all the surrounding dimensions including persons dealing with the system, amount of data routed, number of services interacting, etc. Brokering and routing such a large number of services using a single centralized bus creates a bottleneck and a lack of flexibility. These issues can be tackled through ESB distribution. It represents a key point since the integration logic can be broken-up into distinct manageable pieces. It is worth underlining the fact that, when compared to the centralized ESB, the distributed ESB (DSB) is more adapted to widely distributed environments.

There are both open source and proprietary distributed enterprise service buses. Based on the Apache Service Mix ESB, Fuse ESB [FUSE] is an open source OSGi-based distributed ESB. It supports BPEL processing, and both OSGi and JBI [JBI] deployment and runtime. It is said to be distributed since it offers a remote console to control the runtime bus. From our point of view, the Fiorano ESB [Fiorano] has a different and more scalable vision of distribution. It is built upon a hybrid architecture relying on a hub and spoke management layer and on a peer-to-peer system. This allows ESB peers dissemination over a wide network. It supports main ESB features such as application integration, service orchestration, and event management. It achieves services integration and distributed operations management. Moreover, it focuses on security in distributed domains and achieves BPEL orchestration. Sonic ESB is a proprietary product.

Petals ESB [PetalsLink], developed by the CHOReOS partner EBM, is an open source JBI-based distributed ESB. It provides a multi-node platform leading to a moderate scalable architecture. It handles BPEL process orchestration, business process monitoring and management, as well as heterogeneous application integration through several protocol connectors. Around the bus, a set of open source tools providing process design and configuration, and services management and monitoring, are offered. Distribution is also addressed through the leveraging of the bus service registry. In [CV08], a remote and wide service discovery is achieved by distributing the service registry. Although this solution is interesting due to the way that it helps a more global service discovery and awareness, a larger vision is required. In order to adapt to distributed systems, Petals ESB combines both a distributed service registry and a multi-site architecture.

4.2.2. Baseline

Beyond their size, ULS systems raise challenging research areas. These are namely: i) design and evolution, ii) orchestration and control, as well as iii) monitoring and assessment [ULS06]. Within CHOReOS, emphasis is put on choreographing services in ULS systems. Although the listed ESB solutions ensure service orchestration, they do not address scalable service orchestration and choreography. They need to "scale-up" in order to answer ULS requirements. Petals ESB features are continuously improving through innovative works. Current research work in the FP7 SOA4ALL European project (www.soa4all.eu) includes dealing with evolving Petals distributed ESB to ESB federations for large scale SOA [BFHL+10]. Registries and message routers are scaled to the level of federations. Additional communication and coordination capabilities have been implemented using global semantic space paradigms. These improvements empower Petals ESB capabilities to cloud architectural requirements. Nevertheless, in order to meet the threefold ULS research areas mentioned before, further improvements need to be made:

First, Petals ESB needs to evolve to be compliant with ULS requirements and in particular distribution and high variability. In such environments, users and systems are a continuous source of changes and events. Consequently, a dynamic and distributed infrastructure is strongly required. CHOReOS bus has to provide an elastic and agile architecture hosting running services and choreographies. Meanwhile, we need to provide a centralized view on the running processes and services.

Second, when dealing with large distributed choreographies a custom choreography engine needs to be implemented. For the moment, on top of the Petals ESB there is an orchestration engine able to handle BPEL-like processes, as process orchestration is a main feature in ESB solutions. Nevertheless, this engine needs to be able to handle complex and distributed choreographies. Moreover, as we are dealing with a huge number of services, it is very important to be able to identify those respecting their roles in a given choreography.

4.2.3. Progress

To achieve the aforementioned goals, we will consider, as a starting point, the Petals ESB as our basis for the following reasons. Petals ESB is built on top of agile and modular JBI and Fractal frameworks. The bus benefits from both technologies and is extensible, modular, and natively integrates distribution concerns. Moreover, its open source license allows it to evolve more easily as Petals community members are contributing. Petals ESB is more willing to fulfill ULS systems requirements for its modularity, extensibility, and distribution readiness. Nevertheless, we will still need to improve our architecture by introducing cloud-aware service deployment. Overall, we identify the following main goals for evolving our ESB technology towards supporting ULS choreographies in the Future Internet:

- *The assessment and evolution of the DSB technology.* Petals ESB needs to support specific aspects of ULS systems. These are namely, high distribution, high variability, and heterogeneity. Petals ESB needs to enforce its distributed architecture in order to choreograph disseminated services. Besides enhancing distribution concerns in Petals ESB, an interesting direction to consider is the synergy with cloud computing environments as they provide a valuable enhancement for scalable and elastic infrastructures. This work will be done in close cooperation with our work sketched in Section 4.4 on Scalable Service Provisioning using Grid and Cloud Computing Technology
- *The development of a scalable service registry.* Leveraging the enterprise service bus must be accompanied by enhancements in the service registry capabilities. Service registry on top of the DSB needs to be scalable and able to handle a large number of heterogeneous services. Service registry concerns are addressed in WP2 dealing with the Dynamic Development of Adaptable, QoS-aware ULS Choreographies (see Section 3.2) and will inform the specific solution to be developed on top of the CHOReOS DSB.

- *The development of a custom interaction protocol.* We need to develop an interaction protocol that enacts scalable communications. Event-Driven Architectures (EDA) represents an ideal framework for supporting independency between business process steps and they are fully compliant with ESB solutions. Petals DSB needs to enhance already supported EDA-based solutions to cope with ULS requirements.
- *The development of a custom choreography engine.* This involves identifying the main choreography paradigms and studying the current choreography languages, in close cooperation with WP1 and WP2 work. Languages such as WS-BPEL are dedicated to service orchestration more than to choreography, thus we need to leverage the current language or to find an adequate one. A highly distributed choreography flow is needed. Works such as [YLG07] and [HSBP10] may be helpful. For instance, in the first paper, the authors propose a transformation technique that converts a process conceived for centralized execution to a set of nested processes deployed on dynamically bound services. In the second paper, an event-driven architecture is adopted in order to support communication in decentralized orchestrations. As a consequence, the global process flow is transformed into an event-based orchestration. Petals DSB will account for existing works together with development of CHOReOS work packages WP1 and WP2, to develop the CHOReOS choreography engine.

Thanks to the aforementioned progress, the outcome will then be a highly scalable DSB that allows for choreographing heterogeneous services.

4.3. Middleware Solutions for the Internet of Smart Things

The Internet of (Smart) Things [IOTFR, IOTSRR] concept consists of a highly dynamic and heterogeneous networking environment, which is obviously embodied within the Future Internet vision. In the context of the CHOReOS project, we more specifically concentrate on the Internet of Things as an Internet of resource-constrained devices, whose heterogeneity spans from simple RFID tagged articles to smart components like sensors and actuators, to rich computing devices (such as servers), although we will not address the networking of tagged articles within the time frame of the project. As shown in the following, while there has been a lot of middleware research to enable application development in networks containing large numbers of sensors and actuators, comprehensive solutions to enable their integration in the larger systems found in the classical Internet has been lacking.

4.3.1. State of the art

The Internet of smart things consists mostly of devices that are either a source of data (sensors), or receivers of control messages (actuators). Recently, there has been a lot of work to standardize the network layer protocols on such devices, leading to the specification and adoption of 6LoWPAN (<http://tools.ietf.org/wg/6lowpan/>), which brings IPv6 to low powered devices. It is beginning to be adopted, and is expected to be available for a large class of “smart” things soon, thus providing basic network-layer support for communication of data between them. Additionally, there is a large amount of work to provide the programmer of these systems easy means to access the data generated and needed by these devices. Initial work included Hood [WSBC04] and TeenyLIME [CMMP07], which allowed data-sharing over a limited spatial range. Further work such as [MPBP+07] proposed the use of the DART runtime environment, which exposes the sensor network as a distributed data-store, addressable by using logical addresses such as “all nodes with temperature sensors in Room 503”, or “all fire sprinklers in the fifth and sixth floors”, which are more intuitive than say, IP addresses in the Internet of smart things domain. Note that most of the work above focuses on designing applications that exhibit only intra-network interactions, where the interaction with the outside world is only in the form of sensing it, or controlling it by actuation. The act of connecting this data to other systems outside the sensor network is mostly done using an external gateway. This is then supported by middleware that expose

the sensor network as a database [MFHH05, YG02], allowing the operator to access the data using an SQL-like syntax, augmented with keywords that can be used to specify the rate of sampling, for example.

Taking a different approach towards handling the data in the sensor networks, some middleware propose to manipulate them using semantic techniques, as done in [GL10], which extend the tsc++ [KBSF09] approach to *Triple Space Computing*, and model the data shared by the nodes in the system as RDF triples (subject-predicate-object groups), a standard method for semantic data representation. They propose to make these triples available to the participating nodes using a tuple space, thus giving it the “triple space” moniker. S-APL [KT08] or Semantic-Agent Programming Language uses semantic technologies to integrate the semantic descriptions of the domain resources with the semantic prescription of agent behavior. This can be used to support interoperability among the heterogeneous components of the Future Internet of Things. The authors have further developed their work in the context of the UBIWARE project (http://www.mit.jyu.fi/ai/OntoGroup/UBIWARE_details.htm).

Another possible area for middleware support in the domain of Internet of smart things comes from the fact that a particular sensor or actuator may not be always available. This leads to the need for transparent reconfiguration, where the application developer should not have to care about reliability issues. The work in [Ing09] proposes the PIRATES event-based middleware for resource-rich nodes (hosting sensors/actuators, or just processing data), which includes a third-party-remapping facility that can be used to remap a component’s endpoints without affecting the business logic. In that sense, it is similar to the RUNES [CCGL+07] middleware targeted on embedded systems.

Taking cognizance of the fact that these smart nodes need to also interact with services in the outside world, researchers have recently made attempts for their integration with larger systems such as Web servers. One of the directions in this work has been towards using the REST (REpresentational State Transfer) [Fie00] architectural style, which is already used to access services on the Web as an alternative to SOAP (see Section 2.2). The work in [Sti08] proposes a system that will enable heterogeneous sensors and actuators to expose their sensing and actuation capabilities in a plug and play fashion. It proposes a middleware that defines a set of constraints, support services and interaction patterns that follow the REST architectural style principles. They propose the usage of the ATOM Web publishing protocol to describe the functionalities provided by these systems, and a two-step discovery process within this system. In [GT09], the authors present an implementation of a REST-oriented middleware that runs on embedded devices such as Sun SPOT nodes, and the Plogg wireless energy monitors (<http://www.plogginternational.com/>). They take a two-fold approach – embedding tiny Web servers in devices that can host them, and employing a proxy server in situations where that is not the case. However, they also noticed that the abstractions provided by REST might be too simplistic to compose complex applications over the services provided by smart things. Some of the most recent work in this area includes [BHMRT10], which proposes to convert gateways into smart gateways, by running application code on them; and [TRE10], where the authors have presented a component-based approach for service distribution in sensor networks, to integrate them with existing IT systems using the REST architectural style.

Most recently, the field of “participatory sensing” has emerged, where the smart “things” are increasingly the mobile phones carried by the users of the system, providing data captured using the sound, GPS, accelerometer and other sensors attached to them [LPLC+09, EMLP+07]. Projects and applications based on this paradigm include Censme (<http://www.cenceme.org>), UrbanSensing (<http://urban.cens.ucla.edu/>), and TrafficPulse (<http://www.geoide.ulaval.ca/projects-detail.aspx?i=131>), among others.

Finally, concerning middleware that enables networking mobile and/or resource constrained devices in pervasive computing environments [AHS01, Sat01] that pave the way for the Internet of Things [RDGM+08, VL08, CY08], several promising solutions have been proposed. They address issues such as resource discovery, resource access, proactiveness, location sensitivity, and adaptation and context awareness in a seamless manner, e.g., [CACM05, MZ09, KKS09, LD09, CC08, ZME06, PGXP+09, RRA08, CMMP07]. Other solutions specialize in vehicular sensor networks (related to one of CHOReOS' use cases) [DCJ06, DB08, SKWC08, SBFW+04]; to name a few. The tremendous European effort in the area of service-oriented middleware meeting the challenges of pervasive computing is evident in the various effort, including the AMIGO (<http://www.hitech-projects.com/euprojects/amigo>), PLASTIC (<http://ist-plastic.org>), SMEPP (<http://www.smepp.org/Default.aspx>) and MORE (<http://services.ist-more.org>) projects. Further, the recent DPWS standard [DPWS] is to be acknowledged as being very significant for the area.

4.3.2. Baseline

The work on integrating the low-power sensing and actuation nodes (the “smart” things) into the larger Internet of Things will take two approaches. For IP-capable devices, the work on 6LoWPAN (<http://tools.ietf.org/wg/6lowpan/>), which brings IPv6 to low powered devices, is highly relevant to our work. For embedded devices that still use custom radio and network protocols, we will build upon the work on smart gateway such as [BHPM+10].

The work on enabling logical-scope based addressing on systems consisting of heterogeneous nodes (sensors as well as PC-class nodes) done at INRIA on the Srijan toolkit [PG09] will provide a good basis to work upon. Additionally, INRIA has worked on several aspects of service-oriented middleware for pervasive computing environments including the semantic discovery and composition of services, the heterogeneity of service discovery protocols, the heterogeneity of network interfaces, and privacy [ISTS+05, MLGI05, BI05, RRLC+06, MKGI06, CRI07]. Especially, the iBICOOP middleware developed by INRIA [BSRI09] that allows communication between mobile devices having various types of network access (WiFi, 3G, etc.) running a variety of operating systems (iOS, J2ME, Android) is expected to provide a sound foundation as we work to solve the interoperability problems in the Future Internet of Things.

Since REST has proven to be very crucial for wide adoption by programmers of Web applications, our research will evaluate its relevance in the sensor world and the greater Internet of Things [STR08, GoT09]; especially comparing it with the work on DPWS [DPWS], which provides a much richer set of semantics for messaging, discovery, and eventing on resource-constrained devices in a secure fashion. We will also take into account the work on seamless service remapping such as that introduced in [Ing09].

4.3.3. Progress

In the course of our work on CHOReOS, we will follow the baseline discussed above to address three main related challenges arising from the incorporation of smart things into a larger scale.

- *Data-oriented abstractions.* In the world of smart things, it is natural to write applications that talk about data – which is captured from the environment using sensors, and then processed and finally converted to action by the actuators. Applications for other systems (e.g., using Web servers), however, use a variety of interaction paradigms, ranging from messaging to event-based interactions. Our research in CHOReOS will aim to bridge the gap between these abstractions using our middleware.
- *Extremely large scale.* The future internet of smart things, as targeted by CHOReOS, will consist of many more physical nodes as well as interacting software components

than addressed by currently available systems. Our work over the next few years will focus on addressing issues such as resource discovery, resource access, proactiveness, location sensitivity, adaptation, and context awareness at such large scales.

- *Involving (mobile) users in the sensing process.* Finally, our work in the middleware for the Internet of smart things will take into account the increasing proliferation of smart phones with complex sensing capabilities. We acknowledge the current lack of a structured manner for the description and utilization of both the sensing capabilities, as well as the sensed data from these devices, and will work on providing application developers with better abstractions and middleware support to fully utilize the participatory sensing abilities available in the near future. This will involve treating mobility as a first-class problem to be addressed in our research.

4.4. Scalable Service Provisioning using Grid and Cloud Computing Technology

Grid Computing [FK03,CGCK06] emerged in the 1990s by leveraging previous technologies developed by the Distributed Systems and High-Performance Computing communities to enable the execution of computationally-intensive applications in collections of geographically distributed clusters of machines. This research led to the development of multiple Grid Computing. Middleware such as Globus [FK97], OurGrid [CBAC+06], and InteGrade [GKGF+04, SKGF+10]. These systems addressed the problems of managing a heterogeneous collection of computational and storage resources dispersed across multiple administrative domains. Grid middleware is also responsible for scheduling the execution of tasks, collecting application results, and returning them to their users. In most cases, scientific applications with large processing and data management needs were the motivation for Computational Grids.

4.4.1. State of the art

More recently, a new generation of middleware systems and hardware infrastructures were developed to cope with other kinds of applications requiring large processing power. In this case, the motivation was end-user applications such as email, office suites, calendars, and many other e-commerce, social networks, and Web 2.0-style applications used daily by hundreds of millions of users. Large Internet-based companies such as Amazon and Google used virtualization technologies to develop the basis for what was later called Cloud Computing [ZCB10].

In the Cloud Computing domain, one can find three major types of service models:

- 1) *Infrastructure as a Service (IaaS)* is a model in which the IaaS provider offers infrastructural resources on demand. Normally, the provider offers an API via which users can request a certain number of virtual machines with certain resource characteristics. The software that is executed in these virtual machines can be selected from a set previously defined by the IaaS provider or can be customized by the user before requesting the service. Typical examples of commercial IaaS providers include the Amazon EC2 (<http://aws.amazon.com/ec2>), GoGrid (<http://www.gogrid.com>) and Flexiscale (<http://www.flexiscale.com>). In the research domain, the OpenCirrus initiative coordinated by HP provides an IaaS cloud combining resources from three companies (HP, Intel, and Yahoo!) and several academic institutions such as the University of Illinois at Urbana-Champaign and the Carnegie-Mellon University in the USA and the Karlsruhe Institute of Technology in Germany.
- 2) *Platform as a Service (PaaS)* is a model in which the provider offers higher-level services for executing applications, normally including development frameworks. Typical examples of commercial PaaS providers include Google App Engine (<http://code.google.com/appengine>), Microsoft Windows Azure

(<http://www.microsoft.com/windowsazure>),
(www.salesforce.com/platform).

and

Force.com

- 3) Finally, *Software as a Service* (SaaS) refers to providing applications on the Web as it is done by Google docs and calendar, Facebook, YouTube, etc.

The Amazon Elastic Compute Cloud or EC2 (<http://aws.amazon.com/ec2>) provides a virtual computing environment in which developers can instantiate multiple virtual machines booting, from scratch, standard operating systems such as GNU/Linux and Windows. These sets of machines can then run any application developed for these operating systems.

Google App Engine (<http://code.google.com/appengine>) provides an execution environment for Web applications that can then be executed in the hardware infrastructure provided by Google; developers can write applications in Java or Python and use standard APIs for storage and communication.

Both Grid and Cloud Computing technologies provide mechanisms to offer users large amounts of computational resources on demand. As the CHOReOS project intends to support the enactment of choreographies composed of a large quantity of services, used by a large number of users, involving a large number of devices, this scenario may impose an enormous demand on the middleware, which will need to be executed in a large number of machines. Grid and Cloud Computing technologies can be used to fulfil this need.

4.4.2. Baseline

InteGrade (<http://www.integrate.org.br>) is an open source middleware for opportunistic grid computing, developed by USP, which leverages the existing computational resources of organizations by using the idle time of existing machines to resolve computationally-intensive problems coded as parallel applications [GKGF+04]. InteGrade now supports multiple parallel programming models including MPI, BSP, and bag-of-tasks. Applications from different scientific domains have been executed in InteGrade and they have been written in a variety of programming languages, including C, C++, Java, FORTRAN, and Perl. InteGrade includes mechanisms for fault-tolerance, security, and intelligent scheduling based on user pattern analysis [FBC09,SKGF+10]. Given these advanced characteristics and the USP team's deep knowledge of this grid computing middleware, InteGrade becomes a perfect testbed for researching the synergies between a grid computing platform and large-scale choreographies in the Future Internet.

On the one hand, choreographies involving activities that are computationally-intensive by themselves such as image processing (e.g., face recognition, image classification) or route optimization in transportation applications can benefit from a computational grid by executing these tasks on the grid. On the other hand, the choreography middleware engine itself, as it becomes more intelligent, will need to perform computationally-intensive tasks for its own management. For example, machine learning algorithms can be used to learn about the execution environment dynamically to optimize the performance of the system on-the-fly. Adaptation techniques can also be used to analyze the millions of requests the middleware may receive and detect opportunities for improvement and dynamic reconfiguration. All these heavy computations can be performed on a computational grid.

CHOReOS will also benefit from Cloud Computing technologies as a means to execute the choreographies in a large number of machines distributed across the globe. In this way, the choreographies will be able to deal with millions of users in different countries and continents. To conduct this research, we will experiment with a variety of Cloud platforms, including the commercial Amazon EC2 platform, the open-core Eucalyptus (<http://open.eucalyptus.com>) and the fully open-source OpenNebula (<http://www.opennebula.org>). As an alternative hardware resource platform, we will experiment with Open Cirrus, a Cloud Computing testbed organized by HP (<http://www.opencirrus.org>).

4.4.3. Progress

The CHOReOS project will investigate the level of scalability that will be required by Future Internet applications and the level of scalability that can be provided by alternate architectures. Our initial target is to study choreographies composed of tens to hundreds of services, involving hundreds to thousands of computing nodes, and thousands to millions of users. We will evaluate how these different loads will impact the CHOReOS middleware and choreographies, identifying which parts of the system will become the bottlenecks in each case. These bottlenecks will then be analyzed to verify whether they could be resolved by applying grid and cloud solutions. To be able to perform these experiments, we will develop a framework to enable the construction of a set of synthesized choreographies whose size and topologies are provided as input parameters to the framework. In this way, we will be able to experiment with choreographies of sizes varying a few orders of magnitude.

An interesting research direction lies in combining Enterprise Service Bus solutions with cloud computing. ESBs can provide integration capabilities and the choreography engine, while the cloud supports the deployment of the system on a real network. In this way, research will be conducted in coordination with the work on the distributed service bus previously presented in Section 4.2. To benefit from both approaches, the Petals ESB can be deployed on an IaaS (Infrastructure as a Service) cloud architecture to build a PaaS (Platform as a Service) architecture. In this way, Petals ESB nodes will run on a powerful and elastic platform without having to care about resource allocation. Further studies will be conducted on the Petals ESB to extend it to be compliant with cloud architectures such as OpenNebula and Amazon EC2.

Finally, the overall goal of this task is to provide CHOReOS with the high-performance computing power available in Grid and Cloud Computing infrastructures. Thus, the computationally intensive processes that will be required to serve millions of users issuing thousands of simultaneous service requests to thousands of services will be able to be processed by Grid and Cloud services. The task will involve efforts in (1) Software Architecture and Engineering to implement the interaction protocols and choreography engines onto the specific context of Grid and Cloud middleware infrastructures and (2) investigation of the CHOReOS methods for creating, managing, processing, and adapting choreographies so that part of their computation can be delegated to high performance computing environments.

4.5. Choreography Adaptation

In an ultra large scale and highly dynamic environment such as the Future Internet, the choreography requirements and the properties of choreographed services are subject to constant and independent changes. For that reason, the adaptation of synthesized Future Internet choreographies to changes is amongst the main challenges of CHOReOS. Handling changes in the Future Internet choreographies may span from *dynamically substituting services* to *dynamically re-synthesizing parts of the choreographies*.

4.5.1. State of the art

The adaptation of service-oriented software, in general, has been a very active research topic during the past few years. Recently, [BCDK+10] provides an overview of the various adaptation strategies that can be employed towards adapting a service-oriented system. The documented strategies can be combined to handle changes in the requirements and/or the properties of the services used; they range from simple to more advanced ones.

The advanced strategies identified in [BCDK+10] include the following, which are further detailed in the next paragraphs:

- *Service substitution*: involves substituting a service with another one.
- *Re-composition*: involves an overall reorganization of the system.

- *Re-negotiation*: amounts to re-negotiating service-level-agreements with services.
- *Re-execution*: in this case, the system is rolled-back to a previous consistent state and certain tasks (e.g., service invocations) are re-executed.
- *Compensation*: involves undoing the effects of certain tasks (e.g., service invocations) that took place before the occurrence of an adaptation situation.

Service Substitution

The approaches for service substitution in general heavily rely on the fundamental Adaptor design pattern [GHJV95]. The basic concept is to derive a mapping between the target service, that should be substituted, and a substitute service, that offers similar functionality through a different interface. Based on such a mapping, an adapter is generated, which allows accessing the functionality of the substitute service, through the original target interface. Based on this key concept, the state of the art approaches can be divided in two different categories.

In the first category, *we have approaches that analyze the interfaces of the target and the substitute services*, towards finding incompatibilities and mappings between the operations of these interfaces. Following, based on the analysis the necessary adaptors are generated (semi)automatically. In [PF04], for instance, the authors propose a solution for the semi-automated derivation of pair-wise mappings between a target service and a substitute service. The proposed technique relies on the assumption that the interfaces of the target and the substitute services are derived from the same popular, or standardized interface. In [AZI09], we propose an approach that automates the derivation of mappings between the target service and the substitute services, without making any assumptions regarding the interfaces of the services involved. Moreover, in [NBMC+07,KNBC+09,NXB10], the proposed framework provides mechanisms which allow detecting both structural and protocol incompatibilities. The resolution of these incompatibilities and the generation of adapters are guided by the users. Protocol incompatibilities detection is also supported in the approach that is proposed in [CD08,CNP10]; in this case the detection techniques are fully automated.

In the second category, *we have approaches that actually assume the existence of mappings between the interfaces of the target and the substitute services*. Given this assumption, the issue is that for a given target service there may exist multiple alternative substitute services, characterized by certain quality attributes. The goal is to retrieve these alternatives and make the optimal choice, with respect to the system's requirements and the values of the quality attributes that characterize the alternative substitute services. For instance, [MRD08] and [CVRG09] are similar approaches. The service-oriented system is configured as a BPEL orchestration, while the mappings between the alternative substitute services are specified in terms of XSLT transformation rules. In [AJ10] the authors also assume BPEL orchestrations, while the mappings are specified within the services SAWSDL descriptions, with respect to a common ontology. The semantic concepts that characterize the target and the substitute services should be equivalent. In [KDWL09], the authors also assume a common ontology for deriving the mappings between the interfaces of the target and the substitute services. However, the proposed approach is based on more advanced reasoning, concerning the relations that should hold between the semantic concepts of the target and the substitute services. Finally, [LLRB10] also relies on a common ontology that is used to define mappings between service interfaces. Nevertheless, the proposed approach focuses more on the detection of changes that trigger the need to adapt the system, than to the actual adaptation of the system.

Re-composition

In the typical re-composition approaches, a system is considered as an abstract orchestration, where each particular task may be performed by a given set of alternative

services that can serve as substitutes for each other. The different alternative services are characterized by different values of certain quality attributes. The overall quality of the system is measured with a global score function, whose values are calculated, with respect to the values of the quality attributes of the orchestrated services. Then, the objective of the re-composition approaches is to deal with changes in the system's requirements and/or changes in the values of the services quality attributes. Dealing with such changes amounts to calculating an optimal configuration for the orchestration that satisfies the system's requirements, while maximizing the value of the global quality function. The first approach in this line of research was proposed in [ZBND+04]. The global QoS optimization problem is solved using an integer programming optimization technique. The approach proposed in [AP07, ACMP+07], follows a similar direction; an integer programming optimization technique is used to solve the problem. However, in [AP07, ACMP+07] the authors further consider cases where the same service should be assigned to different dependent tasks. Differently from the aforementioned approaches, in [CDEV08] the authors propose using a genetic algorithm towards solving the global QoS optimization problem involved in the re-composition of a given service orchestration. Moreover, in [ZZL09] the authors propose a method that finds sub-optimal solutions to the global optimization problem. The primary motivation for the proposed method is that the complexity of the methods that find optimal solutions is very high and consequently, the overall performance of these methods is poor. Going one step further, in [CCGL+09, CI10] the proposed approach solves the global QoS optimization problem for sets of independent orchestrations.

Aiming at performance appropriate for dynamic, user-interactive pervasive environments, the work reported in [BBKG+09] introduces a heuristic that is based on clustering techniques for first ranking the best candidate services per task, which are then used to guide selection of a set of near-optimal service compositions. Thus, having more than one possible composition available allows dynamic, just-in-time binding of services and/or re-composition based on actual availability of services and potentially on QoS monitoring, which is appropriate for highly changing environments.

Finally, in [CDFP10] the authors see service-oriented systems as choreographies of services. Specifically, they propose a method for the re-composition of systems that consist of tiles. A tile corresponds to a particular service that may require using other services. The goal is to find the optimal composition of tiles, based on a global quality score function. The problem is solved by employing an integer programming optimization technique. Similarly to approaches that focus on orchestrations, the proposed approach also exhibits poor performance, as the number of tiles scales up.

Re-negotiation, Re-execution & Compensation

Re-negotiation, re-execution and compensation are not standalone strategies, in the sense that they typically support corresponding service substitution or re-composition strategies. In particular, in [AP07] and [ZZL09] the authors state that re-negotiation of service-level agreements takes place if it is not possible to find an optimal configuration of services that satisfies the QoS requirements of the given system.

Moreover, in [ZFGV06, FGIZ08, FZGI10] we focus on the dynamic substitution of stateful services that become unavailable during the execution of a service orchestration. To deal with this problem the proposed solution discovers candidate substitute services out of a set of semantically compatible services that can be used in place of the service that becomes unavailable and selects one amongst these candidates that can be used as an actual substitute; in the best case the selected substitute service is such that its current state can be synchronized with the state of the service that is substituted. In this case the orchestration continues its normal execution. On the other hand, if the state of the substitute service cannot be synchronized with the state of the unavailable service, then the proposed method

finds the orchestration tasks that used data obtained from the unavailable service and rolls back their execution.

4.5.2. Baseline

Our baseline towards dealing with choreography adaptation in CHOReOS shall include the recent advances in adaptation strategies for service-oriented software, in general, and our experience in service substitution [AZI09], re-execution and compensation strategies [FGIZ08, FZGI10], in particular.

Moreover, since the NEXOF [NEXOFRA10] reference architecture shall constitute an overall baseline for CHOReOS, in the case of choreography adaptation we will specifically built upon the NEXOF management concern that focuses on the main management and configuration activities that should be provided by a platform that supports the development of service-oriented systems. More specifically, according to the NEXOF management concern, a service-oriented system is initially configured with respect to a service-level agreement. The agreement is specified and translated in a machine-interpretable form. Then, it is negotiated with service providers. Following, the system is monitored with respect to the specified service-level agreement. The monitoring may trigger the need for re-negotiations, which will be followed by the reconfiguration/adaptation of the system.

The NEXOF management concern is basically in line with a generic adaptation process [ZFGV06] that we previously proposed and successfully applied in the cases of pervasive and service-oriented systems. This generic adaptation process shall also serve as a baseline for the CHOReOS choreography adaptation approach.

4.5.3. Progress

As previously discussed in detail, the majority of the state of the art approaches on the adaptation of service-oriented systems focus on orchestrations of services. On the other hand, CHOReOS goes beyond the state of the art by considering the adaptation of choreographies in the context of the Future Internet. In this context, the adaptation problem becomes much more complex.

Substituting a target service for another one in the case of an orchestration amounts to finding a substitute service that satisfies the orchestration requirements. The substitution itself affects a number of orchestration tasks that rely on the target service. These tasks can be easily located and adapted so as to use an adaptor that accesses the substitute service, instead of using the target service.

On the other hand, *service substitution in a choreography*, involves finding a substitute service that satisfies the requirements of all other services that use the target service. In that sense, in the case of a choreography, the substitution of the target service may trigger the *re-composition* of the choreography. Then, the first issue is to discover the affected services. The second issue is to deal with the possibly conflicting requirements of the affected services. Dealing with the first issue amounts to *performing impact analysis, dynamically and efficiently*, which is not obvious, considering the large scale of the Future Internet choreographies. Similarly, dealing with the second issue is also not straightforward, since it may require *re-negotiations between a possibly large number of services*. Finally, if it is not possible to substitute the target service with another one, it may be necessary to propagate the service substitution request to the services that are actually used by the target service, so as to increase the probability of actually adapting the choreography.

Hence, the adaptation of Future Internet large-scale choreographies calls for revised adaptation strategies. CHOReOS shall provide such revised adaptation strategies in the form a middleware that will extend/refine the key concepts of the NEXOF management concern. Of course, the adaptation middleware shall rely on the functional and non-functional

abstractions of the CHOReOS architectural style and the CHOReOS large-scale service base that will support the overall CHOReOS dynamic development process.

5. WP4: Governance and V&V Support for Choreographies for the Future Internet

The CHOReOS work package WP4 investigates strategies and mechanisms necessary to establish and to exercise governance in a wide heterogeneous inter-organization setting where integration is defined *via* large scale service choreographies (see Section 5.1). Special attention is further devoted to policies and rules related to V&V activities, which are challenged by the target ULS Future Internet (see Section 5.2). Furthermore, Governance and V&V Support for Choreographies consider the functionalities described in the “Management” and “Composition” top-level concerns of the NEXOF Reference Model [NEXOFRM] (upon which the CHOReOS Conceptual Model will be based as introduced in Chapter 2). These concerns will be refined and enriched in CHOReOS by also considering FI improvements derived by this work package investigation.

5.1. Governance

Since its inception, SOA was revealed as being the *de facto* paradigm for future systems. It combines best practices inspired from previous application models. Modularity, encapsulation, fine-grained granularity, publication, and discovery help SOA to be widely used by developers and users. As a consequence, enterprise systems have been moving to this new trend.

As systems are moving from classical IT to innovative SOA, essential functions need also to be exported and adapted. Governance is a first and foremost function in IT systems. It ensures the best interests of an organization to be met through corporate decisions from strategy to execution [Mar08]. The authors of [WR04] give a definition that goes in the same direction; according to them, IT governance is “*specifying the decision rights and accountability framework to encourage desirable behaviour in the use of IT.*” IT managers are then concerned with decisions, processes, and policies to encourage the behaviour that contributes to success. It can even go further including leadership and organizational structures and processes that ensure that the organization’s IT sustains and extends the organization’s strategies and objectives [ACHC+07]. In [BMT06], the authors give a global definition for governance as:

- Establishing chains of responsibility, authority, and communication to empower people (decision rights);
- Establishing measurement, policy, and control mechanisms to enable people to carry out their roles and responsibilities.

5.1.1. State of the art

SOA governance extends IT governance for the purpose of ensuring the SOA success. Lack of governance can be a serious impediment to success and the most common reason for the failure of SOA projects [ACHC+07]. In [PTDL07], the authors identify SOA governance as a major research area in the field of SOA design and development. Nevertheless, SOA governance is not clearly defined in the literature. We rely on the following definitions to identify key topics in SOA governance:

- In [ACHC+07], Oracle researchers define SOA governance as the interaction between policies (what), decision-makers (who), and processes (how) in order to ensure SOA success. SOA governance is able to ensure that all of the independent (SOA) efforts come together to meet enterprise requirements. It covers the following levels: design, development, deployment, and operations of a service.
- In [BMT06], IBM researchers assume SOA governance to be an extension of IT governance specifically focused on the lifecycle of services, metadata and composite

applications in an organization's SOA. SOA governance extends IT governance by assigning decision rights, policies and measures around the services, processes and lifecycle of SOA to address such concerns as: service registration, versioning, ownership, funding, monitoring, auditing, publishing, discovery, etc.

- In [Mar08], the author aggregates several definitions to give his own: SOA governance is the definition, implementation and ongoing execution of a SOA stakeholder decision model, and accountability framework that ensures an organization is pursuing an appropriate SOA strategy aligned with business goals, and is executing that strategy in accordance with guidelines and constraints defined by a body of SOA principles and policies. SOA policies are enforced *via* a policy enforcement model, which is realized in the forms of various policy enforcement mechanisms such as governance boards and committees; governance processes, checkpoints, and reviews; and governance enabling technology and tools. Through the previous definitions the essence of SOA governance is revealed in the trilogy (decision, process and policy):

SOA governance is doing the right SOA things (**processes, decisions**) the right way (**policies**) for the SOA stakeholders (**decision-makers**) [Mar08]

SOA governance needs to be achieved at several levels, from strategy to execution. Governance tools provide the functionality required to support the governance processes associated with a specific SOA initiative including the following: SOA policy management and enforcement; Registry/repository and metadata management; Statistical and Key Performance Indicators data collection; Governance of services in the cloud; Monitoring and management; Application and service life cycle management; and Interoperability with other SOA governance technologies. The following surveys approaches to SOA governance, considering both academic and industrial solutions. Still, examples from industry are more numerous as SOA governance methodologies are mostly driven by SOA vendors.

Authors of [DW07] address SOA governance by proposing a generic model and two governance tools. Services are described according to their life cycle, and activities and roles, which are relevant during the service life cycle, are considered. Three roles are identified: the service developer, the product manager and the administrator. The product manager determines customer requirements, specifies a service for the business logic needed and is responsible for associating the service with a product. The developer is responsible for implementing the service and decides how the service is structured on a technical level. Meanwhile, the generic model contains elements describing clients and service proposals. It also contains elements for products and service modules. Based on the previous model, the authors propose two tools: a service repository console and a service browser. The Service Repository Console is used for creating service proposals and service descriptions, for specifying service relationships, and for defining service installations. The Service Browser is used for searching and browsing the service repository and for investigating service details, service relationships and service status. This approach proposes a standard way and a solid model to consider SOA governance. Nevertheless, the authors do not address governing practices for neither orchestration nor choreography of services. Neither do they consider large scale system requirements.

A methodological approach to SOA governance is presented in [SIV08]. The authors claim that SOA governance is more than a process, it is all about continuously aligning strategic goals. The authors thus define a six-step governance life cycle: (1) defining a SOA strategy to align SOA with business requirements; (2) aligning organization to SOA by assigning responsibilities and establishing project groups; (3) managing service portfolio to ensure that a sound method is used consistently to decide which services need to be developed; (4)

controlling service lifecycle that is concerned with the development and delivery of individual services in a SOA (service granularity and consistency, management procedures, etc.); (5) incorporating policy enforcement to perform service checks to verify it complies with policies; and (6) service level management to specify the contract stating services levels and possible fees. This should be specified for each service. In [SIV08], the authors give clear guidelines to achieve SOA governance; however no detail is given for describing practical governance prototype.

The IBM approach to SOA governance is presented in [BMT06] as a four-step life cycle. It consists of: (1) a planning phase during which the need for governance is established and the existing mechanisms are assessed; (2) a definition phase during which the desired governance framework, including new and modified principles, processes, organizational structures and roles are established; (3) an enabling phase in which the new governance framework is introduced into the enterprise; and finally, (4) a measurement phase during which the metrics are gathered and analyzed to refine the governance process. Relying on this framework, IBM proposes the IBM WebSphere Service Registry and Repository [WebSphere], which offer a governance solution in IBM SOA; it supports service discovery and access. Besides, it offers features for service metadata management. Advanced management capabilities are provided by IBM Tivoli product [Tivoli].

Oracle proposes a framework and a six-step based solution to SOA governance. Relying on a previously stated SOA maturity model, the authors in [ACHC+07] give a roadmap for SOA governance. Key points for policies are captured according to business areas; these are: architecture, technology infrastructure, information, finance, portfolios, people, projects and operations. Then, policies need to be designed and enacted across the cited areas. Depending on the area, policies model and medium may be different; some policies can be captured in technology solutions or simply in policy documents. For instance, operational policies such as governing services at runtime may be addressed through the adoption of a technological solution as a registry/repository or a Web service management, whereas, architectural and funding policies can be captured through documents distributed through the organization. Meanwhile, the authors give six generic steps and best practices in order to apply and benefit from SOA governance; these are: (1) Defining goals, strategies, and constraints, (2) Defining standards, policies, and procedures for financial, portfolio, project, services, (3) Defining metrics for success, (4) Putting Governance mechanisms in place, (5) Analyzing and Improving existing processes, and (6) Refining and going to the next level of SOA maturity. Based on the above framework, Oracle proposes a proprietary product Oracle SOA Governance [Registry] [Repository] and the recently acquired Amberpoint, consisting of an enterprise repository, a service Registry, an enterprise Manager, and a Web service manager. Both Oracle and IBM are leading companies in this domain and their products cover most basic governance features.

Mule Galaxy [Galaxy] is an open source SOA governance platform. It provides a SOA registry/repository. Galaxy aids in the management of SOA by supporting features such as lifecycle, dependency and artifact management, service discovery and reporting, and application deployment management. Although Mule ESB Enterprise includes a service registry/repository that assists in artifact management and publishing, its policy enforcement capabilities (particularly, the implementation and modification of life cycle management processes) fall short of similar capabilities in closed-source offerings.

WSO2 [WSO2] is an open source WS-based SOA governance registry. It is standardized and supports basic SOA governance and integration capabilities such as tracking SOA resources, managing services life cycles and controlling resource access. Moreover, the WSO2 registry provides a repository where resources and collections can be stored and managed, tagged, rated, logged, etc. Besides, SOA governance tools support common schema validation policies such as WSDL validation, Web service discovery, lifecycle

management, dependency relationship management and remote link support. Finally, WSO2 provides advanced features such as ATOM protocol support, and local and remote registry management. Though WSO2 is seen as a visionary governance tool it may be improved by implementing a scalable subscription model and supporting some basic mechanisms dedicated to business applications.

The Petals Master SOA Governance Solution [Master] is an open-source governance tool that is UDDI-based, providing basic governance features such as service registry/repository, organization management, Service Level Agreement management and Integration with Service Runtime Environment. The registry/repository also provides management of policies that govern the behavior of users (persons or systems), dependency management (between services and other SOA assets lifecycle management), and reporting (usage indicators, policy violations, etc.). Petals Master can be deployed as a standalone tool or as integrated in the service bus Petals ESB. Although it offers common governance functionalities and SLA and WS-agreement support, Petals Master needs to be improved by enforcing policies and providing scale-up capabilities in order to suit large distributed systems. The involvement of Petals Link (the EBM partner of CHOReOS) in research projects gives an opportunity to implement new innovative features in Petals Master.

There are some other commercial products, such as Progress's SOA tool CentraSite [CentraSite], Hitachi's product Cosminexus [Cosminexus] or HP Governance Interoperability Framework [HP-GIF], etc. They typically provide service discovery, dependency management, policy management, change notification, authentication and identity management, policy management, and federation with other repositories. An ideal SOA governance tool should benefit from the best practices from all the existing tools.

5.1.2. Baseline

As discussed in the previous section, the literature around SOA governance gives a very wide definition of governance. Main software vendors such as IBM or Oracle, involve a plethora of their products under the term governance. In order to have a clear approach to SOA governance, we present in the table below the main stakeholders. By answering the *What? How? Who? and When?* Questions, we aim at establishing a clear roadmap of what is intended for governance.

What? <ul style="list-style-type: none"> • Services • Components • Registry/repository • Orchestration • Choreography • Performance indicators • Measurement 	How? <ul style="list-style-type: none"> • Life cycle definition • Logs • Impact Analysis • Monitoring, management • Policies • Decisions • Conformance
Who? <ul style="list-style-type: none"> • Developer • Service consumer • Service provider • Service composer 	When? <ul style="list-style-type: none"> • Design time • Development time • Deployment time • Runtime

Current SOA governance tools cover the main needed features and answer the above questions; nevertheless, some issues need to be pointed out. First, there is a lack of well defined and robust formal governance reference Architecture, mostly because SOA

paradigms are quite new to the IT community. Second, products in the SOA governance space are different and implemented in different ways and with various features, and there is thus no common implementation protocol for governance tools. Third, policies that are a corner stone of governance tools are not well defined in a common standard. Finally, when considering large scale choreographies of interacting services, governance must also consider appropriate means to facilitate and support on-line verification and validation.


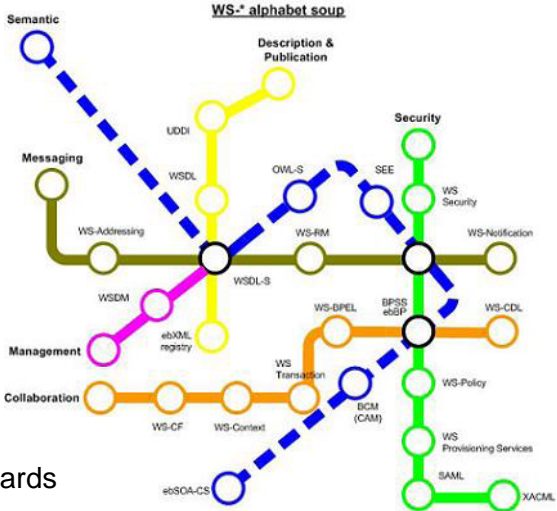

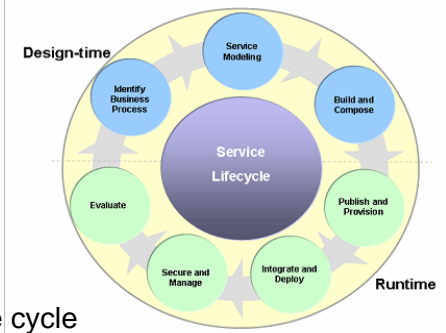
Petals Master is an open source governance tool providing service indexing and documentation features such as looking for a service in different ways either by using its name, category, operation, etc., or in a dynamic way using UDDI requests and Web services. Moreover, it handles WSDL descriptions and allows collaborative annotations and tags. Petals Master can be run as a standalone tool or can be integrated on the Petals bus. When plugged in the bus, it is able to synchronize with services and to search for running ones on the different Petals ESB nodes. Finally, for more convenience Petals Master offers an ergonomic graphical user interface and a UDDI interface to allow the tool to be plugged into the Eclipse environment.

Petals Master implements the CBDI-SAE Meta Model for SOA. CBDI is a reference model giving guidelines for SOA including taxonomy, classification, and policies. It proposes Meta models for technology, organization, service, policy, business modelling, specification, implementation, runtime, and deployment. The CBDI Meta model makes Petals Master a good starting point, built on a well-defined reference model and able to be extended to cover SLA contracts, management policies, and ultra large scale systems requirements.

5.1.3. Progress

SOA governance is needed all along the service life cycle. Thus, proposing a governance tool lies in the appropriate definition of indicators, policies, and assessments for the overall lifecycle, from design- to run-time, with a special concern for governance supporting Verification&Validation (whose approaches are discussed in the next section).

We plan to follow a governance plan centred on the four questions that we previously highlighted, i.e., What?, How?, Who?, and When?. The following table depicts our choices for each question, as further discussed below.

<p>What?</p>  <p>Choreography</p> <p>Services</p> <p>Orchestrations</p>	<p>How?</p>  <p>WS-* alphabet soup</p> <p>WS-* Standards</p>
<p>Who?</p>  <p>Developer</p> <p>Architect</p> <p>External User</p> <p>Analyst</p> <p>Network</p>	<p>When?</p>  <p>Life cycle</p>

- “What?”: answering this question refers to stating the elements to be governed. ULS orchestrations and choreographies of services require special care since they are deployed in widely spread architectures and involve a very large number of participants, services and resources. Moreover, both functional and QoS properties need to be considered. In effect, we will have to define the observed elements and their granularity.
- “How?”: addressing this question lies in having clear guidelines for defining and adopting the right policies and involving the right stakeholders in the right steps. In [ULS06], the authors claim that given the scope and scale of envisioned ULS systems, technical, organizational, and operational policies will emerge as principal vehicles for ensuring harmonious operations at all levels. As a result, conformance to policies will become the price of entry for participation in ULS systems, and fast assessment of policy conformance will become critically important. Thus, in CHOReOS, we need to process policies without limiting the openness of Future Internet systems. In order to achieve such a goal, we will build upon the guidelines of [ULS 2006]:
 - First, defining ULS system policies for flexible collaboration will have to reconcile diverse, and often incompatible and poorly articulated objectives. Conflict resolution, consensus-like and democracy inspired mechanisms may be considered.

- Second, policy mechanisms need to be automated in order to cope with ULS systems requirements. Both local and decentralized governance processes may be triggered automatically in order to perform fast and effective actions.

From a technological point of view, we are firmly convinced that considering technological standards such as WS-* standards, in software development gives us a solid and common reference model and supports interoperability and evolution of software architectures [Thi06]. With regards to the CHOReOS conceptual model that will be devised in WP1, we will introduce governance capabilities. As a relevant background, the NEXOF-RA project has studied the SOA governance issue [NEXOFRM]. Actually, project management and monitoring technologies are needed to ensure services behaviour control during services lifecycle, from development to deployment to runtime. Moreover, services, resources and processes should provide a management interface in order to be controlled. The management capabilities offered by such interfaces include operations and actions supporting the services and processes life cycles (deploy/undeploy/star/stop) and the access configurations (access points, names, etc.). The CHOReOS project will extend the governance framework provided by the NEXOF-RA project by including choreography governance and relevant policies according to the specific CHOReOS requirements.

In the current state, Petals Master implements the UDDI and WSDL standards for service description and publication. Furthermore, we plan to improve Petals Master by studying standards such as WS-Policy and WS-security for handling security concerns and WSDM for including choreography monitoring. We will extend a previous work led in the SOA4All European research project, which deals with semantic extensions for the WS-Policy standard called SA-Policy (Semantic Annotation Policy). This step defines the needed policies and rules enabling governance

- “Who?”: answering this question lies in identifying the persons concerned with roles and responsibilities. Developers, system administrators and services composers are more likely to take on governance roles. This step is dedicated to state which persons are concerned with the governance process and which responsibilities are assigned to them. It is a very important task during a governance process, as it helps giving freedoms and boundaries for each involved person. Responsibilities can be given according to technical, organizational or operational participant involvement, and they can be legal, economical, contractual, design or security rules. This step will help identifying main stakeholders and their action boundaries.
- “When?”: SOA governance is needed during the overall service life cycle. Then, focus will be primarily put on identifying the most relevant steps of the service life cycle from the standpoint of governance. Then, control indicators and assessment mechanisms need to be identified according to each step of the life cycle. Special attention will be paid to the choreography governance from composition to runtime, in order to be able to track behaviour, performance and to easily point out discordances. Policies need to be classified into categories, such as, for example, schema policies, communication policies and behaviour policies. These can then be applied differently according to the specific steps of the service life cycle. Moreover, ULS systems are geographically large and distributed over a federation of networks, requiring the governance tool to be adapted to this specific characteristic. Both centralized and decentralized governance mechanisms need to be adopted in CHOReOS since in some cases, local governance processes are required while in other cases, a more global view is needed. For instance, authorization rules may be required locally for the system users, whereas, security process may be run in a decentralized manner. Finally, answering the *When* and *Where?* Question shall lead to identify the governance scope.

5.2. Verification & Validation

The dynamic development process for ULS choreographies envisaged in WP2 must also include appropriate approaches and tools for Verification and Validation (V&V). To address V&V in CHOReOS, WP4 will target new model-based testing approaches, to be applied both off-line and on-line, relying on the devised Governance framework. V&V of SOAs can be considered a young investigation domain. In 2005, a EU note stated that “the area of testing services and their specifications has until now received limited attention from the research community” [SM05]. Looking at the literature of the elapsed five years, we can observe that the situation has not improved that much since. Certainly this lack of attention is not due to the fact that applicable solutions are already available and research in this new domain is not needed. On the contrary, many issues need to be solved. V&V of ULS choreography based systems is hindered by several factors, among which run-time discovery and binding and high distribution of software elements. In particular, distribution in ULS choreographies is not only evident in terms of space but also in terms of control, since different interacting services are typically owned, deployed, and controlled by different organizations.

5.2.1. State of the art

Research activities need to follow several routes. A first route concerns the support for test case codification and execution. In such a context, automated tests have been proven to be a useful mechanism for detecting and reducing defects [WMV03] and for increasing development speed [Bec02]. There are many automated test frameworks like the xUnit tools (e.g., JUnit, C#Unit) [Mes07] that can be used for testing systems compliant with the client-server model. A similar approach is, for instance, implemented in the commercial tool SoapUI [SOAPUI], which allows developers to build SOAP envelopes automatically, directly from WSDL specifications, to mock and to deploy mocked services, and to derive simple test cases for validation of service functionalities. Even though this tool could be considered as part of the xUnit tool family, it requires a high amount of human intervention to finalize the test suites. With respect to testing orchestration specification, in [ML06], BPELunit is illustrated. With respect to distributed systems we can mention BizUnit [BizUnit], a flexible and extensible declarative test framework that enables the automated testing of distributed systems. In conclusion, in the context of large-scale distributed systems, there is no mature approach with respect to the xUnit tool family.

With respect to choreography testing and development, we can cite the Savara project [SAVARA] by Red Hat and Cognizant Technology Solutions. The Savara Project proposes a methodology for SOA application development. This methodology is called “Testable Architecture” and its goal is to ensure that any artifact (e.g., models, test cases, code) defined in the development lifecycle will be validated against the ones defined before and after each phase. One of its internal components, called PI4SOA tool, allows developers to monitor choreographies by means of simulation. To achieve that, Web services and communication are not real, but messages exchanged among the partners and their roles can be checked using this off-line approach. Therefore in this process, a black-box testing technique is performed to validate the global model using a simulated testing approach.

Another interesting investigation domain is the automatic generation of test cases from data models to test service and service compositions. Several approaches to automatic test case derivation take advantage from the availability of computer readable format such as interface descriptions (WSDL) and data (XML Schemas). For instance, in [BDTC05], data dependencies among the provided operations in the interface are used to automatically derive test cases satisfying the expressed dependency. A random strategy is instead followed in [SH07] to generate invocations directly from a WSDL description and from a set of defined constraints (assertions) on the input data. In [LZZM09], a tool using the same information to drive the selection of data and sequences to manually define test generation rules, selected within possible choices automatically defined, is described.

In order to automatically derive test cases taking into account behavioral specification, Model Based Testing (MBT) techniques have been introduced. MBT refers to a general approach to derive test cases from the exploitation of a formal model of the service under test. In the context of SOA, some works in this direction try to derive test cases from choreography specifications, applying algorithms defined for conformance checking [FTV06], or using a model checker by looking for counter-example executions to defined properties (test purposes) when the service composition model is translated in a format suitable for being processed by a model checker [CDPP08]. Many different MBT approaches can be found in literature, each one trying to apply test case derivation strategies to different formal specification, such as EFSM based format [LZCH08], Petri Nets [DY06] or Control-Flow graphs [YLS06], which are used to describe service behavior or their compositions.

In [BP09], we discuss the importance that governance assumes within the service-oriented computing domain, which is in particular even more relevant within a ULS context. As a result, software engineering activities seem to require a collaborative approach. This is certainly true also with reference to testing activities where the different stakeholders participate to the testing effort performing different tasks. In this area, we can list approaches proposing to enhance the registry service with testing mechanisms, resulting in a registration granted only to those services passing the testing phase [TPCY+03, BP05, HM05].

In [BWDT+07], a similar idea is exploited. In particular, a Decentralized Collaborative Validation and Verification (DCV&V) framework with contracts is proposed. The framework consists of distributed test brokers that handle a specific part of the testing process. Two types of contracts for the DCV&V approach are considered: Test Collaboration Contracts (TCC) that enforce the collaboration among the test brokers and Testing Service Contract (TSC) that is used for contract-based test case generation.

Another interesting collaborative approach is proposed in [ZL10] to derive reliability measures on the base of past failure data of other similar users to predict the Web service reliability for the current user, without requiring real-world Web service invocations.

The idea of moving V&V activities to run-time has been proposed by mainly introducing monitoring activities both for functional and non-functional properties. In such a context, interesting work is [BGKP08], where the authors propose to augment orchestration specification with run-time checks to verify the behavior of interacting participants, and [RSE08] where the authors derive run-time monitor for temporal QoS properties directly from temporal specifications of interacting services. In [BGP08], the authors propose a unified supervision framework for BPEL processes that integrates different monitoring and recovery techniques, and builds upon the decoupling of data collection, monitoring and recovery. In [BGNS09], a general high-level service-centric monitoring specification language is proposed, created as an extension of WS-Agreement, a specification for defining and managing SLAs for Web services. A concrete projection of the model onto three different run-time monitoring frameworks is presented. In [BBGS09], the issue of efficiency in the run-time verification of service compositions is addressed, by proposing an operational semantics for an existing assertion language for the specification of functional and non-functional temporal properties of BPEL processes. Such semantics is based on the correlation between temporal logic and a class of alternating automata.

The above listed approaches do not solve all the V&V issues raised by ULS choreographies. Most notably, since in such context application, (service) integration is increasingly distributed and dynamic, i.e., unpredictable at development time, CHOReOS idea is to make V&V activities themselves more dynamic and move part of them towards runtime. In particular, CHOReOS intends to develop methodologies and techniques enabling the execution of the different checking and testing activities on-line while all the cooperating services (or part of them) are running in the final environment. The idea started to emerge in recent works (i.e., [CFCB10]) but many issues still need to be solved. In particular, available

techniques need to be reconsidered and adapted in order to include information on the current real environment in the derivation and execution of V&V activities. Besides, such type of activities can only be performed as a collaborative approach between the involved stakeholders, hence they must be rooted on the governance policies and supporting infrastructure.

5.2.2. Baseline

Within CHOReOS, the investigations in the area of V&V will lay on previous extensive experience of the CNR, USP, and CEFRIEL partners in testing services and choreographies. In particular, these partners have developed relevant and complementary experience on the different aspects of service V&V, through the participation to several European projects, including the FP6 PLASTIC project [BBDF+08], the FP7 TAS3 project [TAS3], the FP6 SUPER project [SUPER], the FP6 SeCSE project [SeCSE], and the FP7 SOA4ALL project [SOA4All]. Such large background provides a comprehensive and representative baseline that suitably embraces the results above described.

Within PLASTIC, CNR exploited for service validation the new opportunities that the SOA paradigm makes available. In particular, it has been observed that within SOA: (i) there is a wider and public availability of formal service integration specifications, such as for instance orchestration or choreography specifications, (ii) there is a wide availability of computer readable formats specifying different aspects of single services and of their interactions, and finally (iii) there are new points of controls to which testing activities could be associated. Starting from these considerations, CNR developed some novel approaches for service testing. In particular, CNR investigated the possible usage of formal specifications and the application of MBT techniques for the derivation of test beds to assess services before their final deployment, both from a functional and a non-functional point of view [BDFP08]. MBT is a widely investigated subject, in particular in the area of protocol testing. In CHOReOS, we will identify the strategies that best suit our scope, also in consideration of the ULS choreography conceptual model definition that CHOReOS will define in WP1, and related deliverable D1.2. MBT approaches seem particularly appealing as they permit the dynamic and automatic derivation of test cases taking into account information that could emerge only at run-time. At the same time, it will be particularly important to investigate techniques enabling the fast derivation of small and powerful test suites to be executed at run-time, as it is not reasonable to assume that testing sessions will elapse for a long period.

Moreover, thanks to the availability of computer readable formats for service interfaces and for the data that can be exchanged through such interfaces, CNR has defined an approach permitting to automatically derive test suites for developed services just starting from the WSDL description of the service itself [BBMP09]. This solution applies a partition-based approach on the data set used to interact with the service. CNR has defined a testing framework in which a UDDI-based service registry can be augmented with testing functionality so as to guarantee that registered services have been successfully evaluated by a third party [BP05], and appropriate references can be provided. In CHOReOS, we will start from such results and will generalize the developed approaches to the specific exigencies of ULS choreographies and to the discovery services that will be available in the CHOReOS infrastructure. The problem of monitoring functional and QoS characteristics have been studied in [RSE08], where timed automata models were considered. Developed monitoring mechanisms will take inspiration from that developed in [RSE08]. Finally, we will extend the on-line testing infrastructure conceived within the TAS3 EU project [BDPS10]. Such an infrastructure permits to make testing invocations also when authentication and access policies mechanisms are in place.

CHOReOS researchers at USP have been working with Test-Driven Development (TDD) [Bec02] in practice for five years and empirical evaluations of the benefits of TDD are being carried out. However, TDD is normally applied to the development of centralized or client-

server systems. Thus, within the CHOReOS project, our goal will be to investigate how to transport the major elements of TDD to the development of choreographies, studying the impact and adaptations to the methodology that the distributed and large-scale nature of the Future Internet will impose. To be able to execute the automated tests in a reproducible way, during development time, we will build a deployment and execution framework for testing based on the USP previous research on deploying and executing distributed systems for debugging purposes [MK07].

The FP6 SUPER project [SUPER] investigated approaches that aim at monitoring, diagnosing, simulating and mining enacted processes, in order to support the analysis and enhancement of process models. Such approaches are based on Semantic Business Process Management (SBPM). As described in [CKDZ+07], semantic technologies are introduced in the process life-cycle to link the data necessary for the analysis with defined ontological concepts. SBPM provides machine processable semantics that could support business practitioners in the analysis and re-engineering of processes and executable processes definition [PD07] as it will be investigated within CHOReOS.

Negotiation of service contractual agreements and service V&V represent two of the main topics investigated also by the FP6 project SeCSE [SeCSE]. In particular, according to the SeCSE approach, automated SLA negotiation can be used both at design time and at runtime to support discovery and selection of services (e.g., to determine the best candidates for the binding among the set of services found through discovery). Negotiation can be used as an alternative to global rebinding as a recovery action in the event of risk of a global QoS violation, as reported by service monitoring components, i.e., data collectors evaluating both functional and QoS properties [MITbook].

Finally, the FP7 project SOA4All [SOA4All] investigated self-adaptability of service compositions based on context information used to provide an Adaptive Service Composition Execution able to adapt service requests to actual service interfaces at the execution time, to support the service selection according to functional, non-functional and contextual information, and to provide fault handling and dynamic rebinding mechanisms. In other words, the project conceived mechanisms to enable dynamic replacement of services in a service composition at runtime. Such replacement relies on service descriptions extended by means of proper semantic annotations (SAWSDL-based in the case of WSDL services, MicroWSMO based in the case of REST services), and on the monitoring of the execution context. Based on such information, compatibility between services can be automatically evaluated at runtime in order to find the best replacement for a failing service, even in presence of syntactical interface mismatches.

5.2.3. Progress

Run-time V&V presents many challenges, in particular when involved services can refer to stateful resources. For such a reason, proposed approaches to V&V of services typically just refer to strategies to be applied before the final deployment of the service. As already said, for ULS choreographies, off-line V&V approaches are not anymore sufficient, and in CHOReOS we will investigate the opportunity of moving V&V activities also to the run-time phase. To do this, we recognize the necessity of investigating in parallel the relevance of governance as an enabling factor of run-time V&V activities. CHOReOS aims at V&V methods and tools that rely on the introduced governance framework, towards performing dependability analysis of choreography-based software efficiently at runtime.

With respect to the NEXOF Reference Model [NEXOFRM], the activities investigated in the CHOReOS approach to V&V mainly relate to the following functionalities:

- *SLA Management*: this functionality (belonging to the “Management” concern) allows managing both functional and non-functional requirements and capabilities of services,

so as to support the description, creation, negotiation and monitoring of SLAs, as well as the definition of related metrics.

- *Monitoring*: this functionality (belonging to the “Management” concern) deals with the specification of monitoring parameters and monitoring policies and the collection of information (from both resources and services) for supervision, analysis and decision making purposes.
- *Testing*: this functionality (belonging to the “Composition” concern) tackles the testing activities that should be performed during the implementation of service components. Currently envisioned to support service composition only, in CHOReOS, this aspect can be investigated also from a choreography viewpoint.

The result of the work that will be carried out within WP4 will be a comprehensive V&V strategy for services within ULS choreographies, providing support for all the phases of the service life-cycle. To the best of our knowledge, there is no infrastructure foreseeing the inclusion of supports for run-time checking of services, in particular using MBT techniques. We will derive specific service evaluation supports, both before service deployment and during run-time, also conceiving methodologies and techniques to augment models with run-time information using such augmented models for test case derivation purpose. The CHOReOS service interaction and testing infrastructure will be also equipped with mechanisms permitting to perform test cases at run-time even when authentication and access policies are in place. The evaluation will not only consider functional properties but will take into account QoS characteristics. To handle the ULS and its continuous evolution, the proposed methods and tools must be online and compositional. Furthermore, the proposed methods and tools shall be able to consider, for the same choreography, different levels of QoS for different (groups of) users. Addressing ULS choreographies will require investigating lightweight mechanisms that nevertheless still permit to detect and forecast erroneous behavior. Moreover, the framework should also incorporate sampling techniques to deal with the very many interacting services giving rise to unmanageable numbers of potential configurations. As a result, CHOReOS will then provide a highly valuable support for service development evaluation and for service run-time evaluation; these will exploit, and interact with, the service-oriented middleware developed in WP3.

At the same time, one of our goals is to develop a TDD (Test-Driven Development) methodology that will help developers and project leaders to deal with the key issues involved in testing large-scale, ULS systems and will guide them in the production of effective and efficient test suites. To achieve these goals, we first intend to develop an open source environment to support the methodology proposed. This environment will include: (1) the definition or adaptation of a simple language for specifying the deployment of a distributed application (and its associated choreographies) across the network in a reproducible way, (2) the construction of a tool for parsing specifications written in this language and deploying the application, and (3) the development of a framework for writing and executing the tests in the distributed system at (testing) runtime.

The proposed framework and its associated methodology will enable the production of higher-quality large-scale Internet systems, serving as a solid basis for the development of dependable and robust large-scale choreographies. The software tools will be developed incrementally and each release will then be used and analyzed by external users, producing feedback that will be used in refining the tools and the methodology. The methodology and the tools will also be evaluated *via* controlled experiments to assess their performance and effectiveness.

Furthermore, the CHOReOS project will investigate if it is profitable to extend the work carried on in the SUPER, SeCSE, and SOA4All projects in order to include special purpose information, tools and techniques for enhancing some governance and V&V activities to make them suitable for ULS choreography-based scenarios. In particular, the SeCSE

approach to V&V and SLA negotiation can be further investigated so as to evaluate its applicability to ULS choreographies, which requires to rethink different aspects, e.g., content of the 'testing facet' of a choreographed service, monitoring of SLA/QoS violation in the context of decentralized execution of services, automation of V&V tasks (including regression testing) etc. The SOA4All approach to adaptive service composition can provide a starting point to design an augmented execution environment, which is able to provide monitoring information about choreographed services at runtime, also in the open world setting of the ULS Future Internet, and to react to contextual changes of choreographies being executed. Aspects to be investigated in this case include: annotation of service description (e.g., semantic based), mapping languages, automation of some V&V tasks.

6. References

- [ABCD+07] M. Autili, L. Berardinelli, V. Cortellessa, A. Di Marco, D. Di Ruscio, P. Inverardi, and M. Tivoli. "A Development Process for Self-adapting Service Oriented Applications". ICSSOC 2007, LNCS 4749, pp. 442–448, Springer-Verlag, 2007.
- [ACHC+07] M. Afshar, M. Cincinatus, D. Hynes, K. Clugage and V. Patwardhan, SOA Governance – Framework and Best Practices, White Paper, Oracle Corporation May 2007, Version 1.1
- [ACMM+09] F. Arbab, T. Chothia, R. Mei, S. Meng, Y. Moon, and C. Verhoef. From Coordination to Stochastic Models of QoS. In Proceedings of COORDINATION, 2009.
- [ACMM07] F. Arbab, T. Chothia, S. Meng, and Y.-J. Moon. Component Connectors with QoS Guarantees. In Proceedings of COORDINATION, 2007.
- [ACMP+07] D. Ardagna, M. Comuzzi, E. Mussi, B. Pernici, and P. Plebani. PAWS: A Framework for Executing Adaptive Web-Service Processes. IEEE Software, 24(6), 39–46, 2007.
- [AFGJ+10] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. Katz, A. Konwinski, G. N. Lee, Patterson, A. Rabkin, I. Stoica, and M. Zaharia. 2010. A View of Cloud Computing. Commun. ACM 53(4), 50-58, 2010.
- [AH01] L. de Alfaro and T.A. Henzinger. Interface automata. In Proceedings of the Joint 8th European Software Engineering Conference and 9th ACM SIGSOFT International Symposium on the Foundations of Software Engineering, 2001.
- [AHS01] E. Aarts, R. Harwig, M. Schuurmans. Ambient intelligence, in: The Invisible Future: The Seamless Integration of Technology into Everyday Life, McGraw-Hill, 2001, pp. 235–250 (Chapter).
- [AINT07] M. Autili, P. Inverardi, A. Navarra and M. Tivoli. SYNTHESIS: a tool for automatically assembling correct and distributed component-based systems. In Proc. of ICSE, 2007.
- [AJ10] V. Agarwal and P. Jalote. From Specification to Adaptation: An Integrated QoS-driven Approach for Dynamic Adaptation of Web Service Compositions. In Proceedings of the IEEE International Conference on Web Services (ICWS), 2010.
- [AM08] E. Al-Masri, Q. H. Mahmoud. Investigating web services on the world wide web. Proceedings of the 17th International Conference on World Wide Web, WWW 2008, pp. 795-804, Beijing, China, April 21-25, 2008.
- [AMP94] E. Asarin, O. Maler, and A. Pnueli. Symbolic Controller Synthesis for Discrete and Timed Systems. Hybrid Systems, 1994.
- [AP07] D. Ardagna and B. Pernici. Adaptive Service Composition in Flexible Processes. IEEE Transactions on Software Engineering 33(6), 369-384, 2007.
- [AW97] M. F. Arlitt and C. L. Williamson. Internet web servers: workload characterization and performance implications. IEEE/ACM Trans. Netw., 5(5):631–645, 1997.
- [AZI09] D. Athanasopoulos, A. Zarras and V. Issarny. Service Substitution Revisited. In Proceedings of the 24th IEEE/ACM International Conference on Automated Software Engineering (ASE), 2009.
- [BBBE+07] M. Beisiegel, H. Blohm, D. Booz, M. Edwards, O. Hurley, S. Ielceanu, A. Miller, A. Karmarkar, A. Malhotra, J. Marino, M. Nally, E. Newcomer, S. Patil, G. Pavlik, M. Raepple, M. Rowley, K. Tam, S. Vorthmann, P. Walker, L. Waterman. SCA Service Component Architecture - Assembly Model Specification. Open Service Oriented Architecture www.osoa.org/download/attachments/35/SCA_Assembly_Model_V100.pdf?version=1, 2007

- [BBDF+08] A. Bertolino, D. Bianculli, G. De Angelis, L. Frantzen, Z.G. Kiss, C. Ghezzi, A. Polini, F. Raimondi, A. Sabetta, G. Toffetti Carughi, and A. Wolf. Test Framework: Assessment and Revision. Technical Report Deliverable D4.3, PLASTIC Consortium, May 2008. IST STREP Project. <http://www.ist-plastic.org>
- [BBF09] G. Blair, N. Bencomo and R. B. France. Models@run.time. Computer journal, volume 42, issn: 0018-9162, 2009, pages: 22-27, IEEE Computer Society.
- [BBFJ09] N. Bencomo, G. Blair, F. Fleurey, and C. Jeanneret Editors. Proceedings of the 5th Workshop on Models@run.time at MODELS 2010 Oslo, Norway, October 5th 2010.
- [BBGS09] L. Baresi, D. Bianculli, S. Guinea, P. Spoletini: Keep It Small, Keep It Real: Efficient Run-Time Verification of Web Service Compositions. 29th Formal Techniques for Networked and Distributed Systems, 2009 (to appear)
- [BBIT+00] G. S. Blair, L. Blair, V. Issarny, P. Tuma, and A. Zarras. The Role of Software Architecture in Constraining Adaptation in Component-Based Middleware Platforms. In Proceedings of the 2nd ACM-IFIP-USENIX International Middleware Conference (MIDDLEWARE), 2000.
- [BBKG+09] N. Ben Mabrouk, S. Beauche, E. Kuznetsova, N. Georgantas, V. Issarny. QoS-aware Service Composition in Dynamic Service Oriented Environments. In Proc. Middleware 2009 - ACM/IFIP/USENIX, 10th International Conference, Urbana-Champaign, IL, USA, 2009
- [BBMP09] C. Bartolini, A. Bertolino, E. Marchetti, A. Polini. WS-TAXI: A WSDL based testing tool for Web Services. In Proc. ICST2009. 326-335, 2009
- [BCDF+07] G. Behrmann, A. Coughard, A. David, E. Fleury, K. G. Larsen, and D. Lime. UPPAAL-Tiga: Time for Playing Games!. CAV 2007
- [BCDK+10] A. Bucchiarone, C. Cappiello, E. Di Nitto, R. Kazhamiakin, V. Mazza, M. Pistore. Design for Adaptation of Service-Based Applications: Main Issues and Requirements. In Proceedings of the 2009 ICSOC/ServiceWave Workshops, LNCS 6275, 2010.
- [BDFP08] A. Bertolino, G. De Angelis, L. Frantzen, A. Polini. Model-Based Generation of Testbeds for Web Services, in Proc. of Testcom/Fates 2008, June 10-13, Tokyo, Japan, Springer LNCS 5047, 2008, pp.266-282.
- [BDO06] A. Barros, M. Dumas, and P. Oaks. Standards for Web Service Choreography and Orchestration: Status and Perspectives, presented at Business Process Management Workshops, Nancy, France, 2006
- [BDPS10] A. Bertolino, G. De Angelis, A. Polini, A. Sabetta: Trends and research issues in SOA validation, in Performance and Dependability in Service Computing: Concepts, Techniques and Research Directions, Valeria Cardellini, Emiliano Casalicchio, Kalinka Regina. J. L. Castelo Branco, Julio Cezar Estrella, Francisco Jose Monaco, published by IGI Global, accepted for publication
- [BDTC05] X. Bai, W. Dong, W.-T. Tsai, and Y. Chen. WSDL-based automatic test case generation for web services testing. In Proc. of IEEE Int. Work. SOSE, pages 215-220, Washington, USA, 2005. IEEE Comp. Soc.
- [Bec02] K. Beck. Test-Driven Development: By Example. Addison-Wesley, 2002.
- [BFHL+10] F. Baude, I. Filali, F. Huet, V. Legrand, E. Mathias, P. Merle, C. Ruz, R. Krummenacher, E. Simperl, C. Hammerling, and J. Lorre. 2010. ESB federation for large-scale SOA. In Proceedings of the 2010 ACM Symposium on Applied Computing (Sierre, Switzerland, March 22 - 26, 2010). SAC '10. ACM, New York, NY, 2459-2466.
- [BGG06] L. Baresi, C. Ghezzi, and S. Guinea. Studies in Computational Intelligence. Springer, 2006, vol. 42, ch. Towards Self-healing Compositions of Services.
- [BGKP08] L. Baresi, S. Guinea, R. Kazhamiakin, M. Pistore: An Integrated Approach for the Run-Time Monitoring of BPEL Orchestrations. In Towards a Service-Based Internet, ServiceWave 2008,

Madrid, Spain.

- [BGLB+04] C. Baier, M. Größer, M. Leucker, B. Bollig, F. Ciesinski. Controller Synthesis for Probabilistic Systems. IFIP TCS 2004: 493-506.
- [BGNS09] L. Baresi, S. Guinea, O. Nano, G. Spanoudakis: Comprehensive Monitoring of BPEL Processes. IEEE Internet Computing, 2009 (accepted)
- [BGP08] L. Baresi, S. Guinea, L. Pasquale: Integrated and Composible Supervision of BPEL Processes. 6th International Conference on Service Oriented Computing (ICSOC 2008), Sydney, Australia, 2008
- [BHMN+07] D. Booth, H. Haas, F. McCabe, E. Newcomer, M. Champion, C. Ferris and D. Orchard. Web Services Architecture. Technical Report, W3C, 2004
- [BHPM+10] D. Bimschas, H. Hellbrück, D. Pfisterer, R. Mietz, K. Römer, and T. Teubler, Middleware for Smart Gateways Connecting Sensor networks to the Internet, International workshop on Middleware Tools, Services and Run-Time Support for Sensor Networks (MidSens'10), Colocated with Middleware 2010, November 2010
- [BI03] M. Bernardo, P. Inverardi (Eds): Formal Methods for Software Architectures, LNCS 2804. 2003.
- [BI05] Y-D. Bromberg, V. Issarny. INDISS: Interoperable Discovery System for Networked Services. In Proceedings of ACM/IFIP/USENIX 6th International Middleware Conference (Middleware), 2005.
- [BISZ98] C. Bidan, V. Issarny, T. Saridakis and A. Zarras. A Dynamic Reconfiguration Service for CORBA. In Proceedings of the 4th International Conference on Configurable Distributed Systems (ICCDs), 1998.
- [Biztalk] Microsoft Biztalk server Enterprise Service Bus <http://www.microsoft.com/biztalk/en/us/esb-guidance.aspx> 2010
- [BizUnit] A framework for Automated Testing of Distributed Systems. <http://bizunit.codeplex.com/>
- [BK06] F. van Breugel and M. Koshkina. Models and verification of BPEL, 2006. <http://www.cse.yorku.ca/~franck/research/drafts/tutorial.pdf>.
- [BKK03] D.M. Berry, E. Kamsties, and M.M. Krieger. 'From contact drafting to software specification: Linguistic sources of ambiguity, a handbook.' Tech. rep., School of Computer Science, University of Waterloo, Canada, 2003.
- [BMS07] A. Bucchiarone, H. Melgratti, F. Severoni. "Testing Service Composition" in Proceedings of the 8th Argentine Symposium on Software Engineering. Mar del Plata, 2007.
- [BMT06] W. A. Brown, G. Moore and W. Tegan, SOA governance—IBM's approach., Application Innovation IBM Global Services Effective governance through the IBM SOA Governance Management Method approach White paper, August 2006
- [BP05] A. Bertolino, A. Polini. The Audition Framework for Testing Web Services Interoperability. In Proceedings of 31th IEEE Euromicro Conference on Software Engineering and Advanced Applications (EUROMICRO), 2005.
- [BP06] A. Brogi and R. Popescu. Automated generation of BPEL adapters. In ICSOC 2006.
- [BPB08] I. Brandic, S. Pillana, and S. Benkner, "Specification, Planning, and Execution of QoS-aware Grid Workflows within the Amadeus Environment," pp. 331–345, Mar. 2008
- [BPMNv1.2] Business Process Modelling Notation, Version 1.2, OMG <http://www.omg.org/spec/BPMN/1.2/PDF>
- [BPMNv2.0] Business Process Modelling Notation, Version 2, OMG

- [BPSS] UN/CEFACT and OASIS. UN/CEFACT - ebXML Business Process Specification Schema, Nov. 2003. Version 1.11
- [BS08] S. Bliudze and J. Sifakis. The Algebra of Connectors - Structuring Interaction in BIP. IEEE Transactions on Computers, 57(10):1315–1330, 2008.
- [BSRI09] A. Bennaceur, P. Singh, P.-G. Raverdy, V. Issarny, "The iBICOOP middleware: Enablers and services for emerging pervasive computing environments," Pervasive Computing and Communications, IEEE International Conference on, pp. 1-6, 2009 IEEE International Conference on Pervasive Computing and Communications, 2009
- [BTECH] Bostech ChainBuilder ESB, <http://www.chainforge.net>
- [BTRR04] N. Badr, A. Taleb-Bendiab, M. Randles, D. Reilly, "A Deliberative Model for Self-Adaptation Middleware Using Architectural Dependency", Proceedings of the 15th International Workshop on Database and Expert Systems Applications (DEXA'04)
- [BWDT+07] X. Bai, Y. Wang, G. Dai, W.-T. Tsai, and Y. Chen: A framework for contract-based collaborative verification and validation of web services, in CBSE 2007: Proceedings of the 10th International Symposium on Component-Based Software Engineering, vol. 4608 of LNCS, pp. 258–273, Medford, Massachusetts, USA, 2007, Springer.
- [BYVB+09] R. Buyya, C-S. Yeo, S. Venugopal, J. Broberg, I. Brandic. Cloud Computing and Emerging IT Platforms: Vision, hype, and reality for delivering computing as the 5th utility. Future Generation Computer Systems, 25(6), 599-616, 2009.
- [CACM05] S. Chetan, J. Al-Muhtadi, R. Campbell and M.D. Mickunas. Mobile Gaia: A Middleware for Ad-hoc Pervasive Computing. In IEEE Consumer Communications & Networking Conference (CCNC 2005) , Las Vegas, Jan. 2005.
- [CBAC+06] W. Cirne, F. Brasileiro, N. Andrade, L. Costa, A. Andrade, R. Novaes, and M. Mowbray, "Labs of the world, unite!!!" in Journal of Grid Computing, 4(3):225–246, 2006.
- [CBDI] CBDI. CBDI-SAETM Meta Model for SOA Version 2.0. Technical Report, www.cbdiforum.com/public/meta_model_v2.php, 2007
- [CC08] S.-N. Chuang, and A.T.S. Chan, "Dynamic QoS Adaptation for Mobile Middleware", Software Engineering, IEEE Transactions on , vol.34, no.6, pp.738-752, Nov.-Dec. 2008
- [CCGL+07] P. Costa, G. Coulson, R. Gold, M. Lad, C. Mascolo, L. Mottola, G.P. Picco, T. Sivaharan, N. Weerasinghe, and S. Zachariadis, "The RUNES Middleware for Networked Embedded Systems and its Application in a Disaster Management Scenario", In Proceedings of the 5th Annual IEEE International Conference on Pervasive Computing and Communications (PERCOM'07), White Plains, New York, 19-23 March 2007.
- [CCGL+09] V. Cardellini, E. Casalicchio, V. Grassi, F. Lo Presti, and R. Mirandola. Qos-driven Runtime Adaptation of Service Oriented Architectures. In Proceedings of the 7th joint European Software Engineering Conference and the ACM SIGSOFT Symposium on the Foundations of Software Engineering (ESEC/FSE), 2009.
- [CD07] A. Cicchetti and D. Di Ruscio. "Decoupling Web Application Concerns through Weaving Operations". In Science of Computer Programming, Elsevier, 2007.
- [CD08] L. Cavallaro and E. Di Nitto. An Approach to Adapt Service Requests to Actual Service Interfaces. In Proceedings of the International Workshop on Software Engineering for Adaptive and Self-Managing Systems (SEAMS), 2008.
- [CDDD+05] M. Colombo, E. Di Nitto, M. Di Penta, D. Distanto, M. Zuccalà. Speaking a Common Language: A Conceptual Model for Describing Service-Oriented Systems. In Proceedings of the 3rd International Conference on Service Oriented Computing (ICSOC 2005), Springer, 2005

- [CDEV08] G. Canfora, M. Di Penta, R. Esposito, and M. L. Villani. A Framework for QoS-aware Binding and Re-binding of Composite Web Services. *Journal of Systems and Software*, 81(10), 1754-1769, 2008.
- [CDFL+05] F. Cassez, A. David, E. Fleury, K. G. Larsen, and D. Lime. Efficient On-the-Fly Algorithms for the Analysis of Timed Games. *CONCUR 2005*.
- [CDFP10] L. Cavallaro, E. Di Nitto, C. A. Furia and M. Pradella. A Tile-Based Approach for Self-Assembling Service Compositions. In *Proceedings of the 15th IEEE International Conference on Engineering of Complex Computer Systems*, 2010.
- [CDLM+08] D. Calvanese, G. De Giacomo, M. Lenzerini, M. Mecella, and F. Patrizi. Automatic Service Composition and Synthesis: the Roman Model. *IEEE Data Eng. Bull.*, 31(3), 2008
- [CDM06] M. Colombo, E. Di Nitto, and M. Mauri. SCENE: A Service Composition Execution Environment Supporting Dynamic Changes Disciplined Through Rules. In *Proceedings of the 4th International Conference on Service Oriented Computing (ICSOC)*, 2006.
- [CDP09] L. Cavallaro, E. Di Nitto and M. Pradella. An Automatic Approach to Enable Replacement of Conversational Services. In *Proceedings of the 7th International Joint Conference on Service-Oriented Computing (ICSOC-ServiceWave)*, 2009.
- [CDPP09] F. Corradini, F. De Angelis, A. Polini, A. Polzonetti: A participant testing strategy for service orchestration. *ICDIM 2008*: 308-313
- [CentraSite] CentraSite A little structure goes a long way Software AG 2010 available at <http://www.softwareag.com/corporate/products/wm/soagovernance/centrasite/partner/default.asp>
- [CER02] F. Curbera, D. Ehnebuske and D. Rogers. Using WSDL in a UDDI Registry, Version 1.07. UDDI Best Practice Report. May 21, 2002 Available at <http://www.uddi.org/pubs/wsdlbestpractices.pdf>, last accessed 17-11-2010.
- [CFCB10] T. Cao, P. Felix, R. Castanet, I. Berrada: Online Testing Framework for Web Services. *ICST 2010* p. 363-372
- [CGCK06] R. Y. de Camargo, A. Goldchleger, M. Carneiro, and F. Kon, "The Grid Architectural Pattern: Leveraging Distributed Processing Capabilities" in *PloPD5*. Dragos Manolescu; James Noble; Markus Völter (editors). Addison Wesley, pages 337-56. 2006.
- [CGFP09] C. Cetina, P. Giner, J. Fons and V. Pelechano. Autonomic Computing through Reuse of Variability Models at Runtime: The Case of Smart Homes. *Computer journal*, volume 42, issn: 0018-9162, 2009, pages: 37-43, IEEE Computer Society.
- [CH06] H. Cervantes, and R.S. Hall, Automating Service Dependency Management in a Service-Oriented Component Model., *ICSE CBSE6 Workshop*, 2003 .
- [Cha04] D. Chappell, *Enterprise Service Bus*, O'Reilly Media, Inc. 2004
- [Che09] B. H.C. Cheng. Using Models@Run Time to Manage Ultra-Large Scale Systems. Sidebar in Guest Editor's Introduction, Gordon Blair, Nelly Bencomo, Robert France, Models at Run Time," *IEEE Computer*, vol. 42, no, 10, October, 2009, pp. 30.
- [CI10] V. Cardellini and S. Iannucci. Designing a Broker for QoS-driven Runtime Adaptation of SOA Applications. In *Proceedings of the IEEE International Conference on Web Services (ICWS)*. 2010.
- [CKDZ+07] I. Celino, A. Karla, A. De Medeiros, G. Zeissler, M. Oppitz, F. Facca, S. Zoeller: Semantic Business Process Analysis. *Proceedings of the Workshop SBPM 2007*, Innsbruck, 2007
- [CLQ08] H. Casanova, A. Legrand, and M. Quinson. Simgrid: A generic framework for large-scale distributed experiments. In *UKSIM '08: Proceedings of the Tenth International Conference on Computer Modeling and Simulation*, pages 126–131, Washington, DC, USA, 2008. IEEE

- [CMMP07] P. Costa, L. Mottola, A. L. Murphy, G.P. Picco. "Programming Wireless Sensor Networks with the TeenyLIME Middleware". In Proceedings of the 8th ACM/IFIP/USENIX International Middleware Conference (Middleware 2007), Newport Beach (CA, USA), November 26-30, 2007.
- [Cosminexus] Cosminexus, Hitachi, Ltd 2010 available at <http://www.hitachi.co.jp/Prod/comp/soft1/global/prod/cosminexus/>
- [CPS06] C. Canal, P. Poizat, and G. Salaün. Synchronizing Behavioural Mismatch in Software Composition. FMOODS, 2006.
- [CRI07] R. S. Cardoso, P-G. Raverdy, V. Issarny. A Privacy-Aware Service Discovery Middleware for Pervasive Environments. In IFIPTM 2007 Joint iTrust and PST Conferences on Privacy, Trust Management and Security, 2007
- [Cross-ETP] Future Internet, The Cross-ETP Vision Document, 2009 http://www.future-internet.eu/fileadmin/documents/reports/Cross-ETPs_FI_Vision_Document_v1_0.pdf
- [CRZ09] L. Cavallaro G. Ripa, and M. Zuccalà, Adapting Service Requests to Actual Service Interfaces through Semantic Annotations. In Proceedings of Principles of Engineering Service Oriented Systems (PESOS 2009), ICSE 2009 Workshop, Vancouver, Canada, 18-19 May, 2009
- [CVRB+08] R.D. Callaway, Y. Viniotis, A. Rodriguez, K.G. Brown, and R. Robinson. Enabling federations of enterprise service buses using a distributed service registry. Proceedings of the Second international workshop and Summer School on Service Science, Management and Engineering, SSME2008, Palermo Italy, 2-6 June 2008.
- [CVRG09] K. Christos, C. Vassilakis, E. Rouvas and P. Georgiadis. QoS-Driven Adaptation of BPEL Scenario Execution. In Proceedings of the IEEE International Conference on Web Services (ICWS), 2009.
- [CY08] J. Choi, C.-W. Yoo. Connect with Things through Instant Messaging. Internet of Things 2008, International Conference for Industry and Academia. March 26-28, 2008. Zurich.
- [CZ08] C. Chao, Q. Zongyan. An Approach to Check Choreography with Channel Passing in WS-CDL, icws, pp.700-707, 2008 IEEE International Conference on Web Services, 2008
- [DAML-S] DAML-S. The DARPA Agent Markup Language. Available at <http://www.ai.sri.com/daml/services/owl-s/>
- [DB08] J. J. Davies, A. R. Beresford. Scalable Inter-Vehicular Applications. On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops (Part II). LNCS 4806, pp 876--885. Springer-Verlag, November 2007.
- [DCJ06] J. J. Davies, D. N. Cottingham, B. D. Jones. A Sensor Platform for Sentient Transportation Research. 1st European Conference on Smart Sensing and Context (EuroSSC) 2006, Enschede, The Netherlands.(LNCS volume 4272, pages 226--229, 2006)
- [DD04] R. Dijkman and M. Dumas. Service-oriented Design: A Multi-viewpoint Approach. International Journal of Cooperative Information Systems, 13(4):337–368, 2004
- [DDGR+07] E. D. Nitto, M. Di Penta, A. Gambi, G. Ripa, and M. Villani. Negotiation of Service Level Agreements: An Architecture and a Search-Based Approach. In Proceedings of the 5th International Conference on Service-Oriented Computing (ISOC), 2007.
- [DDH07] T. Dyba, T. Dingsoyr, and G. K. Hanssen. Applying systematic reviews to diverse study types: An experience report. In ESEM '07: Proceedings of the First International Symposium on Empirical Software Engineering and Measurement, pages 225–234, 2007.
- [DDO08] R. M. Dijkman, M. Dumas, and C. Ouyang. Semantics and analysis of business process models in bpmn. Inf. Softw. Technol., 50(12):1281–1294, 2008.

- [Dec09] G. Decker. Design and Analysis of Process Choreographies. PhD thesis, Hasso Plattner Institute, University of Potsdam, Germany, 2009.
- [DHL05] Z. Du, J. Huai, and Y. Liu. Ad-UDDI: An Active and Distributed Service Registry. TES 2005, LNCS 3811, pp. 58 – 71, 2005.
- [Dil06] J. Dilley. Web server workload characterization. Technical report, HP Labs, December 1996. <http://www.hpl.hp.com/techreports/96/HPL-96-160.html>.
- [DKLW07] G. Decker, O. Kopp, F. Leymann, M. Weske. BPEL4Chor: Extending BPEL for Modeling Choreographies. In Proceedings of the International Conference on Web Services, 2007
- [DLS05] G. Dobson, R. Lock, and I. Sommerville, “Quality of Service Requirements Specification using an Ontology” Proc. Service-Oriented Computing: Consequences for Engineering Requirements (SOCCER 05) at 13th Int’l Requirements Engineering Conf. (RE 05), 2005
- [DMMP+10] D. Di Ruscio, I. Malavolta, H. Muccini, P. Pelliccione and A. Pierantonio. Developing Next Generation ADLs Through MDE Techniques. In Proceedings of the 32nd International Conference on Software Engineering (ICSE 2010), 2010.
- [DP05] D. Di Ruscio and A. Pierantonio. “Model Transformations in the Development of data-intensive Web Applications”. CAiSE’05, O. Pastor and J. F. e Cunha (Eds.), Springer LNCS 3520, 2005, pp 475-490.
- [DRW07] L. Duboc, D. Rosenblum, and T. Wicks. A framework for characterization and analysis of software system scalability. In ESEC-FSE ’07: Proceedings of the the 6th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering, pages 375–384, New York, NY, USA, 2007. ACM
- [DRZ10] T. De Giorgio, G. Ripa, and M. Zuccalà, An Approach to Enable Replacement of SOAP Services and REST Services in Lightweight Processes 2nd International Workshop on Lightweight Composition on the Web (ComposableWeb 2010), of the 10th International Conference on Web Engineering (ICWE 2010), Vienna, Austria, 7-9 July, 2010
- [DT05] S. Dustdar and M. Treiber. A View Based Analysis on Web Service Registries. Distributed and Parallel Databases 18(2):147-171, 2005.
- [DW07] P. Derler and R. Weinreich, Models and Tools for SOA Governance, in Proceedings of Trends in Enterprise Application Architecture Lecture Notes in Computer Science 4473, 2007, Springer Berlin / Heidelberg – pages: 112-126
- [DY06] W.-L. Dong and H. Yu, “Web service testing method based on fault-coverage,” in EDOC Workshops: The 10th IEEE International Enterprise Distributed Object Computing Conference (EDOC’06), pp. 43–43, Hong Kong, China, Oct. 2006, IEEE Computer Society.
- [EGA1] Enterprise Grid Alliance. Reference Model and Use Cases - Part 1 of 2. EGA, 2006
- [EGA2] Enterprise Grid Alliance. Reference Model and Use Cases - Part 2 of 2. EGA, 2006
- [EHM10] K. Elgazzar, A. E. Hassan, P. Martin. Clustering WSDL Documents to Bootstrap the Discovery of Web Services. IEEE International Conference on Web Services, ICWS 2010, pp. 147-154, Miami, Florida, USA, July 5-10, 2010.
- [EMLP+07] S. B. Eisenman, E. Miluzzo, N. D. Lane, R. A. Peterson, G-s. Ahn, A. T. Campbell, “The BikeNet Mobile Sensing System for Cyclist Experience Mapping”, 5th ACM Conf. on Embedded Networked Sensor Systems, 2007
- [EMMP+10] R. Eramo, I. Malavolta, H. Muccini, P. Pelliccione and A. Pierantonio. A Model-Driven Approach to Automate the Propagation of Changes Among Architecture Description Languages. Journal of Software and Systems Modeling, 2010 (to appear).
- [Eng05] R. van Engele. Are web services scale free? <http://www.cs.fsu.edu/~engelen/powerlaw.html>,

june 16 2005.

- [FBC09] M. Finger, G. Bezerra, D. M. R. Conde. "Resource use pattern analysis for predicting resource availability in opportunistic grids" in Concurrency and Computation. Practice & Experience, v. 22, p. 295-313, 2010.
- [Fer04] A. Ferreira. Building a reference combinatorial model for manets. IEEE Network, 18(5):24–29, 2004.
- [FF95] S. Fickas, M.S. Feather, Requirements Monitoring in Dynamic Environments, Proceedings of the Second IEEE International Symposium on Requirements Engineering, IEEE Computer Society Press, York, England, March 1995.
- [FGIZ08] M. Fredj, N. Georgantas, V. Issarny, and A. Zarras. Dynamic Service Substitution in Service-Oriented Architectures. In Proceedings of the IEEE International Conference on Service Computing (SCC), 2008.
- [FGM10] A. Ferreira, A. Goldman, and J. Monteiro. Performance evaluation of routing protocols for manets with known connectivity patterns using evolving graphs. Wireless Networks, 16(3):627–640, 2010.
- [FIA-FCN] FIA Future Content Networks (FCN), Scenarios, May 2009, http://www.future-internet.eu/fileadmin/documents/prague_documents/FIA-FCN_Internet_Scenarios_20090507.pdf
- [FIA-FISE] FIA Future Internet Socio-Economics (FISE), FISE Scenarios and Effects, February 2009, <http://www.smoothit.org/wiki/pmwiki.php/FISE/Scenarios>
- [FIA-FISO] FIA Future Internet Service Offer (FISO), Scenarios, February 2009, http://services.future-internet.eu/index.php/FISO_Scenarios
- [FIA-MANA] FIA Management and Service-aware Architectures (MANA), Scenarios for Future Internet, May 2009, http://www.futureinternet.eu/fileadmin/documents/prague_documents/MANA_Scenarios-Final.pdf
- [FIA-RWI] FIA Real World Internet (RWI), Scenarios, March 2009, http://rwi.future-internet.eu/index.php/RWI_Scenarios
- [FIA-TI] FIA Trust & Identity (T&I), Scenarios, May 2009, http://security.future-internet.eu/images/1/1d/TI_Scenarios.pdf, and http://www.futureinternet.eu/fileadmin/documents/prague_documents/FIA_Prague_TI_Session_s_Agenda_and_ToR_new.pdf
- [Fie00] R. T. Fielding, Architectural Styles and the Design of Network-based Architectures, PhD thesis, University of California, Irvine, Irvine, California, 2000.
- [Fiorano] Fiorano ESB (Enterprise Service Bus) Fiorano Software Technologies2010 available at <http://www.fiorano.com/products/ESB-enterprise-service-bus/Fiorano-ESB-enterprise-service-bus.php>
- [FIRE09] FIRE Use Scenarios (FIRE), May 2009, http://www.ict-fireworks.eu/fileadmin/events/FIA-Prague/1_Avessta.pdf
- [FK03] I. Foster and C. Kesselman (Eds.). The Grid 2: Blueprint for a New Computing Infrastructure. Morgan Kaufmann Publishers, 2003.
- [FK97] I. Foster and C. Kesselman. (1997). "Globus: A Metacomputing Infrastructure Toolkit" in International Journal of Supercomputer Applications, 11(2):115–118.
- [FT02] R Fielding and R Taylor. Principled Design of the Modern Web Architecture. ACM Transactions on Internet Technology, 2(2), 115-150, 2002.
- [FT04] M. Fluegge and D. Tourtchaninova. Ontology-derived activity components for composing travel

web services. The International Workshop on Semantic Web Technologies in Electronic Business (SWEB2004), 2004.

- [FTV06] L. Frantzen, J. Tretmans, and R. d. Vries. Towards Model-Based Testing of Web Services. In International Workshop on Web Services – Modeling and Testing – WS-MaTe 2006, pages 67-82. Palermo, Italy, 2006.
- [FUMK06] H. Foster, S. Uchitel, J. Magee, and J. Kramer. Model-Based Analysis of Obligations in Web Service Choreography, aict-iciw, pp.149, Advanced International Conference on Telecommunications and International Conference on Internet and Web Applications and Services (AICT-ICIW'06), 2006
- [FUSE] Fuse ESB 4 - OSGi ESB (ServiceMix 4) <http://fusesource.com/products/enterprise-servicemix/#documentation> Progress Software Corporation 2010
- [FZGI10] M. Fredj, A. Zarras, N. Georgantas and V. Issarny. Dynamic AMaintenance of Service Orchestrations. In Service Intelligence and Service Science: Evolutionary Technologies and Challenges, IGI, 2010.
- [FZRL09] I. T. Foster, Y. Zhao, I. Raicu, and S. Lu. Cloud computing and grid computing 360-degree compared. CoRR, abs/0901.0131, 2009.
- [Galaxy] Galaxy MuleSoft 2009 available at <http://www.mulesoft.org/documentation/display/GALAXY/Home>
- [GC08] H. J. Goldsby and B. H. C. Cheng. Automatically Generating Behavioral Models of Adaptive Systems to Address Uncertainty. Proc. 11th Int'l Conf. Model Driven Eng. Languages and Systems (MoDELS 08). LNCS 5301. Springer-Verlang. 2008.
- [GHJV95] E. Gamma, R. Helm, R. Johnson and J. Vlissides. Design Patterns: Elements of Reusable Object- Oriented Software. Addison-Wesley, 1995.
- [GHT09] J. C. Georgas, A. van der Hoek and R. N. Taylor. Using Architectural Models to Manage and Visualize Runtime Adaptation. Computer journal, volume 42, issn: 0018-9162, 2009, pages: 52-60, IEEE Computer Society.
- [Gil05] T. Gilb, 'Competitive Engineering: A Handbook for Systems & Software Engineering Management using Planguage', 2005.
- [GIWO09] T. Gjerlufsen, M. Ingstrup, J. Wolff and O. Olsen. Mirrors of Meaning: Supporting Inspectable Runtime Models. Computer journal, volume 42, issn: 0018-9162, 2009, pages: 61-68, IEEE Computer Society.
- [GK96] K.M. Goudarzi and J. Kramer. Maintaining Node Consistency in the Face of Dynamic Change. In Proceedings of the 3rd International Conference on Configurable Distributed Systems (ICCDs), 1996.
- [GKGF+04] A. Goldchleger, F. Kon, A. Goldman, M. Finger, and G. C. Bezerra. "InteGrade: Object-Oriented Grid Middleware Leveraging Idle Computing Power of Desktop Machines" in Concurrency and Computation: Practice & Experience, 16, 2004.
- [GL10] A. Gómez-Goiri and D. López-de-Ipiña, A Triple Space-Based Semantic Distributed Middleware for Internet of Things, TouchTheWeb'10 International Workshop on Web-enabled Objects, ICWE 2010, The Tenth International Conference on Web Engineering, 2010
- [GL94] O. Grumberg and D. Long. Model checking and modular verification. ACM Transactions on Programming Languages and Systems 13(6), pp. 843-871, 1994.
- [Gro07] A. Grosskopf. XBPMN. formal control flow specification of a BPMN based process execution language. Master thesis, 2007.
- [GS93] D. Garlan, and M. Shaw. An Introduction to Software Architecture. In V. Ambriola and G. Tortora (ed.), Advances in Software Engineering and Knowledge Engineering, World Scientific

- [GS96] D. Garlan and M. Shaw. Software Architecture: Perspectives on an Emerging Discipline, Prentice-Hall 1996.
- [GSCL+04] M. Govindaraju, E. Slominski, K. Chiu, P. Liu, R. Van Engelen, and M. J. Lewis. Toward characterizing the performance of soap toolkits. In In GRID 04, pages 365–372, 2004.
- [GT06] S. Gilmore and M. Tribastone. Evaluating the scalability of a web service-based distributed e-learning and course management system. In Third International Workshop on Web Services and Formal Methods (WS-FM06). Volume 4184 of Lecture Notes in Computer Science, pages 156–170. Springer, 2006.
- [GT09] D. Guinard and V. Trifa, Towards the Web of Things: Web Mashups for Embedded Devices, Workshop on Mashups, Enterprise Mashups and Lightweight Composition on the Web (MEM 2009), in proceedings of WWW (International World Wide Web Conferences), Madrid, Spain, April 2009.
- [Hay08] B. Hayes. Cloud Computing. Comm. of the ACM, 51(7), 9-11, 2008
- [HH08] B. Hofreiter, C. Huemer. A model-driven top-down approach to inter-organizational systems: From global choreography models to executable BPEL. IEEE Conference on E-Commerce Technology (CEC'08), Washington D.C., USA; 22.07.2008 - 24.07.2008; in IEEE Joint Conference on E-Commerce Technology (CEC'08) and Enterprise Computing, E-Commerce, and E-Services (EEE'08), IEEE Computer Society, (2008), ISBN: 978-0-7695-3340-7; S. 136 - 145
- [HHLS+06] B. Hofreiter, C. Huemer, P. Liegl, R. Schuster, and M. Zapletal. UN/CEFACT'S Modeling Methodology (UMM): A UML Profile for B2B e-Commerce. In Advances in Conceptual Modeling - Theory and Practice, ER 2006 Workshops, pages 19–31. Springer LNCS 4231, 2006
- [HM05] R. Heckel and L. Mariani: Automatic Conformance Testing of Web Services, proceedings of the 8th International Conference on Fundamental Approaches to Software Engineering (FASE 2005), 2005
- [HMY06] G. Huang, H. Mei, and F. Yang. Runtime Recovery and Manipulation of Software Architecture of Component-based Systems. International Journal of Automated Software Engineering, 13(2), 2006.
- [HP-GIF] HP SOA Governance Interoperability Framework (GIF) GIF - Hewlett-Packard Development Company, L.P. 2010 available at https://h10078.www1.hp.com/cda/hpms/display/main/hpms_content.jsp?zn=bto&cp=1-11-130-27^2804_4000_100__
- [HSBP10] P. Hens, M. Snoeck, M. de Backer, and G. Poels. Decentralized Event-based Orchestration in edBPM 2010 Proceedings of the 4th international workshop on Event Driven Business Process Management, pages 1-12, Hoboken, NJ, USA, 2010, BPM 2010.
- [HW96] S. Hauptmann and J. Wassel. On-line Maintenance with On-the-Fly Software Replacement. In Proceedings of the 3rd International Conference on Configurable Distributed Systems (ICCDs), 1996.
- [HXWX+07] Y. Huang, C. Xu, H. Wang, Y. Xia, J. Zhu, C. Zhu, Formalizing Web Service Choreography Interface. In AINAW, vol. 2, pp.576-581, 21st International Conference on Advanced Information Networking and Applications Workshops (AINAW'07), 2007
- [Ing09] D. Ingram. 2009. Reconfigurable middleware for high availability sensor systems. In Proceedings of the Third ACM International Conference on Distributed Event-Based Systems (DEBS '09). ACM, New York, NY, USA, Article 20, 11 pages.
- [Inv07] P. Inverardi. "Software of the future is the future of Software?" TGC 2006, LNCS 4661, Springer-Verlag.

- [IOTFR] INFISO D.4 Networked Enterprise & RFID, INFISO G.2 Micro & Nanosystems, RFID Working Group Of The European Technology Platform On Smart Systems Integration (EPOSS). Internet of Things in 2020; A Roadmap For The Future. 05 September, 2008.
- [IOTSRR] SRA Cluster of European Projects on RFID and Internet of Things. Internet of Things; Strategic Research Roadmap. 15 September 2009.
- [ISOSA] ISO. Fourth working draft of Systems and Software Engineering – Architectural Description (ISO/IECWD4 42010). Working doc.: ISO/IEC JTC 1/SC 7 N 000, 2009.
- [ISTS+05] V. Issarny, D. Sacchetti, F. Tartanoglu, F. Sailhan, R. Chibout, N. Levy, A. Talamona. Developing Ambient Intelligence Systems: A Solution based on Web Services. Journal of Automated Software Engineering. 12. 2005.
- [IT03] P. Inverardi, M. Tivoli. Deadlock-free software architectures for COM/DCOM Applications. Elsevier Journal of Systems and Software, 2003.
- [IT04] P. Inverardi, M. Tivoli. Software Architecture for Correct Components Assembly, Springer, LNCS 2804, 2004
- [ITEM] ANR ITEMIS project, <http://itemis-anr.org>
- [Jac95] M. Jackson. "Software Requirements and Specifications". Addison-Wesley, 1995.
- [JBI] Oracle Corporation available at <http://jcp.org/aboutJava/communityprocess/final/jsr208/index.html> 2010
- [JW00] P. Jogalekar and M. Woodside. Evaluating the scalability of distributed systems. IEEE Trans. Parallel Distrib. Syst., 11(6):589–603, 2000.
- [KBSF09] R. Krummenacher, D. Blunder, E. Simperl, M. Fried, An open distributed middleware for the semantic web. International Conference on Semantic Systems (I-SEMANTICS) (2009)
- [KDWL09] L. Kuang, S. Deng, J. Wu, and Y. Li. Towards Adaptation of Service Interface Semantics. In Proceedings of the IEEE International Conference on Web Services (ICWS), 2009.
- [KEMS07] A. Kassim, B. Esfandiari, S. Majumdar and L. Serghi. A Flexible Hybrid Architecture for Management of Distributed Web Service Registries. In Proceedings of the 5th Annual Conference on Communication Networks and Services Research (CNSR), 2007
- [KG94] V. Kumar and A. Gupta. Analyzing scalability of parallel algorithms and architectures. J. Parallel Distrib. Comput., 22(3):379–391, 1994.
- [KKS09] S. Kalasapur, M. Kumar, and B. Shirazi, Dynamic Service Composition in Pervasive Computing Systems, IEEE Transactions on Parallel and Distributed Systems, In Press.
- [KLV06] P. Kruchten, P. Lago, and H. van Vliet. Building Up and Reasoning about Architectural Knowledge. In Proceedings of the 2nd International Conference on Quality of Software Architectures (QoSA), 2006.
- [KM90] J. Kramer and J. Magee. The Evolving Philosophers Problem: Dynamic Change Management. IEEE Transactions on Software Engineering 16(11), 1990.
- [KNBC+09] W. Kongdenfha, H. R. M. Nezhad, B. Benatallah, F. Casati, R. Saint-Paul. Mismatch Patterns and Adaptation Aspects: A Foundation for Rapid Development of Web Service Adapters. IEEE Transactions on Services Computing (TSC) 2(2), 94-107, 2009.
- [KPR04] R. Kazhamiakin, M. Pistore, M. Roveri, A framework for integrating Business Processes and Business Requirements 9th International IEEE Enterprise Distributed Object Computing Conference (EDOC) 2004.
- [Kro09] K.L. Kroeker. The Evolution of Virtualization. Communications of the ACM 52(3), 2009

- [KS05] A. Kucera and O. Strazovský. On the Controller Synthesis for Finite-State Markov Decision Processes. FSTTCS 2005: 541-552.
- [KT08] A. Katasonov and V. Terziyan,. Semantic Agent Programming Language (S-APL): A Middleware Platform for the Semantic Web. In: Proc. 2nd IEEE International Conference on Semantic Computing (ICSC'08), August 4-7, 2008, Santa Clara, USA, pp.504-511
- [Lak96] J. Lakos. Large-Scale C++ Software Design, Addison-Wesley Professional, 1996.
- [LBTA10] G. Lehmann, M. Blumendorf, F. Trollmann, and S. Albayrak. Meta-Modeling Runtime Models. In proceedings of the 5th Workshop on Models@run.time at MODELS 2010 Oslo, Norway, October 5th 2010.
- [LD09] B. Lim, and A.K. Dey, Assessing Intelligibility in Context-Aware Applications. Proceedings of Ubicomp 2009, Orlando, Florida, USA.
- [LHZP07] J. Li, J. He, H. Zhu, G. Pu. Modeling and Verifying Web Services Choreography Using Process Algebra, pp.256-268, 31st IEEE Software Engineering Workshop (SEW 2007), 2007
- [LL07] K-H. Lee and K-C. Lee. To Maximize Web Service Retrieval. International Conference on Convergence Information Technology (ICCIT) 2007.
- [LL08] Q. A. Liang, H. Lam. Web Service Matching By Ontology Instance Categorization. 2008 IEEE International Conference on Services Computing, pp. 202-209, 8-11 July 2008, Honolulu, Hawaii, USA
- [LLRB10] X. Liu, C. Liu, M. Rege, and A. Bouguettaya. Semantic Support for Adaptive Long Term Composed Services. In Proceedings of the IEEE International Conference on Web Services (ICWS), 2010.
- [LMKK+06] J. Luo, B. E. Montrose, A. Kim, A. Khashnobish, and M. H. Kang. Adding OWL-S Support to the Existing UDDI Infrastructure. 2006 IEEE International Conference on Web Services (ICWS 2006), pp. 153-162, 18-22 September 2006, Chicago, Illinois, USA.
- [LPLC+09] H. Lu, W. Pan, N. D. Lane, T. Choudhury and A. T. Campbell, SoundSense: Scalable Sound Sensing for People-Centric Applications on Mobile Phones, ACM MobiSys 2009, June 22–25, 2009, Kraków, Poland
- [LSYW+10] F. Liu, Y. Shi, J. Yu, T. Wang, and J. Wu. Measuring Similarity of Web Services Based on WSDL. IEEE International Conference on Web Services, ICWS 2010, pp. 147-154, Miami, Florida, USA, July 5-10, 2010.
- [LW94] B. Liskov and J. Wing. A Behavioral Notion of Subtyping. ACM Transactions on Programming Languages and Systems, 16(6), 1994.
- [LZCH08] M. Lallali, F. Zaidi, A. Cavalli, and I. Hwang: Automatic timed test case generation for web services composition, in ECOWS '08: Proceedings of the 2008 6th European Conference on Web Services, pp. 53–62, Dublin, Ireland, Nov. 2008, IEEE Computer Society.
- [LZSK+05] R. Li, Z. Zhang, W. Song, F. Ke, and Z. Lu. Service Publishing and Discovering Model in a Web Services Oriented Peer-to-Peer System. In Proceedings of the ICWE 2005, LNCS 3579, 2005.
- [LZZM09] Z. J. Li, J. Zhu, L.-J. Zhang, and N. Mitsumori: Towards a practical and effective method for web services test case generation, in Proceedings of the ICSE Workshop on Automation of Software Test (AST'09), pp. 106–114, May 2009.
- [Mao09] S. Maoz. Using Model-Based Traces as Runtime Models. Computer journal, volume 42, issn: 0018-9162, 2009, pages: 28-36, IEEE Computer Society.
- [Mar02] R. C. Martin. Agile Software Development, Principles, Patterns, and Practices, Prentice Hall, 2002.

- [Mar06] R. C. Martin. Agile Principles, Patterns and Practices in C#. Prentice Hall, 2006.
- [Mar08] E. A. Marks, Service-Oriented Architecture Governance for the Services Driven Enterprise, John Wiley & Sons, Inc. Editions, 2008
- [Master] Petals Master SOA Governance Solution 2010 available at <http://petalsmaster.ow2.org/>
- [MBHL+04] D. Martin, M. Burstein, J. Hobbs, O. Lassila, D. McDermott, S. McIlraith, S. Narayanan, M. Paolucci, B. Parsia, and T. Payne, "OWL-S: Semantic Markup for Web Services," World Wide Web Consortium, 2004.
- [MBJF+09] B. Morin, O. Barais, J. Jezequel, F. Fleurey, A. Solberg. Models@ Run.time to Support Dynamic Adaptation. Computer journal, volume 42, issn: 0018-9162, 2009, pages: 44-51, IEEE Computer Society.
- [MBMC+07] H. Motahari Nezhad, B. Benatallah, A. Martens, F. Curbera and F. Casati, F. SemiAutomated Adaptation of Service Interactions. In Proceedings of the International World Wide Web Conference (WWW), 2007.
- [MD10] T. Mens, S. Demeyer. Software Evolution, Springer, 2010.
- [Mes07] G. Meszaros. xUnit Test Patterns: Refactoring Test Code. Addison-Wesley, May 2007.
- [Mey88] B. Meyer. Object-Oriented Software Construction, Prentice Hall, 1988.
- [MF04] L. Melloul and A. Fox. Reusable Functional Composition Patterns for Web Services. In Proceedings of the IEEE International Conference on Web Services (ICWS), 2004.
- [MFHH05] S. R. Madden, M. J. Franklin, J. M. Hellerstein, and W. Hong. 2005. TinyDB: an acquisitional query processing system for sensor networks. ACM Trans. Database Syst. 30, 1 (March 2005), 122-173.
- [MGI07] S. B. Mokhtar, N. Georgantas, V. Issarny. COCOA: COnversation-based Service Composition in Pervasive Computing Environments with QoS Support. Journal of System and Software. 80(12). December 2007
- [MHZJ06] H. Mei, G. Huang, H. Zhao, and W. Jiao. An Architecture Centric Engineering Approach to Internetware. Science in China, series F, Springer, 49(6), 2006.
- [Min08] A. Mintchev. Interoperability among Service Registry Implementations: Is UDDI Standard Enough? IEEE International Conference on Web Services (ICWS), 2008.
- [MITbook] At Your Service - Service-Oriented Computing from an EU Perspective, Eds. E. Di Nitto, A.M. Sassen, P. Traverso, A. Zwegers, The MIT Press, 2009
- [MK07] G. Mega, F. Kon. "An eclipse-based tool for symbolic debugging of distributed object systems" in Proceedings of the 9th International Symposium on Distributed Objects, Middleware, and Applications (DOA'2007) , LNCS 4803, pp. 648-666. Portugal, November, 2007.
- [MKGI06] S. Ben Mokhtar, A. Kaul, N. Georgantas, V. Issarny. Efficient Semantic Service Discovery in Pervasive Computing Environments. In Proceedings of ACM/IFIP/USENIX 7th International Middleware Conference (Middleware), 2006.
- [MLGI05] S. Ben Mokhtar, J. Liu, N. Georgantas, V. Issarny. QoS-aware Dynamic Service Composition in Ambient Intelligence Environments. In Proceedings of the 2005 IEEE/ACM International Conference on Automated Software Engineering (ASE), 2005.
- [MLMB+06] C. M. MacKenzie, K. Laskey, F. McCabe, P. F. Brown, R. Metz. OASIS Reference Model for Service Oriented Architecture 1.0. Committee Specification 1, <http://www.oasis-open.org/committees/download.php/19679/soa-rm-cs.pdf>, 2006
- [MMPT10] I. Malavolta, H. Muccini, P. Pelliccione, and D. A. Tamburri. Providing Architectural Languages

and Tools Interoperability Through Model Transformation Technologies. IEEE Transaction on Software Engineering, 36(1), 2010.

- [MPBP+07] L. Mottola, A. Pathak, A. Bakshi, V. K. Prasanna and G. P. Picco, Enabling scope-based interactions in sensor network macroprogramming, 4th International Conference on Mobile, Ad-Hoc and Sensor Systems, MASS 2007, Pisa, Italy, 2007
- [MPM08] T. Melliti, P. Poizat, and S. B. Mokhtar. Distributed Behavioural Adaptation for the Automatic Composition of Semantic Services. In FASE 2008, LNCS 4961, Springer
- [MPT08] A. Marconi, M. Pistore, and P. Traverso. Automated Composition of Web Services: the ASTRO Approach. IEEE Data Eng. Bull., 31(3), 2008.
- [MRD08] O. Moser, F. Rosenberg, and S. Dustdar. Non-Intrusive Monitoring and Service Adaptation for WS-BPEL. In Proceedings of the 17th International World Wide Web Conference (WWW), 2008.
- [MRRR02] N. Medvidovic, D. Rosenblum, D. Redmiles, and J. Robbins. Modeling Software Architectures in the Unified Modeling Language. ACM Transactions on Software Engineering and Methodology, 11(1), 2002.
- [MT00] N. Medvidovic and R. Taylor. A Classification and Comparison Framework for Software Architecture Description Languages. IEEE Transaction on Software Engineering, 26(1), 70-93, 2000.
- [MULE] Mule, www.mulesoft.org
- [MZ09] M. Mamei, F. Zambonelli, "Programming Pervasive and Mobile Computing Applications: the TOTA Approach ", ACM Transactions on Software Engineering and Methodology, 2009.
- [NBMC+07] H. M. Nezhad, B. Benatallah, A. Martens, F. Curbera and F. Casati, F. SemiAutomated Adaptation of Service Interactions. In Proceedings of the 16th IEEE International World Wide Web Conference (WWW), 2007.
- [NEXOFRM] NEXOF-RA project (<http://www.nexof-ra.eu/>). Deliverable D6.3: The NEXOF-RA Reference Model V3.0. 2010 http://www.nexofra.eu/sites/default/files/D6.3_v1.0.pdf
- [NL07] R. Nayak and B. Lee. Web Service Discovery with additional Semantics and Clustering. In Proceedings of the IEEE / WIC / ACM International Conference on Web Intelligence, 2007
- [NLKL07] J. Nitzsche, T. van Lessen, D. Karastoyanova, and F. Leymann, "BPEL for Semantic Web Services (BPEL4SWS)," in On the Move to Meaningful Internet Systems 2007: OTM 2007 Workshops, ser. LNCS, Springer, Ed., vol. 4805/2007, 2007, pp. 179–188
- [NS2] The network simulator ns-2. <http://www.isi.edu/nsnam/ns/>.
- [NXB10] H. R. M. Nezhad, G-Y. Xu, B. Benatallah. Protocol-aware matching of web service interfaces for adapter development. In Proceedings of the International World Wide Web Conference (WWW), 2010.
- [OMNET] OMNET++. <http://www.omnetpp.org/>.
- [OPESB] OpenESB, <https://open-esb.dev.java.net/>
- [ORESBS] Oracle ESB, <http://www.oracle.com/appserver/esb.html>
- [OWL-S] Semantic Markup for Web Services. Available at <http://www.w3.org/Submission/OWL-S/>
- [PA91] B. Plateau, K. Atif, Stochastic automata network for modeling parallel systems. IEEE Transactions on Software Engineering 17, 1093–1108, 1991
- [PBLH08] J. Pathak, S. Basu, R. R. Lutz, and V. Honavar. MOSCOE: an Approach for Composing Web Services through Iterative Reformulation of Functional Specifications. Int. Journal on Artificial

- [PD02] R. Passerone and L. De Alfaro. Convertibility Verification and Converter Synthesis: two faces of the same coin. In Proc. of the Int. Conference on Computer-Aided Design, 2002.
- [PD07] C. Pedrinaci, J. Domingue: Towards an Ontology for Process Monitoring and Mining. In Semantic Business Process and Product Lifecycle Management (SBPM 2007), Innsbruck, Austria, 2007
- [PetalsLink] Petals ESB <http://www.petalslink.com/fr/produits/petals-esb> Petals Link EBM Websourcing, 2010
- [PETLS] OW2 PEtALS, <http://petals.ow2.org>
- [PF02] S. Ponnekanti and A. Fox. SWORD: A Developer Toolkit for Web Service Composition (2002). In Proceedings of the 11th International WWW Conference WWW2002, 2002.
- [PF04] S. Ponnekanti and A. Fox. Interoperability Among Independently Evolving Web Services. In Proceedings of the 5th ACM/IFIP/USENIX International Middleware Conference (MIDDLEWARE), 2004.
- [PG09] A. Pathak and M. K. Gowda. Srijan: a graphical toolkit for sensor network macroprogramming. In Proceedings of the the 7th joint meeting of the European software engineering conference and the ACM SIGSOFT symposium on The foundations of software engineering (ESEC/FSE '09). ACM, New York, NY, USA, 301-302, 2009.
- [PGXP+09] HK Pung, T Gu, W. Xue, PP Palmes, J Zhu, WL Ng, CW Tang and NH Chung, Context-Aware Middleware for Pervasive Elderly Homecare, IEEE Journal of Selected Areas of Communications, Volume 27, Issue 4, May 2009 Page(s):510 – 524
- [PIM09] P. Pelliccione, P. Inverardi and H. Muccini, CHARMY: A Framework for Designing and Verifying Architectural Specifications. IEEE Transactions on Software Engineering (TSE), 35(3), 325 – 346, 2009.
- [PKPS02] M. Paolucci, T. Kawamura, T. R. Payne, K. P. Sycara. Importing the Semantic Web in UDDI. Revised Papers for International Workshop on Web Services, E-Business, and the Semantic Web (WES 2002), in conj. with CAiSE 2002, Toronto, Canada, May 27-28, 2002, Lecture Notes in Computer Science 2512 Springer 2002, pp: 225-236
- [Plastic08] PLASTIC consortium. “A B3G Service Platform: The IST PLASTIC Project”. PLASTIC White Paper, <http://plastic.paris-rocquencourt.inria.fr/plasticwhitepaper.pdf>, 2008.
- [PlasticD1.2] PLASTIC consortium. PLASTIC Deliverable D1.2: “Formal description of the PLASTIC conceptual model and of its relationship with the PLASTIC platform toolset”. http://wwwc.inria.fr/plastic/deliverables/plastic-d1_2-final.pdf/download.
- [PlasticD2.1] PLASTIC consortium. PLASTIC Deliverable D2.1: “SLA language and analysis techniques for adaptable and resource-aware components”. http://wwwc.inria.fr/plastic/deliverables/plastic-d2_1-finalpdf.pdf/download.
- [PlasticD2.2] PLASTIC consortium. PLASTIC Deliverable D2.2: “Graphical design language and tools for resource-aware adaptable components and services”. http://wwwc.inria.fr/plastic/deliverables/plastic-d2_2-finalpdf.pdf/download.
- [PlasticD2.3] PLASTIC consortium. PLASTIC Deliverable D2.3: “Development process and tools: assessment and revision”. http://wwwc.inria.fr/plastic/deliverables/plastic-d2_3-finalpdf.pdf/download
- [PlasticWS] PLASTIC consortium. The PLATIC web site: <http://plastic.paris-rocquencourt.inria.fr/>
- [Pnu84] A. Pnueli. In transition from global to modular temporal reasoning about programs. Logics and Models of Concurrent Systems. Springer, 1984

- [Pon03] S.Ponnekanti. Application-Service Interoperation Without Standardized Service Interfaces. In Proceedings of the 1st IEEE International Conference on Pervasive Computing and Communications, 2003.
- [PSOnc] Progress Sonic ESB, <http://web.progress.com/en/sonic/sonic-esb.html>
- [PTDL07] M. P. Papazoglou, P. Traverso, S. Dustdar and F. Leymann, Service-Oriented Computing: State of the Art and Research Challenges, in proceedings of IEEE Computer Journal.
- [PW05] F. Puhlmann and M. Weske. Using the pi-calculus for formalizing workflow patterns. In In Proceedings 3rd International Conference on Business Process Management (BPM), volume 3649, pages 153–168, Nancy, France, 2005. Springer Verlag.
- [PZL08] C. Pautasso, O. Zimmermann and F. Leymann. RESTful Web Services vs. “Big” Web Services: Making the Right Architectural Decision. In Proceedings of the International World Wide Web Conference (WWW), 2008.
- [QZCY07] Z. Qiu, X. Zhao, C. Cai, and H. Yang. Towards the theoretical foundation of choreography. In WWW '07: Proceedings of the 16th international conference on World Wide Web, pages 973–982, New York, NY, USA, 2007. ACM.
- [RBK05] M. Randic, B. Blaskovic, and P. Knezevic, Modeling Service Dependencies in Ad Hoc Collaborative Systems. EUROCON 2005, 1-4244-0049-X, pp 1842-1845
- [RDGM+08] J. Rellermeyer, M. Duller, K. Gilmer, D. Maragkos, D. Papageorgiou, G. Alonso. The Software Fabric for the Internet of Things. Internet of Things 2008, International Conference for Industry and Academia. March 26-28, 2008. Zurich.
- [Registry] Oracle Enterprise Registry Oracle & Sun 2010 available at <http://www.oracle.com/us/technologies/soa/service-registry-066462.html>
- [REMPD07] Rosenberg, F., Enzi, C., Michlmayr, A., Platzer, C., and Dustdar, S. 2007. Integrating Quality of Service Aspects in Top-Down Business Process Development Using WS-CDL and WS-BPEL. In Proceedings of the 11th IEEE international Enterprise Distributed Object Computing Conference (October 15 - 19, 2007). EDOC. IEEE Computer Society.
- [Repository] Oracle Enterprise Repository Oracle & Sun 2010 available at <http://www.oracle.com/us/technologies/soa/enterprise-repository-066456.html>
- [RKL09] M. Rambold, H. Kasinger, F. Lautenbacher and B. Bauer. Towards Autonomic Service Discovery A Survey and Comparison. In Proceedings of the IEEE SCC, 2009.
- [RRA08] J. S. Rellermeyer, O. Riva, and G. Alonso. AlfredO: An Architecture for Flexible Interaction with Electronic Devices. In Proceedings of the 9th International Middleware Conference (Middleware'08), December 1-5 2008.
- [RRLC+06] P-G. Raverdy, O. Riva, A. de La Chapelle, R. Chibout, V. Issarny. Efficient Context-aware Service Discovery in Multi-Protocol Pervasive Environments. In Proceedings of the 7th International Conference on Mobile Data Management (MDM), 2006.
- [RS04] J. Rao and X. Su, “A Survey of Automated Web Service Composition Methods,” in SWSWPC 2004: Proceedings of the First International Workshop on Semantic Web Services and Web Process Composition. San Diego, CA, USA: Springer, 2004, pp. 43-54
- [RSE08] F. Raimondi, J. Skene, and W. Emmerich. Efficient Online Monitoring of Web Service SLAs. In Proceedings of the 16th ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE), 2008.
- [RW89] P. J. Ramadge and W. M Wonham. The Control of Discrete Event Systems. Proceedings of the IEEE, 77(1), 1989.
- [Sal08] G. Salaun, Generation of Service Wrapper Protocols from Choreography Specifications, sefm, pp.313-322, 2008 Sixth IEEE International Conference on Software Engineering and Formal

- [Sat01] M. Satyanarayanan. Pervasive computing: Vision and challenges, IEEE Personal Communications 8 (4) (2001) 10–17.
- [SAVARA] Savara: Testable Architecture methodology. <http://www.jboss.org/savara>
- [SAWSDL] Semantic Annotations for WSDL and XML Schema <http://www.w3.org/2002/ws/sawSDL/spec/>
- [SBFW+04] T. Sivaharan, G. Blair, A. Friday, M. Wu, H. Duran-Limon, P. Okanda, and C.F. Sorensen. Cooperating Sentient Vehicles for Next Generation Automobiles. In Proceedings of ACM MobiSys International Workshop on Applications of Mobile Embedded Systems, 2004.
- [SBFZ07] J. Su, T. Bultan, X. Fu and X. Zhao. Towards a Theory of Web Service Choreographies. In Proc. of WS-FM'07 Proceedings of the 4th international conference on Web services and formal methods, volume 4937 of LNCS, pages 1--16. Springer-Verlag
- [SCMK07] H. Song, D. Cheng, A. Messer, S. Kalasapur. Web Service Discovery Using General-Purpose Search Engines. 2007 IEEE International Conference on Web Services (ICWS 2007), pp. 265-271, July 9-13, 2007, Salt Lake City, Utah, USA.
- [SeCSE] SeCSE, Project web site: <http://www.secse-project.eu/>
- [Sel03] B. Selic. "The Pragmatics of Model-driven Development". IEEE Software 20(5), 19–25 (2003).
- [SF07] N. Salatge and J-C. Fabre. Fault Tolerance Connectors for Unreliable Web Services. In Proceedings of the 37th IEEE/IFIP International Conference on Dependable Systems and Networks, 2007.
- [SH07] H. M. Sneed and S. Huang. The design and use of WSDL-test: a tool for testing web services. Journal of Software Maintenance and Evolution: Research and Practice, 19:297-314, 2007.
- [SHCS10] H. Song, G. Huang, F. Chauvel, and Y. Sun. Applying MDE Tools at Runtime: Experiments upon Runtime Models. In proceedings of the 5th Workshop on Models@run.time at MODELS 2010 Oslo, Norway, October 5th 2010.
- [SHLP05] M.-T. Schmidt, B. Hutchison, P. Lambros, R. Phippen. The Enterprise Service Bus: Making service-oriented architecture real. IBM SYSTEMS JOURNAL, VOL 44, NO 4, 2005
- [SHP03] E. Sirin, J. Hendler, and B. Parsia. Semi-automatic composition of web services using semantic descriptions. In Proceedings of Web Services: Modeling, Architecture and Infrastructure workshop in conjunction with ICEIS2003, 2003.
- [SHWS05] P. Sawyer, J. Hutchinson, J. Walkerdine and I. Sommerville, Faceted Service Specification, Proceedings SOCCER (Service-Oriented Computing: Consequences for Engineering Requirements) Workshop, at RE'05 Conference, Paris, August 2005.
- [SIV08] T.G.J. Schepers, M.E. Iacob and P.A.T. Van Eck, A lifecycle approach to SOA governance, in Proceedings of the 25th Symposium On Applied Computing, SAC'08, 2008, March 16-20, pages 1055-1061.
- [SJSJ05] N. Sangal, E. Jordan, V. Sinha, and D. Jackson, D. Using Dependency Models to Manage Complex Software Architecture. In Proceedings of the 20th Annual ACM SIGPLAN Conference on Object-Oriented Programming, Systems, Languages, and Applications (OOPSLA), 2005
- [SKGF+10] F. J, da Silva e Silva, F. Kon, A. Goldman, M. Finger, R. Y. de Camargo, F. Castor Filho, F. M. Costa. "Application execution management on the InteGrade opportunistic grid middleware" in Journal of Parallel and Distributed Computing, v. 70(5), pp. 573-583. Elsevier. May, 2010.
- [SKWC08] A. Senart, M. Karpinski, M. Wieckowski and V. Cahill. Using Sensor Networks for Pedestrian Detection. Fifth IEEE Consumer Communications and Networking Conference (CCNC'2008), Las Vegas, Nevada, 2008.

- [SM05] A. Sassen and C. Macmillan. The Service engineering area: An overview of its current state and a vision of its future. Technical report, EU Commis., July 2005.
- [Smart] The smart project. <http://www.cs.ucr.edu/~ciardo/SMART/>.
- [SMIX] Apache ServiceMix, <http://servicemix.apache.org/home.html>
- [SOA4All] SOA4All, Project web site: <http://www.soa4all.eu/>
- [SoaML] <http://www.omg.org/spec/SoaML/>
- [SOAP] SOAP Version 1.2 <http://www.w3.org/TR/2007/REC-soap12-part0-20070427/>
- [SOAPUI] SoapUI. Web Services Functional Testing Tool. <http://www.soapui.org>.
- [Som10] I. Sommerville. "Software Engineering". Addison Wesley, ISBN: 978-0137035151, 9th Edition(March 13, 2010).
- [Sonic] Sonic ESB Progress Software Corporation 2010 available at <http://web.progress.com/en/sonic/sonic-esb.html>
- [SPS04] N. Srinivasan, M. Paolucci, K. P. Sycara. An Efficient Algorithm for OWL-S Based Semantic Search in UDDI. Revised Selected Papers of the First International Workshop on Semantic Web Services and Web Process Composition (SWSWPC 2004), San Diego, CA, USA, July 6, 2004, Lecture Notes in Computer Science 3387 Springer 2004, pp. 96-110.
- [SR09] G. Salaün, N. Roohi. On Realizability and Dynamic Reconfiguration of Choreographies, WASELF-'09. 2009.
- [SRAI10] I. Silva-Lepe, I. Rouvellou, R. Akolkar, and A. Iyengar. Seamless cross-domain connectivity for enabling domain autonomy in a federated soa. In ICWS '10: Proceedings of the 2010 IEEE International Conference on Web Services, pages 669–670, Washington, DC, USA, 2010 IEEE Computer Society
- [SS10] G. Spanoudakis and A. Sisman. Discovering Services During Service-Based System Design Using UML. IEEE Transactions on Software Engineering, 36(3):371-389, 2010.
- [STDD07] G. van Seghbroeck, F. de Turck, B. Dhoedt, P. Demeester. Web Service Choreography Conformance Verification in M2M Systems through the piX-model, perser, pp.385-390, IEEE International Conference on Pervasive Services, 2007
- [Sti08] V. Stirbu. 2008. Towards a RESTful Plug and Play Experience in the Web of Things. In Proceedings of the 2008 IEEE International Conference on Semantic Computing (ICSC '08). IEEE Computer Society, Washington, DC, USA, 512-517.
- [SUPER] SUPER, Project web site: <http://www.ip-super.org/>
- [SVS04] K. Sivashanmugam, K. Verma and A. Sheth. Discovery of Web Services in a Federated Registry Environment. In Proceedings of the IEEE International Conference on Web Services (ICWS), 2004.
- [SVSM03] K. Sivashanmugam, K. Verma, A. P. Sheth, J. A. Miller. Adding Semantics to Web Services Standards. Proceedings of the International Conference on Web Services, ICWS '03, pp: 395-401, June 23 - 26, 2003, Las Vegas, Nevada, USA.
- [SWHA+05] L. Schubert, M. Wilson, J. Haller, A. Arenas, A. Svirskas, P. Giambiagi, J. Doser, E. Lupo. The TrustCoM Conceptual Models. 2005
- [TFGG07] M. Tivoli, P. Fradet, A. Girault, and G. Goessler. Adaptor synthesis for real-time components, In proceedings of the International Conference on Tools and Algorithms for the Construction and Analysis of Systems (TACAS 2007), member of the European Joint Conferences on Theory and Practice of Software (ETAPS 2007). Braga, Portugal. 24 March - 1 April 2007.

Volume: LNCS 4424. Pages: 185-200. Year: 2007. ISBN: 978-3-540-71208-4. Publisher: Springer-Verlag Berlin Heidelberg

- [Thi06] C. Thilloy. SOA in the Enterprise: A Survey of the Technical Landscape SOA Magazine Issue I: September/October 2006 August 28, 2006.
- [TII08] M.Tivoli and P.Inverardi. Failure-free coordinators synthesis for component-based architectures, *Science of Computer Programming*, 71(3), pp. 181-212, May 2008. DOI REF: <http://dx.doi.org/10.1016/j.scico.2008.03.001>
- [Tivoli] Tivoli Management Framework IBM available at <http://www-01.ibm.com/software/tivoli/products/mgt-framework/>
- [TMD09] R. N. Taylor, N. Medvidovic, and E. M. Dashofy. *Software Architecture: Foundations, Theory, and Practice*. John Wiley & Sons, January 2009.
- [TPCY+03] W. T. Tsai, R. Paul, Z. Cao, L. Yu, and A. Saimi, "Verification of web services using an enhanced UDDI server, in *Proceedings of the 8th International Workshop on Object-Oriented Real-Time Dependable Systems (WORDS 2003)*, pp. 131–138, Turku, Finland, Jan. 2003
- [TRE10] A. Taherkordi, R. Rouvoy, and F. Eliassen, A Component-based Approach for Service Distribution in Sensor Networks, *International workshop on Middleware Tools, Services and Run-Time Support for Sensor Networks (MidSens'10)*, Colocated with *Middleware 2010*, November 2010
- [UDDI] OASIS group. Universal Description, Discovery and Integration (UDDI), version 3.0.1. Technical Committee Specification. October 2003. Available at <http://uddi.xml.org/> and also, http://uddi.org/pubs/uddi_v3.htm
- [ULS06] Ultra-Large-Scale Systems the software challenge of the Future. Carnegie Mellon. Software engineering institute. 2006
- [UML 2.0] Unified Modelling Language 2.2 <http://www.omg.org/spec/UML/2.2/>
- [UMM] UN/CEFACT. UN/CEFACT's Modeling Methodology (UMM), UMM Meta Model - Foundation Module, Mar. 2006. Technical Specification V1.0
- [UPDM] UPDM Request for Proposals (RFP) <http://www.updm.com/index.htm>, http://www.omg.org/techprocess/meetings/schedule/UPDM_RFP.html, <http://www.omg.org/cgi-bin/doc?dtc/2005-09-12>
- [UPSS+07] B. Urgaonkar, G. Pacifici, P. Shenoy, M. Spreitzer, and A. Tantawi. Analytic modeling of multitier internet applications. *ACM Trans. Web*, 1(1):2, 2007.
- [VL08] J. I. Vazquez, D. Lopez-de-Ipina. Social Devices: Autonomous Artifacts that Communicate on the Internet. *Internet of Things 2008, International Conference for Industry and Academia*. March 26-28, 2008. Zurich.
- [Vou08] M.A. Vouk. Cloud Computing, Issues Research and Implementation. In *Proceedings of the 30th International Conference on Information Technology Interfaces*, 2008
- [VPB09] C. Vecchiola, S. Pandey, and R. Buyya. High-performance cloud computing: A view of scientific applications. In *ISPAN*, pages 4–16, 2009.
- [Wal00] R. J. Waldinger, "Web Agents Cooperating Deductively," in *FAABS '00: Proceedings of the First International Workshop on Formal Approaches to Agent-Based Systems-Revised Papers*. Greenbelt, MD, USA: Springer-Verlag, 2000, pp. 250–262
- [WBSPH] IBM Websphere ESB, <http://www-01.ibm.com/software/integration/wsesb/>
- [WebSphere] IBM WebSphere Service Registry and Repository available at <http://www-01.ibm.com/software/integration/wsrr/>

- [WMV03] L. A. Williams, E. M. Maximilien, M. A. Vouk. "Test-Driven Development as a Defect-Reduction Practice" in 14th International Symposium on Software Reliability Engineering. IEEE Computer Society, 2003.
- [WR04] Peter Weill, Jeanne W. Ross. IT governance: how top performers manage IT decision rights for superior results. Harvard Business School Press, 2004.
- [WRGM08] A. Wegmann, G. Regev, G.A. Garret, and F. Marechal. Specifying Services for ITIL Service Management, International Workshop on Service-Oriented Computing Consequences for Engineering Requirements (SOCCER'08) in the 16th IEEE International Requirements Engineering Conference (RE'08), 2008.
- [WSA] Web Services Architecture. W3C, 2004.
- [WSBC04] K. Whitehouse, C. Sharp, E. Brewer, and D. Culler. 2004. Hood: a neighborhood abstraction for sensor networks. In Proceedings of the 2nd international conference on Mobile systems, applications, and services (MobiSys '04). ACM, New York, NY, USA, 99-110.
- [WS-BPEL] Business Process Execution Language for Web Services Version 2.0 <http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html>
- [WSCDL] Web Services Choreography Description Language Version 1.0 <http://www.w3.org/TR/ws-cdl-10/>
- [WSCl] Web Service Choreography Interface Version 1.0 <http://www.w3.org/TR/wsci/>
- [WSDL] Web Services Description Language (WSDL) Version 1.2 <http://www.w3.org/TR/2003/WD-wsdl12-20030611/>
- [WSDL-S] Web Service Semantics - WSDL-S <http://www.w3.org/Submission/WSDL-S/>
- [WSFL] Web Service Flow Language Version 1.0.
- [WSO2] WSO2 Governance Registry available at <http://wso2.com/products/governance-registry> WSO2 2010
- [Wu09] F. Wu. Mapping interconnection choreography to interaction choreography models. Master thesis, 2009.
- [WW10] R. Weber and R. Weber. Internet of Things. Springer, 2010.
- [XLANG] XLANG Web Services for Business Process Design.
- [Yeu08] W.L. Yeung, "A Formal Basis for Cross-Checking ebXML BPSS Choreography and Web Service Orchestration," apsc, pp.524-529, 2008 IEEE Asia-Pacific Services Computing Conference, 2008.
- [YG02] Y. Yao and J. Gehrke. 2002. The cougar approach to in-network query processing in sensor networks. SIGMOD Rec. 31, 3 (September 2002), 9-18.
- [YLG07] U. Yildiz, G. Lippmann, C. Godart Towards decentralized service orchestrations Symposium on Applied Computing Proceedings of the 2007 ACM symposium on Applied computing Seoul, Korea SESSION: Web technologies Pages: 1662 - 1666, 2007 ISBN:1-59593-480-4
- [YLS06] Y. Yuan, Z. Li, and W. Sun: A graph-search based approach to BPEL4WS test generation, in ICSEA'06 Proceedings of the International Conference on Software Engineering Advances, Tahiti, French Polynesia, Oct. 2006, IEEE Computer Society.
- [YM94] E. Yu and J.M. Mylopoulos. Understanding "Why" in Software Process Modelling, Analysis and Design. Proceedings 16th International Conference on Software Engineering, IEEE Computer Society Press, 159-168, 1994.

- [YS97] D. M. Yellin and R. E. Strom. Protocol Specifications and Component Adaptors. ACM Trans. Program. Lang. Syst. 19(2), 1997
- [YZCQ08] H. Yang, X. Zhao, C. Cai, Z. Qiu. Model-Checking of Web Services Choreography, pp.79-84, 2008 IEEE International Symposium on Service-Oriented System Engineering, 2008
- [Zar00] A. Zarras. Systematic Customization of Middleware. PhD Thesis, University of Rennes I, 2000.
- [Zar04] A. Zarras. Online Upgrade of Object-Oriented Middleware. Journal of Object Technology 3(7), 2004.
- [ZB09] G. Zheng and A. Bouguettaya. Service Mining on the Web. IEEE Transactions on Services Computing, 2(1):65-78, 2009.
- [ZBDH06] J. M. Zaha, A. Barros, M. Dumas, A. terHofstede. Let's Dance: A Language for Service Behavior Modeling. In Proceedings of the 14th International Conference on Cooperative Information Systems (CoopIS), 2006
- [ZBND+04] L. Zeng, B. Benatallah, A.H.H. Ngu, M. Dumas, J. Kalagnanam, and H. Chang. QoS-Aware Middleware for Web Services Composition. IEEE Transactions on Software Engineering 30(5), 311-327, 2004.
- [ZC06] J. Zhang and B. H. C. Cheng. Model-Based Development of Dynamically Adaptive Software. Proc. 28th Int'l Conf. Software Eng. (ICSE06). ACM Press. 2006.
- [ZCB10] Q. Zhang, L. Cheng and R. Boutaba. "Cloud computing: state-of-the-art and research challenges" in Journal of Internet Services and Applications, 1(1):7-18, Springer London, 2010
- [ZCL04] C. Zhou, L.-T. Chia and B.-S. Lee, "DAML-QoS Ontology for Web Services", ICWS, pp 472-479, 2004
- [ZDS08] K. Zachos, G. Dobson and P. Sawyer. Ontology-aided Translation in the Comparison of Candidate Service Quality. Proceedings SOCCER workshop, IEEE Computer Society Press, 2008
- [ZL10] Z. Zheng, M. R. Lyu: Collaborative reliability prediction of service-oriented system, Proceedings of the 32nd ACM/IEEE International Conference on Software Engineering ICSE'10.
- [ZLMZ+08] P. Zhang, B. Li, H. Muccini, Y. Zhou, M. Sun. Data-Enriched Modeling and Verification of WS-CDL Based on UML Models, icws, pp.752-753, 2008 IEEE International Conference on Web Services, 2008
- [ZM08] K. Zachos and N.A.M. Maiden, Inventing Requirements from Software: An Empirical Investigation with Web Services, Proceedings 16th IEEE International Conference on Requirements Engineering, IEEE Computer Society Press, 145-154.
- [ZME06] S. Zachariadis, C. Mascolo and W. Emmerich (2006). The SATIN Component System - A Meta Model For Engineering Adaptable Mobile Systems. IEEE Transactions on Software Engineering, 32(11):910-927.
- [ZMZJ07] K. Zachos, N.A.M. Maiden, X. Zhu X. and S. Jones, Discovering Web Services To Specify More Complete System Requirements, Proceedings CaiSE'2007, Springer-Verlag Lecture Notes on Computer Science LNCS 4495, 142-157
- [ZPPN+07] J. Zhou, D. Pakkala, J. Perala, E. Niemela, J. Riekkki, M. Ylianttila, Dependency-aware Service Oriented Architecture and Service Composition, IEEE International Conference on Web Services 2007, pp. 1146-1149
- [ZSD08] A. Zisman, G. Spanoudakis, and J. Dooley. A Framework for Dynamic Service Discovery. In Proceedings of the 23rd IEEE/ACM International Conference on Automated Software Engineering (ASE), 2008.

[ZZL09]

Y. Zhai, J. Zhang, K-J. Lin. SOA Middleware Support for Service Process Reconfiguration with End-to-End QoS Constraints. In Proceedings of the IEEE International Conference on Web Services (ICWS), 2009.