Athman Bouguettaya · Quan Z. Sheng
Florian Daniel *Editors*

Web Services Foundations

Springer

Web Services Foundations

Athman Bouguettaya · Quan Z. Sheng
Florian Daniel
Editors

Web Services Foundations

Foreword by Michael P. Papazoglou

Springer

Editors
Athman Bouguettaya
School of Computer Science
and Information Technology
RMIT University
Melbourne, VIC
Australia

Quan Z. Sheng
School of Computer Science
University of Adelaide
Adelaide, SA
Australia

Florian Daniel
Dipartimento di Ingegneria e Scienza
dell'Informazione
Università di Trento
Povo, Trento
Italy

ISBN 978-1-4614-7517-0 ISBN 978-1-4614-7518-7 (eBook)
DOI 10.1007/978-1-4614-7518-7
Springer New York Heidelberg Dordrecht London

Library of Congress Control Number: 2013946213

Printed on acid-free paper

Springer is part of Springer Science+Business Media (www.springer.com)

To my parents, Horia and Mahmoud, and my wife Malika

Athman Bouguettaya

To my parents Shuilian and Jianwu, my brothers Guanzheng and Xinzheng, my wife Yaping and my daughters Fiona and Phoebe

Quan Z. Sheng

To Cinzia, my family, my friends

Florian Daniel

Foreword

Service-Oriented Computing (SOC) is the computing paradigm that utilizes software services as fundamental elements for developing and deploying distributed software applications. Services are self-describing, platform-agnostic computational elements that support rapid, low-cost composition of distributed applications. They perform functions, which can be anything from simple requests to complicated business processes. Services allow organizations to expose their core competencies programmatically via a self-describing interface based on open standards over the Internet (or intranet) using standard (XML-based) languages and protocols. Because services provide a uniform and ubiquitous information distributor for wide range of computing devices (such as handheld computers, PDAs, cellular telephones, or appliances) and software platforms (e.g., UNIX or Windows), they constitute a major transition in distributed computing.

A Web service is a specific kind of service that is identified by a URI that exposes its features programmatically over the Internet using standard Internet languages and protocols, and can be implemented via a self-describing interface based on open Internet standards (e.g., XML interfaces which are published in network-based repositories).

Understanding the conceptual underpinnings and mastering the technical intricacies of Web services is anything but trivial and is absolutely necessary to construct a well-functioning service-based system or application. Web service technology is undergoing continuous, rapid evolution, thanks to both standardization efforts pushed forward by industry and the research efforts of the scientific community.

Web services standards are still evolving. However, they seem to converge today on a handful of standards: the Simple Object Access Protocol (SOAP) for service communication, Web Services Description Language (WSDL) for service description, Universal Description, Discovery, and Integration Infrastructure (UDDI) for registering and discovering services, and the Business Process Execution Language (BPEL) for service composition. A plethora of WS-* specifications also exists to describe the full spectrum of activities related to Web services in topics such as reliable messaging, security, privacy, policies, event processing, and coordination, to name but a few.

Leading international conferences, such as the International Conference on Service-Oriented Computing (ICSOC), the International Conference on Web Services (ICWS), the International Conference on Service Computing (SCC), and others, have spearheaded groundbreaking research efforts. This has led to the emergence of novel topics such as semantic Web services, automated Web service composition, Web service recommendations, quality of service, trust, and a range of other interesting themes. Related conference series such as Web Engineering, Cloud Computing, Business Process Management, HCI, and Database related conferences, have all been strongly influenced by the emergence of Web services and consistently feature Web service related topics in their calls for papers. These conferences contribute to the wealth of knowledge that is growing exponentially around Web services.

The content of this book and that of its companion book *Advanced Web Services* (Springer, 2013) reflect such activities. It is a testimonial of the leading role of its editors and their highly influential work in the area of Web services. Together, both books cover an enormous wealth of important topics and technologies that mirror the evolution of Web services. They provide an exhaustive overview of the challenges and solutions of all major achievements pertaining to Web services. Each chapter is an authoritative piece of work that synthesizes all pertinent literature and highlights important accomplishments and advances in its subject matter.

To my knowledge, this is the first attempt of its kind, providing complete coverage of the key subjects in Web services. I am not aware of any other book that is as thorough, comprehensive and ambitious in explaining the current state of the art of scientific research and in synthesizing the perspectives and know-how of so many experts in the field. Both books are a must-read for everyone interested in the field. They cater for the needs of both novices to the field as well as seasoned researchers and practitioners. They are a major step in this field's maturation and will serve to unify, advance, and challenge the scientific community in many important ways.

It is a real pleasure to have been asked to provide the foreword for this book collection. I am happy to commend the editors and authors on their accomplishment, and to inform the readers that they are looking at a landmark in the development of the Web services field. Anybody serious about Web services ought to have handy a copy of *Web Services Foundations* and *Advanced Web Services* in their private library!

Tilburg, The Netherlands Michael P. Papazoglou
December 2012

Preface

Web Service technology is undeniably the preferred delivery method for the Service-Oriented Computing (SOC) paradigm. It has evolved over the years to be a comprehensive, interdisciplinary approach to modern software development. Web services have gone beyond software componentization technology to embody and express the software manifestation of a general trend transforming our modern society from an industrial, production-centric economy into a digital, service-centric economy. Web services aim to provide the missing conceptual links that unify a variety of different disciplines, such as networking, distributed systems, cloud computing, autonomic computing, data and knowledge management, knowledge-based systems, and business process management. Web services are the technological proxies of services that power much of the developed and increasingly developing economies. In this respect, Web services play a central role in enabling and sustaining the growth of service-centric economies and help modernizing organizations, companies and institutions also from an IT perspective.

Over the last decade, Web services have become a thriving area of research and academic endeavors. Yet, despite a substantial body of research and scientific publications, the Web services community has been hitherto missing a one stop-shop that would provide a consolidated understanding of the scientific and technical progress of this important subject. This book (the second of a two-book collection) is a serious attempt to fill this gap and serve as a primary point of reference reflecting the pervasive nature of Web services.

This book is the first installment of a two-book collection (we discuss the advanced topics in the second book, *Advanced Web Services*, Springer, 2013). Together, they comprise approximately 1,400 pages covering state-of-the-art theoretical and practical aspects as well as experience using and deploying Web services. The collection offers a comprehensive overview of the scientific and technical progress in Web services technologies, design, architectures, applications, and performance. The first book of the collection consists of two major parts:

I Foundations of Web Services (12 chapters)—It explores the most representative theoretical and practical approaches to Web services, with a special focus on the general state-of-the-art approaches to Web service composition;
II Service Selection and Assisted Composition (16 chapters)—It focuses on other aspects of Web service composition problem, specifically takes a deep look at non-functional aspects (e.g., quality of service), Web service recommendations, and how Web service composition is made easy for less expert developers.

The second book (*Advanced Web Services*, Springer, 2013) consists of three major parts:

I Advanced Services Engineering and Management (11 chapters)—It explores advanced engineering problems, such as Web service transactions and recovery, security and identity management, trust and contracts, and Web service evolution and management;
II Web Service Applications and Case Studies (5 chapters)—It covers concrete scenarios of the use of Web service technology and reports on empirical studies of real-world Web service ecosystems;
III Novel Perspectives and Future Directions (10 chapters)—It surveys approaches of the applications on how the Web service paradigm can be applied to novel contexts, such as human-centric computing, human work and the Internet of Things, and discusses the value of Web services in the context of mobile and cloud computing.

The topics covered in the collection are reflective of their intent: they aim to become the primary source for all pertinent information regarding Web service technologies, research, deployment and future directions. The purpose of the two books is to serve as a trusted and valuable reference point to researchers and educators who are working in the area of Web services, to students who wish to learn about this important research and development area, and to practitioners who are using Web services and the service paradigm daily in their software development projects.

This collection is the result of an enormous community effort, and their production involved more than 100 authors, consisting of the worlds leading experts in this field. We would like to thank the authors for their high-quality contributions and the reviewers for their time and professional expertise. All contributions have undergone a rigorous review process, involving three independent experts in two rounds of review. We are also very grateful to Springer for their continuous help and assistance.

Melbourne, Australia, December 2012 — Athman Bouguettaya
Adelaide, Australia — Quan Z. Sheng
Trento, Italy — Florian Daniel

Contents

Contributors

Dhaminda B. Abeywickrama Faculty of IT, Monash University, Clayton Campus, Wellington Road, Clayton, VIC 3800, Australia, e-mail: dhaminda.abeywickrama@gmail.com

Mohammed AbuJarour SAP AG, Potsdam, Germany, e-mail: mohammed.abujarour@sap.com

Amal Alhosban Wayne State University, Detroit, MI, USA, e-mail: ea1179@wayne.edu

Mohsen Asadi Simon Fraser University, Burnaby, BC, Canada, e-mail: masadi@sfu.ca

Ahmed Awad Faculty of Computers and Information, Cairo University, Giza, Egypt, e-mail: a.gaafar@fci-cu.edu.eg

Boualem Benatallah CSE, University of New South Wales, Sydney, NSW, Australia, e-mail: boualem@cse.unsw.edu.au

Salima Benbernou Université Paris Descartes, Paris, France, e-mail: salima.benbernou@parisdescartes.fr

Swapan Bhattacharya Jadavpur University, Kolkata, India, e-mail: bswapan2000@yahoo.co.in

M. Brian Blake Graduate School, University of Miami, Coral Gables, FL 33124-3220, USA, e-mail: m.brian.blake@miami.edu

Steffen Bleul Munich, Germany, e-mail: stbleul@gmx.de

Fabio Casati University of Trento, Via Sommarive 5, 38123 Trento, Italy, e-mail: casati@disi.unitn.it

Jayeeta Chanda BPPIMT, 137, VIP Road, Kolkata, India, e-mail: jayeeta.chanda@gmail.com

Min Chen Concordia University, Montreal, QC, Canada, e-mail: minchen2008halifax@yahoo.com

Xi Chen Schlumberger Technologies (Beijing) Ltd., Beijing, China, e-mail: bargittachen@gmail.com

Alan Colman Faculty of Information and Communication Technology, Swinburne University of Technology, Melbourne, VIC, Australia, e-mail: acolman@swin.edu.au

Florian Daniel University of Trento, Via Sommarive 5, Trento, Italy, e-mail: daniel@disi.unitn.it

Joseph G. Davis School of Information Technologies, University of Sydney, Sydney, NSW 2006, Australia, e-mail: joseph.davis@sydney.edu.au

Giuseppe De Giacomo Dipartimento di Ingegneria Informatica Automatica e Gestionale Antonio Ruberti, Sapienza Università di Roma, via Ariosto 25, 00185 Rome, Italy, e-mail: degiacomo@dis.uniroma1.it

Khalil Drira CNRS, LAAS, University of Toulouse, 7 avenue du colonel Roche, 31400 Toulouse, France, e-mail: khalil.drira@laas.fr

Zaiwen Feng State Key Laboratory of Software Engineering, School of Computer, Wuhan University, Wuhan, China, e-mail: fengzaiwen@whu.edu.cn

Dragan Gašević Athabasca University, Athabasca, AB, Canada; Simon Fraser University, Burnaby, BC, Canada, e-mail: dgasevic@acm.org

Claude Godart LORIA-INRIA-UMR 7503, 54506 Vandoeuvre-les-Nancy, France, e-mail: claude.godart@loria.fr

Nawal Guermouche CNRS, LAAS, University of Toulouse, 7 avenue du colonel Roche, 31400 Toulouse, France, e-mail: nawal.guermouche@laas.fr

Claudio Guidi Dipartimento di Matematica, University of Padua, Via Trieste 63, 35121 Padua, Italy, e-mail: cguidi@math.unipd.it

Riadh Ben Halima ReDCAD, University of Sfax, 3038 Sfax, Tunisia, e-mail: riadh.benhalima@enis.rnu.tn

Jun Han Faculty of Information and Communication Technology, Swinburne University of Technology, Melbourne, VIC, Australia, e-mail: jhan@swin.edu.au

Khayyam Hashmi Wayne State University, Detroit, MI, USA, e-mail: eh2304@wayne.edu

Keqing He State Key Laboratory of Software Engineering, School of Computer, Wuhan University, Wuhan, China, e-mail: hekeqing@whu.edu.cn

Patrick C. K. Hung University of Ontario Institute of Technology, Oshawa, ON, Canada, e-mail: patrick.hung@uoit.ca

Vu Hung University of New South Wales, Sydney, NSW, Australia, e-mail: vthung@gmail.com

Fuyuki Ishikawa National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda-ku, Tokyo 101-8430, Japan, e-mail: f-ishikawa@nii.ac.jp

Ananya Kanjilal BPPIMT, 137, VIP Road, Kolkata, India, e-mail: ag_k@rediffmail.com

Georgia Kleanthous Manchester Centre for Service Research, University of Manchester, Manchester M60 1QD, UK, e-mail: georgia.kleanthous@gmail.com

Woralak Kongdenfha ECPE, Naresuan University, Phitsanulok, Thailand, e-mail: woralakk@gmail.com

Freddy Lecue IBM Research, Dublin, Ireland, e-mail: freddy.lecue@ie.ibm.com

Jaejoon Lee Lancaster University, Lancashire, UK, e-mail: j.lee@comp.lancs.ac.uk

Angel Lagares Lemos University of New South Wales, Sydney, NSW, Australia, e-mail: angell@cse.unsw.edu.au

Niels Lohmann University of Rostock, Rostock, Germany, e-mail: niels.lohmann@uni-rostock.de

Michael R. Lyu Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong, China, e-mail: lyu@cse.cuhk.edu.hk

Zaki Malik Department of Computer Science, Wayne State University, Detroit, MI, USA, e-mail: zaki@cs.wayne.edu

Massimo Mecella Dipartimento di Ingegneria Informatica Automatica e Gestionale Antonio Ruberti, Sapienza Università di Roma, via Ariosto 25, 00185 Rome, Italy, e-mail: mecella@dis.uniroma1.it

Brahim Medjahed Department of Computer and Information Science, University of Michigan, Dearborn, MI, USA, e-mail: brahim@umich.edu

Nikolay Mehandjiev Manchester Centre for Service Research, University of Manchester, Manchester M60 1QD, UK, e-mail: n.mehandjiev@manchester.ac.uk

Emna Mezghani CNRS, LAAS, University of Toulouse, 7 avenue du colonel Roche, 31400 Toulouse, France; ReDCAD, University of Sfax, 3038 Sfax, Tunisia, e-mail: emna.mezghani@laas.fr

Mahboobeh Moghaddam School of Information Technologies, University of Sydney, Sydney, NSW 2015, Australia; National ICT Australia, Australian Technology Park, Sydney, NSW, Australia, e-mail: mahboobe@it.usyd.edu.au

Bardia Mohabbati Simon Fraser University, Burnaby, BC, Canada, e-mail: mohabbati@sfu.ca

Fabrizio Montesi IT University of Copenhagen, Rued Langgaards Vej 7, 2300 Copenhagen, Denmark, e-mail: fmontesi@itu.dk

Hamid R. Motahari-Nezhad HP Labs, Palo Alto, CA, USA, e-mail: hamid-reza.motahari-nezhad@hp.com

Abdallah Namoun Manchester Centre for Service Research, University of Manchester, Manchester M60 1QD, UK, e-mail: abdallah.namoune@mbs.ac.uk

Tuan Nguyen Faculty of Information and Communication Technology, Swinburne University of Technology, Melbourne, VIC, Australia, e-mail: tmnguyen@swin.edu.au

Fabio Patrizi Dipartimento di Ingegneria Informatica Automatica e Gestionale Antonio Ruberti, Sapienza Università di Roma, via Ariosto 25, 00185 Rome, Italy, e-mail: patrizi@dis.uniroma1.it

Cesare Pautasso Faculty of Informatics, University of Lugano, via Buffi 13, 6900 Lugano, Switzerland, e-mail: c.pautasso@ieee.org

Pascal Poizat LRI UMR CNRS, 8623 Orsay, France, e-mail: pascal.poizat@lri.fr

Carlos Rodríguez University of Trento, Via Sommarive 5, 38123 Trento, Italy, e-mail: crodriguez@disi.unitn.it

Soudip Roy Chowdhury University of Trento, Via Sommarive 5, 38123 Trento, Italy, e-mail: rchowdhury@disi.unitn.it

Regis Saint-Paul Oceanet Technology, Nantes, France, e-mail: regis.saintpaul@gmail.com

Sabnam Sengupta BPPIMT, 137, VIP Road, Kolkata, India, e-mail: sabnamsg@gmail.com

Stefano Soi University of Trento, Via Sommarive 5, 38123 Trento, Italy, e-mail: soi@disi.unitn.it

Christian Stahl Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, 5600 MB Eindhoven, 513, Eindhoven, The Netherlands, e-mail: C.Stahl@tue.nl

Wil M. P. van der Aalst Department of Mathematics and Computer Science, Technische Universiteit Eindhoven, 5600 MB Eindhoven, 513, Eindhoven, The Netherlands, e-mail: W.M.P.v.d.Aalst@tue.n

Usman Wajid Manchester Centre for Service Research, University of Manchester, Manchester M60 1QD, UK, e-mail: usman.wajid@manchester.ac.uk

Jian Wang State Key Laboratory of Software Engineering, School of Computer, Wuhan University, Wuhan, China, e-mail: jianwang@whu.edu.cn

Thomas Weise School of Computer Science and Technology, University of Science and Technology of China, 230027 Hefei, Anhui, China, e-mail: tweise@ustc.edu.cn

Karsten Wolf University of Rostock, Rostock, Germany, e-mail: karsten.wolf@uni-rostock.de

Hua Xiao IBM Canada Laboratory, Markham, ON, Canada, e-mail: huaxiao@ca.ibm.com

Yuhong Yan Concordia University, Montreal, QC, Canada, e-mail: yuhong@encs.concordia.ca

Jian Yang Macquarie University, Sydney, NSW 2109, Australia, e-mail: jian.yang@mq.edu.au

Qi Yu Rochester Institute of Technology, Rochester, NY, USA, e-mail: qi.yu@rit.edu

Gianluigi Zavattaro INRIA Focus Research Team, University of Bologna, Mura A. Zamboni 7, 40127 Bologna, Italy, e-mail: zavattar@cs.unibo.it

Jia Zhang Carnegie Mellon University, Silicon Valley, Mountain View, CA, USA, e-mail: jia.zhang@sv.cmu.edu

Liang-Jie Zhang Kingdee International Software Group Co. Ltd., Shenzhen, China, e-mail: zhanglj@ieee.org

Weiliang Zhao University of Wollongong, Wollongong, NSW 2522, Australia, e-mail: wzhao@uow.edu.au

Huiyuan Zheng Macquarie University, Sydney, NSW 2109, Australia, e-mail: huiyuan.zheng@mq.edu.au

Zibin Zheng Department of Computer Science and Engineering, The Chinese University of Hong Kong, Shatin, Hong Kong, China, e-mail: zbzheng@cse.cuhk.edu.hk

Ying Zou Department of Electrical and Computer Engineering, Queen's University, Kingston, ON, Canada, e-mail: ying.zou@queensu.ca

Part I
Foundations of Web Services

Chapter 1
Web Services and Business Processes: A Round Trip

Mohammed AbuJarour and Ahmed Awad

Abstract Service-oriented Architecture (SOA) is considered as an implementation for business processes (BP). However, the relation between SOA and BPs is usually inspected in one direction only. In this chapter, we investigate the *bi-directional* relation between web services and business processes, and explore potential benefits therefrom. In particular, we introduce a novel approach to generate additional information about web services based on the configurations of business processes that consume these web services. This information is then used to enhance and smooth the modeling and configuration of future business processes. Through our approach, we can generate three types of information from consumers' business processes, namely annotations, context, and relations among web services. To evaluate our approach, we use the SAP reference model and we show the results in this chapter.

1.1 The Relation Between Business Processes and Web Services

Service-oriented Architecture (SOA) has been considered as an implementation platform for business processes (BP), nevertheless, each of them is typically investigated separately. Investigating both worlds (i.e., SOA and BP) together is expected to result in several benefits for both SOA and BP communities. For instance, getting a running instance of a business process model requires mapping its *service* tasks to (web) services. This mapping step requires sufficient information about the used web

M. AbuJarour (✉)
SAP AG, Potsdam, Germany
e-mail: mohammed.abujarour@sap.com

A. Awad
Faculty of Computers and Information,
Cairo University, Giza, Egypt
e-mail: a.gaafar@fci-cu.edu.eg

A. Bouguettaya et al. (eds.), *Web Services Foundations*,
DOI: 10.1007/978-1-4614-7518-7_1,

services that is understandable by process engineers or business people who create such mappings. Finding candidate web services to execute each service task is one of the key challenges in SOA, e.g., due to poor service descriptions [10]. We consider the configurations of BPs as a rich source of information about their consuming web services that enhance service discovery and future BP configurations.

Due to the increasing number of BPs, service consumers in several application domains maintain repositories of business process models (BPM) for their daily activities, e.g., "ship ordered item". Each business process is composed of a set of manual (i.e., performed by employees) or service tasks (i.e., performed through web services). Building a new BPM incorporates three steps: (1) creating and labeling its tasks (2) determining the interaction between them, i.e., data and control flow (3) configuring the created model, i.e., *selecting* web services to perform service tasks. The built BPM is typically stored in the consumer's repository for future requests.

Using the aforementioned scenario in practice involves several challenges, such as, service discovery, service selection, BP configuration. *Service discovery* is one of the main challenges in Service-oriented Computing (SOC), where a list of candidate web services are returned to service consumer as a result for their queries. Choosing a particular web service to invoke from this list is known as *service selection* [18], which has become a complex task due to several factors, e.g., lack of rich service descriptions. Therefore, additional information about web services—e.g., annotations, relations among web services, etc.—is expected to help meet this challenge [7]. *Business process configuration* represents the service selection step, where each service task is assigned a web service to execute it. Performing this task dynamically requires sufficient information about web services so that process engineers configure their BPMs accordingly. Technical information only is not sufficient, because it does not fit their backgrounds and knowledge [19].

Using web services within distributed business processes brings the challenges of service discovery and selection to business processes. In this work, we introduce a novel approach to bridge both worlds (SOA and BPM), where we use business process configurations to derive additional information about their implementing web services that reflect service consumers' perspective (business view) to enrich their technical descriptions released by their providers. We are able to generate three types of information about web services, namely annotations, context, and relations among web services. Annotations for web services are generated from tasks' labels and documentations, whereas context is derived from models' titles and descriptions. We discover *additional* realistic and rich relations among web services in the form of linkage patterns using behavioral profiles [24] of their consuming business processes. Additionally, we derive a global representation of all relations among web services that are derived from multiple business processes. We use this global representation to predict relations among web services that are not used together in a single business process using the gained knowledge in this global representation.

The contributions of the work introduced in this chapter are:

1. Supporting smooth configuration of business processes by enabling context-aware service selection.

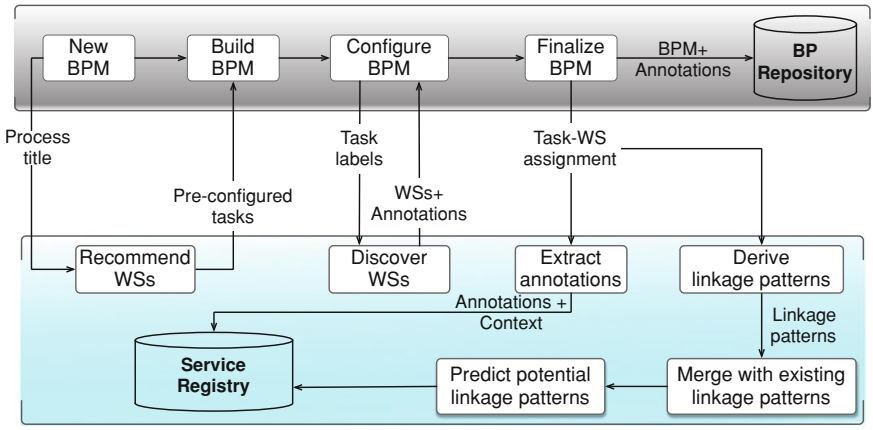

Fig. 1.1 An overview of our approach of integrating business processes and web services

2. Finding realistic, rich relations among web services as *linkage patterns.*
3. Disambiguating exclusive relations between web services using lexical ontologies, e.g., WordNet.
4. Merging behavioral profiles of BPs into a single global behavioral profile.
5. Revealing relations among web services that have not yet been used together.

The rest of this chapter is organized as follows: We give an overview of our approach in Sect. 1.2. Then, we introduce the fundamental concepts that are used throughout this chapter in Sect. 1.3. After that, we present our approach to generate annotations for web services from business process configurations in Sect. 1.4. In Sect. 1.5, we describe our approach to derive rich relations among web services in the form of linkage patterns. Deriving global behavioral profiles among web services is described in Sect. 1.6. Implementation details and experiments are introduced in Sect. 1.7. Related work is summarized in Sect. 1.8. We summarize this chapter in Sect. 1.9.

1.2 Overview of Our Approach

In this section we give an overview of our approach that represents a round trip between web services and business processes as shown in Fig. 1.1. The scenario starts when a business process designer creates a new BPM. At that point, the designer gives a descriptive name and summary for the new process, e.g., *establish a company in Germany*. Behind the scene, a request is sent to a service registry to find relevant web services that have been used in similar models, e.g., *establish a company in UK*. The returned recommended services are provided as *pre-configured* tasks that are made available to the designer to accelerate the process design.

The process designer might not use all web services recommended by the service registry in their new model. Therefore, they introduce new tasks to express the particular business needs in the process at hand. Each new task is given an identifying label that we pass to the service registry to find potential web services that can be candidate matches. For each new task in the model, a list of candidate web services is returned to the process designer during the configuration phase. Each web service in our collection is associated with a set of annotations that explain its functionality. These annotations are extracted and generated automatically from the websites of their providers [2], invocation analysis [3], and previous BP configurations. The new BPM is finalized when each service task is configured by assigning a web service to execute it. With the finalized BPM, two sources of information can be identified and generated, namely task-to-web service assignment and annotated BPM.

On the one hand, the *task-to-web service assignment* is passed from the modeling framework to the service registry, where annotations, context, and relations are generated therefrom. On the other hand, the tasks in the created BPM are automatically annotated with the annotations of the web services they are bound with, resulting in an *annotated BPM*. These annotations are crucial for BPM lookup, because not only task labels are used to index and find tasks, but enriched annotations are also used to achieve this goal [4] leading to better discovery of models from process repositories.

We use this assignment list also to generate behavioral profiles, based on which we derive rich relations among web services. Even more, we discover *hidden* relations among services that have not been used together in any business process configuration yet. The notion of behavioral profiles is developed by Weidlich et al. [24] to give a behavioral abstraction over business processes. We use this notion and extend it according to requirements for discovering relations among web services.

1.3 Fundamentals: Business Process Knowledge

Researchers have proposed several approaches that investigate the behavioral relations among tasks within a process model. For instance, the α-algorithm [22], causal footprints [23] and behavioral profiles [24].[1] Although these approaches are developed for different purposes, they have a fundamental common feature; generating a set of behavioral relations among tasks in a process model. We use these behavioral relations in our approach to derive rich relations among web services. Causal footprints [23] and behavioral profiles [24] take as input a process model, represented as a WF-net [21]. Whereas the α-algorithm requires as input a set of process execution traces (i.e., log). We use the behavioral profiles approach as a starting point, because we have process models as input and because the behavioral profile approach is much more efficient than causal footprints. Nevertheless, our approach is independent of this selection and works with process behavior abstraction approaches that support the fundamental behavioral relations.

[1] There are other related approaches that share similar underlying concepts.

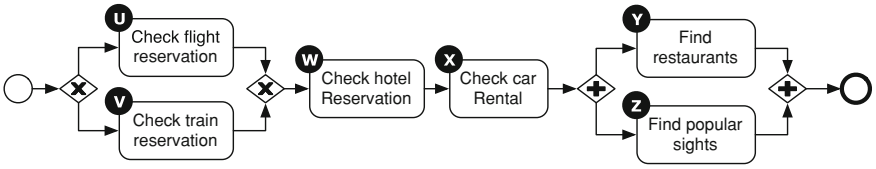

Fig. 1.2 A journey organizer business process modeled in BPMN 1.0

Behavioral profiles represent an abstract description of a business process and identifies the behavioral relationship between any pair of its nodes. This relationship can be: (1) strict order $\rightsquigarrow$, (2) concurrent $\|$, (3) exclusive #, or (4) inverse order $\leftsquigarrow$. The formal definition of behavioral profiles is introduced in Definition 1.1.

Definition 1.1 *(Behavioral Profile)* Let N be the set of nodes within a business process model. The *behavioral profile* of a business process model is a function $bhp: N \times N \rightarrow \{\rightsquigarrow, \leftsquigarrow, \|, \#\}$ that assigns a behavioral property, strict order, inverse order, parallel, or exclusive, between each pair of nodes within the business process model.

If two tasks a, b appear in strict order, $bhp(a, b) = \rightsquigarrow$, then task a always executes before task b. Similarly, if two tasks are concurrent then they can be executed in any order. Exclusiveness means that at most one of the two tasks can execute within a process instance. The behavioral profile of the BP shown in Fig. 1.2 includes several behavioral properties, such as: $bhp(U, V) = \#$, $bhp(W, X) = \rightsquigarrow$, $bhp(X, W) = \leftsquigarrow$, $bhp(Y, Z) = \|$, $bhp(U, X) = \rightsquigarrow$, $bhp(Y, V) = \leftsquigarrow$, etc.

The definition of behavioral profiles is not sufficient to achieve our goal of discovering fine-grained linkage patterns among web services, in particular assigning *weights* to the discovered linkage patterns. Therefore, we extend it by incorporating the shortest distance between each pair of tasks in addition to their behavioral property. The distance between a pair of tasks is calculated by counting the number of edges between them in the considered BPM. A preliminary definition of the extended behavioral profile is given in Definition 1.2. A comprehensive definition of the "Extended Behavioral Profile" is introduced in Definition 1.3.

Definition 1.2 *(Extended Behavioral Profile)* Let N be a set of nodes within a business process model. The *extended behavioral profile* of a business process model is a function $bhp': N \times N \rightarrow \{\rightsquigarrow, \leftsquigarrow, \|, \#\} \times \mathbb{N}$ that assigns a behavioral property, strict order, inverse order, parallel, or exclusive, and a *distance*, between each pair of nodes within the business process model.

For instance, the extended behavioral profile of the BP in Fig. 1.2 includes several pairs, such as: $bhp'(U, V) = (\#, 0)$, $bhp'(W, X) = (\rightsquigarrow, 1)$, $bhp'(X, W) = (\leftsquigarrow, 1)$, $bhp'(Y, Z) = (\|, 0)$, $bhp'(U, X) = (\rightsquigarrow, 3)$, $bhp'(Y, V) = (\leftsquigarrow, 5)$, etc. To derive useful behavioral properties between tasks of a BP, we remove cyclic edges, because their existence makes all tasks inside each BP concurrent.

Transforming BPMs into executable processes is achieved through a *configuration* step, where process engineers assign operations of web services to *service* tasks in the considered BPM. Configuring a BP can be expressed as a function that takes a task of a BP and assigns an operation to that task if it is a *service* task. The business process shown in Fig. 1.2 can be configured as follows: $conf(U) =$ `BookFlightTicket`, $conf(V) =$ `BookTrainTicket`, $conf(W) =$ `HotelReservation`, $conf(X) =$ `CarRental`, $conf(Y) =$ `FindRestaurants`, and $conf(Z) =$ `FindSights`. Where values to the right of the $conf$ function are operations of web services.

1.4 Annotating Web Services Using Business Process Knowledge

In this section, we describe our approach to generate annotations and derive contexts for web services based on the configurations of their consuming business processes. From each finalized (i.e., configured) BPM, we generate a *task-to-web service assignment* list, based on which we generate annotations for web services from the tasks of their consuming business processes. Task labels and documentation are extracted and their assigned web services are annotated with this extracted information. These labels and documentations are created by service consumers that represent the application level, i.e., business people. Additionally, the title of the created BPM is used to derive the context in which these web services are typically used. This information is then used to enable context-aware service selection for similar cases in the future.

Sharing information about business processes and web services used to execute them is not usually desired by service consumers, because this information might be considered one aspect of their competitive advantage. Nevertheless, our approach can be valuable in several scenarios and application domains, in particular, where high potentials for collaboration are expected and low potentials for competition among service consumers exist, such as government services, education and research, online modeling platforms, and quality-based service brokers.

Figure 1.3 shows a process model using BPMN for establishing a UK limited company. The first six activities of the process are services of the UK Companies

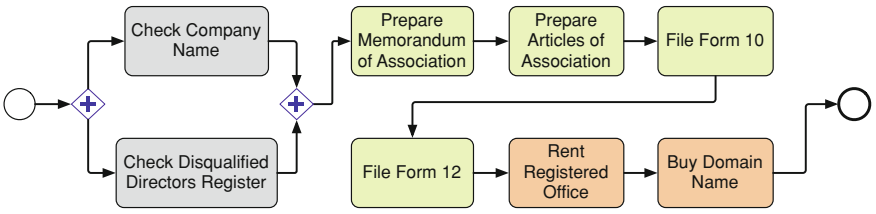

Fig. 1.3 Example process model of establishing a UK limited company

House. In the beginning, it has to be checked whether the desired company name is not already in use and the directors are not disqualified from leading a company. For the remaining steps, electronic forms are provided in the Portable Document Format (PDF). The last two activities in the model are web services offered by private service providers, e.g., "Buy domain name" can be executed using the `whois` web service (http://www.webservicex.net/whois.asmx).

Using our approach, a process engineer can configure their BPM to establish a company in UK using the existing annotations for web services and their operations used in such a model. These annotations are generated from the websites of their providers and through invocation analysis. Although they might be not rich enough, such annotation can be helpful in some cases. Generating additional annotations for such web services and operations from this BPM enrich their descriptions. In this particular use case, all used operations are associated with the context "establish a company in UK". Additionally, each operation is annotated with the label and documentation of each task that uses it. For instance, the `whois` web service is annotated with "buy domain name".

According to the German Federal Ministry of Economics and Technology, establishing a company in Germany incorporates 9 major steps.[2] For instance, check the company name, notarize the articles of association and foundation agreement, notify the Office of Business and Standards, register at the Trade Office (involves check manager's qualifications), etc. Some of these steps are similar to the ones involved in establishing a limited company in UK, such as "check the company name", "check qualified managers", "rent office", "buy domain name". For instance, `whois` web service (http://www.webservicex.net/whois.asmx), that is used to execute the "buy domain name" task in the case of UK company, can be suggested to execute its counterpart task in the German case.

Saving this model adds additional information to the service registry about the considered web services, such as the new labels and the current context. This additional information helps the service registry provide better results in future similar business processes, such as "Establishing a company in France".

1.5 Fine-Grained Linkage Patterns Among Web Services

Relations among web services are important to understand the functionalities of these web services and the interaction among them. We discover preliminary relations among web services from their `WSDL` files, and derive additional fine-grained ones in the form of linkage patterns using their consuming BPs (Fig. 1.1). Each of these linkage patterns has one of these types: *Predecessors*, *successors*, *similar*, *complementary*, and *related*. Moreover, we assign weights to such relations based on the usage of their web services in the corresponding BP. This weight is used to rank web services that have the same linkage pattern, e.g., rank web services that have

[2] http://www.existenzgruender.de/englisch/

the *predecessor* relation with a particular web service. In this section, we describe the types and weights of linkage patterns that we find based on business process knowledge.

1.5.1 Types of Linkage Patterns

Traditional approaches to discover relations among operations of web services usually give binary decisions whether two web services are related or not, without providing control flow dependency, e.g., parallel, sequence, etc. Such relations are not sufficient given the increasing number and complexity of web services and business processes. In our approach—based on extended behavioral profiles—we are able to identify five types of relations among operations of web services based on their usage in BPMs. Consider two tasks, A and B, that are configured with OP_1 and OP_2, respectively. Based on their behavioral properties, the following five types of linkage patterns can be identified:

1. **Predecessor**: An operation OP_1 is a predecessor of another operation OP_2 if it appears *before* OP_2 in the configurations of BPMs where both operations have been used, i.e., $bhp(A, B) = \rightsquigarrow$.
2. **Successor**: An operation OP_1 is a successor of another operation OP_2 if it appears *after* OP_2 in the configurations of BPMs where both operations have been used, i.e., $bhp(A, B) = \leftsquigarrow$.
3. **Similar**: An operation OP_1 is similar to another operation OP_2 if it appears within exclusive relations with OP_2 in the configurations of BPMs where both operations have been used (i.e., $bhp(A, B) = \#$) and there is a high semantic similarity between the terms used to label both tasks and their executing operations, e.g., *"rent a bike"* and *"buy a bike"*.
4. **Complementary**: An operation OP_1 is complementary to another operation OP_2 if it appears within exclusive relations with OP_2 in the configurations of BPMs where both operations have been used (i.e., $bhp(A, B) = \#$) but there is *no high* semantic similarity between the terms used to label both tasks and their executing operations, e.g., *accept* and *reject*.
5. **Related**: An operation OP_1 is related to another operation OP_2 if it appears *concurrently* to OP_2 in the configurations of BPMs where both operations have been used, i.e., $bhp(A, B) = \|$. For instance, *"validate address"* and *"validate email address"*.

1.5.2 Weights of Linkage Patterns

Existing approaches of discovering relations among web services give binary decisions on whether there is a relation between two web services or not. Such decisions are based on the co-occurrence of both services in service compositions,

for instance. In our approach, we are able to discover fine-grained relations and assign a weight (between 0 and 1) to each relation to reflect its strength.

The first type of information that we use to calculate the weight of a relation between two web services is the distance between their consuming tasks in the corresponding BP. This information is provided in the extended behavioral profile of the BP. The distance between any two tasks in a BP is greater than 0 if their behavioral property is either strict order or inverse order. Therefore, the distance is used to assign a weight to predecessor and successor linkage patterns only. The weight, ω, of a linkage pattern, r, between two operations where the distance between their consuming tasks is d, and the maximum distance between any pair of tasks in their BP is len, is given by Eq. 1.1.

$$\omega(r) = \frac{len - d}{len} \tag{1.1}$$

Exclusive tasks can be similar (doing the same functionality) or complementary to each other (doing different functionalities). For instance, "Get weather by city" and "Get weather by post code" are *similar* tasks. Whereas, "Send acceptance" and "Send rejection" are *complementary*. To determine the linkage pattern between two web services whose consuming tasks are exclusive to each other, we investigate the semantics of terms appearing in their names and their consuming tasks. We use WordNet [13] to find `synsets` for these terms and calculate the average distance, (syn_dist), among their *nearest common ancestors (NCA)* in WordNet [26]. The special value (-1) means that there is no similarity between both terms, i.e., they do not have a common ancestor in WordNet, e.g., acceptance and rejection. If the average distance, (syn_dist), is between 0 and a predefined threshold, then the linkage pattern between both services is *similar* and its weight is calculated using the same equation above, where len is replaced by our threshold value, and d is replaced by syn_dist. For instance, *syn_dist* ("*bookFlightTicket*", "*BookTrainTicket*") $= 16$. Based on our experiments, we set the value of the maximum WordNet distance threshold to 20. Given this value, the aforementioned web services are *similar* and the weight of their linkage pattern is 0.2. Linkage patterns are classified as *complementary* if the semantic similarity is low, i.e., syn_dist is higher than the predefined threshold. Linkage patterns *complementary* and *related* are assigned the weight 1.

Whenever a new BPM is created by a service consumer, we discover all possible linkage patterns from that BPM and store them in the database of the service registry. Frequencies and weights of linkage patterns are used to derive scores for these patterns to rank recommended web services within each type of recommendation. The score of each linkage pattern is the aggregation of weights of all instances of this pattern that are typically discovered from multiple BPMs. In practice, web services are used by different service consumers in multiple business processes with different arrangements. These differences result in incompatible relations among web services, i.e., ws_1 is a predecessor for ws_2 in one business process, but ws_1 is similar to ws_2 in another business process. To handle such situations, we merge all behavioral profiles of business process in a single global profile. Additionally, we use gained

knowledge in this global profile to predict relations among web services that are not used together in the same business process, yet (Sect. 1.6.).

1.5.3 Example: Linkage Patterns of Purchase Order Processing

In this section, we apply our approach to a real-world purchase order processing scenario from the SAP Reference Model, whose BP is shown in Fig. 1.4. When this process is configured, we assume that a single operation of a web service is assigned to each task in this model. For instance, operation *A* is assigned to task "process purchase requisition order". Following the traditional approaches of discovering relations among web services, we get the result that there is a relation between *A* and *B*. No further information about the type and strength of this relation is provided. In our approach, we get the extended behavioral profile that encapsulates business process knowledge for these operations as shown previously. This extended behavioral profile is shown in Table 1.1.

From Table 1.1, we notice that operations *A* and *E* are exclusive and also are the operations *A* and *B*. We refine this relation further as either *similar* or *complementary*. To achieve this refinement, we analyze the semantics of the terms in the labels of their corresponding tasks. Based on our experiments, we set our threshold maximum WordNet distance to 20 to control the similarity search in WordNet. We repeat this step for each pair of operations that are exclusive to each other. With result obtained, we establish the linkage patterns among the operations as shown in Table 1.2. Based on the semantic analysis, the *exclusive* relation between operations *A*, *E*—obtained from the profile—is refined to a complementary linkage pattern. On the other hand, the exclusive relation between *A*, *B* is identified as similar, because of the high similarity between terms appearing in the labels of their counterpart tasks.

Using these linkage patterns, users who search for a particular service, e.g., *A*, get useful lists of recommendations. These recommendations represent inter-links among web services that help service consumers explore web service comfortably.

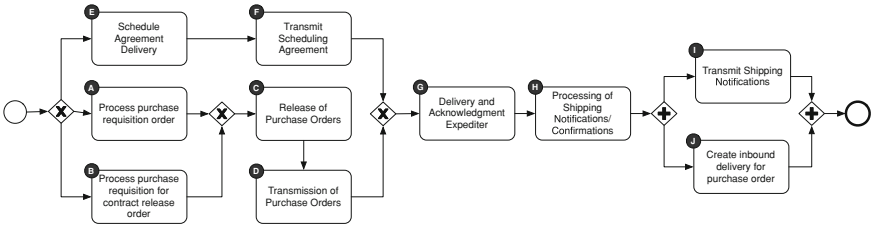

Fig. 1.4 A business process for "purchase order processing" from *SAP* reference model represented in BPMN 1.0

Table 1.1 Extended behavioral profile for business process in Fig. 1.4

	A	B	C	D	E	F	G	H	I	J
A	(‖ , 0)	(#, 0)	(⇝, 2)	(⇝, 3)	(#, 0)	(#, 0)	(⇝, 5)	(⇝, 6)	(⇝, 8)	(⇝, 8)
B	(# , 0)	(‖, 0)	(⇝, 2)	(⇝, 3)	(#, 0)	(#, 0)	(⇝, 5)	(⇝, 6)	(⇝, 8)	(⇝, 8)
C	(⇜, 2)	(⇜, 2)	(‖, 0)	(⇝, 1)	(#, 0)	(#, 0)	(⇝, 3)	(⇝, 4)	(⇝, 6)	(⇝, 6)
D	(⇜, 3)	(⇜, 3)	(⇜, 1)	(‖, 0)	(#, 0)	(#, 0)	(⇝, 2)	(⇝, 3)	(⇝, 5)	(⇝, 5)
E	(# , 0)	(#, 0)	(#, 0)	(#, 0)	(‖, 0)	(⇝, 1)	(⇝, 3)	(⇝, 4)	(⇝, 6)	(⇝, 6)
F	(# , 0)	(#, 0)	(#, 0)	(#, 0)	(⇜, 1)	(‖, 0)	(⇝, 2)	(⇝, 3)	(⇝, 5)	(⇝, 5)
G	(⇜, 5)	(⇜, 5)	(⇜, 3)	(⇜, 2)	(⇜, 3)	(⇜, 2)	(‖, 0)	(⇝, 1)	(⇝, 3)	(⇝, 3)
H	(⇜, 5)	(⇜, 5)	(⇜, 3)	(⇜, 2)	(⇜, 4)	(⇜, 3)	(⇜, 1)	(‖, 0)	(⇝, 2)	(⇝, 2)
I	(⇜, 8)	(⇜, 8)	(⇜, 6)	(⇜, 5)	(⇜, 6)	(⇜, 5)	(⇜, 3)	(⇜, 2)	(‖, 0)	(‖, 0)
J	(⇜, 8)	(⇜, 8)	(⇜, 6)	(⇜, 5)	(⇜, 6)	(⇜, 5)	(⇜, 3)	(⇜, 2)	(‖, 0)	(‖, 0)

Table 1.2 Linkage patterns for business process in Fig. 1.4. *P* Predecessor, *S* Successor, *M* Similar, *C* Complementary, *R* Related

	A	B	C	D	E	F	G	H	I	J
A	–	(*M*, 0.300)	(*S*, 0.875)	(*S*, 0.750)	(*C*, 1.000)	(*C*, 1.000)	(*S*, 0.500)	(*S*, 0.375)	(*S*, 0.125)	(*S*, 0.125)
B	(*M*, 0.300)	–	(*S*, 0.875)	(*S*, 0.750)	(*C*, 1.000)	(*C*, 1.000)	(*S*, 0.500)	(*S*, 0.375)	(*S*, 0.125)	(*S*, 0.125)
C	(*P*, 0.875)	(*P*, 0.875)	–	(*S*, 1.000)	(*C*, 1.000)	(*C*, 1.000)	(*S*, 0.750)	(*S*, 0.625)	(*S*, 0.375)	(*S*, 0.375)
D	(*P*, 0.750)	(*P*, 0.750)	(*P*, 1.000)	–	(*C*, 1.000)	(*C*, 1.000)	(*S*, 0.875)	(*S*, 0.750)	(*S*, 0.500)	(*S*, 0.500)
E	(*C*, 1.000)	(*C*, 1.000)	(*C*, 1.000)	(*C*, 1.000)	–	(*S*, 1.000)	(*S*, 0.750)	(*S*, 0.625)	(*S*, 0.375)	(*S*, 0.375)
F	(*C*, 1.000)	(*C*, 1.000)	(*C*, 1.000)	(*C*, 1.000)	(*P*, 1.000)	–	(*S*, 0.875)	(*S*, 0.750)	(*S*, 0.500)	(*S*, 0.500)
G	(*P*, 0.500)	(*P*, 0.500)	(*P*, 0.750)	(*P*, 0.875)	(*P*, 0.750)	(*P*, 0.875)	–	(*S*, 1.000)	(*S*, 0.750)	(*S*, 0.750)
H	(*P*, 0.500)	(*P*, 0.500)	(*P*, 0.750)	(*P*, 0.875)	(*P*, 0.625)	(*P*, 0.750)	(*P*, 1.000)	–	(*S*, 0.875)	(*S*, 0.875)
I	(*P*, 0.125)	(*P*, 0.125)	(*P*, 0.375)	(*P*, 0.500)	(*P*, 0.375)	(*P*, 0.500)	(*P*, 0.750)	(*P*, 0.875)	–	(*R*, 1.000)
J	(*P*, 0.125)	(*P*, 0.125)	(*P*, 0.375)	(*P*, 0.500)	(*P*, 0.375)	(*P*, 0.500)	(*P*, 0.750)	(*P*, 0.875)	(*R*, 1.000)	–