Foreword

by Cesare Pautasso

Computer scientists have always been in search of suitable metaphors to explain what computing is about. Objects, Components, and more recently, Services have been proposed as the key abstraction to structure the architecture (another metaphor) of large-scale software systems. Delivering software components as a service is more than a technological feat. Services have profound implications on the business model of the software industry. Components were shipped (originally on magnetic or optical disks) and locally installed before they could be used or reused and embedded within software applications. License agreements to grant the right to use a software component were instrumental to the establishment and rapid growth of the software industry. Services radically changed this "industrial manufacturing" paradigm, since, to be used, software-delivered-as-a-service only requires a working network connection and a standard Web browser. Depending on their terms of use, software services can be made available for free (with development and operational costs offset against advertising revenues) or accessed through a pay-per-use subscription model. Service providers can potentially earn revenue with every call into their system (or click through their Web pages), while component manufacturers may only charge licensing fees at every component installation (or upgrade).

Nowadays, not only software is delivered as a service, but also data, business processes, entire operating systems, and (virtualized) hardware platforms are made remotely accessible through the Cloud, another metaphor, representing the result of recent technological advances (virtualization, data centers, broadband networks) focusing on scalability and reliability of logically centralized but physically redundant infrastructures. Again, the Cloud represents more than technology, as it enables a fast-growing service marketplace that leverages economies of scale to apply the pay-per-use model across all layers of IT architectures. Assuming the availability of reliable network connectivity and assuming that the service provider can be trusted, the Cloud offers the opportunity to reduce capital expenditures for building and maintaining in-house IT infrastructures, which can be partially or completely outsourced. This flexible model has led to new terms, such as "elasticity" or "elastic scalability," which use a powerful physical metaphor to indicate one of the defining quality attributes of the Cloud, whereby all kinds of computing, storage, or communication resources can be dynamically acquired when needed and only in the necessary amount. This achievement would not have been possible without the success of the "service-driven" paradigm and the corresponding technological advances.

There have been many attempts to define what a service is, and on a more technical level, determine how a service can be described, designed, developed, and delivered to be discovered and accessed by clients and customers. The growth of an open-service ecosystem can only be enabled with suitable interoperability standards, where all services can interact with minimal friction. Within the boundaries of such standards, the ecosystem will of course be closed with respect to outside interactions, but if the

standards are designed to enable more and more services to join at minimal cost, and the size of the ecosystem grows beyond a critical mass, we can consider the result to be "a brave new open world" that relaxes many traditional software engineering assumptions. For example, as witnessed by some of the work presented in this book, it becomes critical to study and solve how dependable software architectures can be built out of services that may (or may not) be always available during the lifetime of the system depending on them. This is a significantly different problem, since, by definition, components are controlled by their users, while services are operated by their providers, which independently control their interface, their implementation, and their state.

While service-oriented computing is closely aligned with SOA, service-driven computing—the theme of this book—refers to any computing technology that is primarily driven by services in any form and therefore encompasses a larger set of technologies. Over the past decades, we have witnessed several technological waves of middleware, connectors, and enterprise service buses that are used throughout the whole service lifecycle to build service-oriented software architectures. Architects and developers have struggled with many design decisions: Are services simply a form of distributed objects? Should reliable enterprise services rely on distributed transactions and message queues? What is the difference between document-style and RPC-style services? Are Web services misusing the Web as a tunnel? Should Web services be part of the Web? What about real-time, streaming services? How can we capture the semantics of a service so that it can be automatically discovered and integrated? Is it possible to dynamically find the optimal configuration of a service deployed in a Cloud data center outside the control of its original service provider? In this book, you will find many answers to these challenging questions. These will give you an up-to-date outlook on the state of the field from the perspective of the selected contributions collected in the book, which revolve around three of the latest architectural trends driven by the service metaphor: dynamic adaptation, integration, and the Cloud.

Cesare Pautasso
University of Lugano, Switzerland

Cesare Pautasso *is an Associate Professor at the Faculty of Informatics of the University of Lugano, Switzerland. Previously, he was a researcher at the IBM Zurich Research Lab. His research group focuses on building experimental systems to explore the architecture, design, and engineering of next-generation Web information systems. His teaching, training, and consulting activities both in academia and in industry cover advanced topics related to Cloud computing, RESTful Web Service design, business process management, enterprise software architecture, Service-Oriented Architectures and emerging Web services, and middleware technologies. In 2012, he co-authored the book SOA with REST, published by Prentice Hall. He was the ICSOC 2013 program co-chair, the ECOWS 2011 general chair, and started the WS-REST workshop series at WWW. He regularly referees for Swiss, EU, and international funding agencies. Since 2010, he is an advisory board member of EnterpriseWeb LLC.*