Preface

INTRODUCTION

Most enterprises contain a systemic infrastructure of several heterogeneous information technology systems, creating a complex, fuzzy network of interconnected applications, services and data sources, which is usually not well documented and expensive to maintain. Moreover, the introduction of multi-oriented, separate systems concerning enterprise resource planning (ERP), customer relationship management (CRM), supply chain management (SCM), e-business portals and B2B transactions, increases the complexity of systems integration, making the support of the interoperability among these systems a challenging task.

In this emerging business context, a clear need appears to link these former incompatible systems to improve productivity and efficiency. The solution to this need is what is called enterprise application integration (EAI), which can be defined as the use of software and architectural principles to bring together (integrate) a set of enterprise computer applications.

The goal of EAI is to integrate and streamline heterogeneous business processes across different applications and business units. We can distinguish between intra- and inter-organizational enterprise application integration. Intraorganizational EAI, commonly referred as “application to application”-integration (A2A), specifies the automated and event-driven exchange of information between heterogeneous enterprise applications and systems operating within an organization or enterprise. On the other hand, interorganizational EAI, or else B2B integration, specifies the automated and event-driven information exchange between various systems of several collaborating organizations and enterprises.

In recent years, most enterprises and organizations have made extensive investments in several EAI systems and solutions that promise to solve the major integration problem among their existing systems and resources. The business driver behind all these traditional EAI projects is to integrate processes across third-party applications as well as legacy systems to decrease the number of adapters one has to develop if connecting two systems. Traditional EAI focuses on the message-based communication of software applications interfaces, by pipelining different middleware technologies and developing various adapters, connectors, and plug-ins to provide efficient messaging support among heterogeneous systems, allowing their effective interconnection.

However, traditional EAI efforts lack an upper abstraction layer, as well as standardized architectures and implementations, making customers and end-users captive of EAI vendor-specific solutions, and creating the need for a new, high level integration for interconnecting various EAI systems with one another.

Such a need for higher level integration can be supported by exploiting recent research in service-oriented architectures and Semantic Web technologies.
Service-oriented architecture (SOA) is an approach to defining integration architectures based on the concept of a service. The goal of SOA can be described as bringing the benefits of loose coupling and encapsulation to integration at an enterprise level. In order to describe SOA, it is first necessary to define what we understand by a “service” in this context. The most commonly agreed-on aspects of the definition of a service in SOA are: services are defined by explicit, implementation-independent interfaces; services are loosely bound and invoked through communication protocols that stress location transparency and interoperability; and services encapsulate reusable business functions. After a function has been encapsulated and defined as a service in an SOA, it can be used and reused by one or more systems that participate in the architecture. Note that in contrast to reusing service implementations at runtime, the encapsulation of functions as services and their definition using interfaces also enables the substitution of one service implementation for another. For example, the same service might be provided by multiple providers, and individual service requesters might be routed to individual service providers through some intermediary agent. The encapsulation of services by interfaces and their invocation through location-transparent, interoperable protocols are the basic means by which SOA enables increased flexibility and reusability.

The purpose of the Semantic Web effort is to make the web machine processable rather than merely “human processable.” The upcoming standards of the Semantic Web provide a set of concepts that can be used to annotate services and processes in a way machines can analyze. The key components of the Semantic Web technology which enable the design and development of ontologies for structuring and annotation of services and processes, include: a general-purpose language for representing information in the Web called resource definition framework (RDF); an RDF abstraction layer called RDF schema (RDFS); a language with well defined, formal semantics, built on RDF, such as the Web ontology language (OWL); and formal ontologies for marking up Web resources, used by semantically enriched service level descriptions. Enriching Web services descriptions with formally defined semantics by introducing the notion of semantic mark-up, leading towards Semantic Web services, enables machine-interpretable profiles of services and applications, realizing the vision of dynamic and seamless integration. As this semantic mark-up is machine-processable and–interpretable, the semantic profiles of Web services can be exploited to automate the tasks of discovering Web services, executing them, composing them, and interoperating with them.

The combination of SOA architectures and Semantic Web technologies supports and allows a set of essential automated services: automatic Web service discovery, involving automatic location of Web services that provide a particular functionality and that adhere to requested properties expressed as a user goal; automatic Web service composition, involving dynamic combination and aggregation of several Web services to provide a given functionality, automatic Web service invocation, involving automatic execution of an identified Web service by an agent, and automatic Web service interoperation within and across organizational boundaries.

These semantically enriched Web service oriented features can constitute the ideal solution to integration problems, as they enable dynamic, scalable and reusable cooperation between different systems and organizations.

This book presents innovative research results on the use of Semantic Web technologies and service-oriented architectures for enterprise application integration. It presents novel methods and tools, which apply semantic technologies in real-world complex systemic environments. It provides the theoretical foundations, principles, methodologies, architectures, technical frameworks, and case studies for the design and development of semantic enterprise application integration.

The audience of this book are scholars and researchers in the fields of EAI, Semantic Web technologies, interoperability, and semantic integration. The book aims to provide “food for thought” for future
research. However, it also aims to address the needs of practitioners working on complex supply-chain environments, who may find within the book the foundations for new ideas in the EAI field. Graduate and post-graduate students would also find this book to be a useful reference resource.

CHAPTER OVERVIEW

Chapter 1 presents a methodology, which has resulted in the implementation of a highly customizable collaborative environment focused to support ontology-based enterprise interoperability. A key issue addressed by the particular platform is the variety and number of different resources that concur to achieve a cross-enterprise business service. A second key issue is the diversity of agreed (e.g., meaning negotiation when creating online contracts) models, and the difficulty in adapting its integrated features and services to different situations. These problems are addressed with a flexible solution, avoiding rigidity that occurs in the implementation and maintenance of existing cooperation platforms and their integration with an advanced semantic repository. The proposed platform operates at two levels: at the front end, it enables the end users to access seamless collaborative (e.g., synchronous, asynchronous and semi-synchronous) as well as individual mode tools and services to extract valuable information; at the back end, it uses a sophisticated ontology framework to support and record the collaborative work, enhancing interoperability among different enterprises and other service providers.

Chapter 2 proposes a semantically-enriched approach for dynamic data mediation in EAI scenarios, focusing on the resolution of message level heterogeneities between collaborative enterprise services and facilitating automated data mediation during execution time by providing formal transformations of the output and input messages (of the participating services) to a common reference business data model, that is, an enterprise interoperability ontology. The chapter presents a tool that has been developed to support the user to provide business data-related semantic annotations and XSLT transformations of the input and output message parts of collaborative enterprise services. Finally, it demonstrates the utilization of the proposed approach and tool in a real-world EAI scenario.

Chapter 3 discusses architecture abstractions like patterns and styles which can be used to capture design knowledge and allow the reuse of successfully applied designs, thus improving the quality of software. Knowledge gained from integration projects can be captured to build a repository of semantically enriched, experience-based solutions. Business patterns identify the interaction and structure between users, business processes, and data. Specific integration and composition patterns at a more technical level address enterprise application integration and capture reliable architecture solutions. The chapter uses an ontology-based approach to capture architecture and process patterns. Ontology techniques for pattern definition, extension, and composition are developed and their applicability in business process-driven application integration is demonstrated.

Chapter 4 introduces a three-tier framework for managing and implementing interoperable and cross-organizational business processes. The chapter aims to fill the gap currently existing between processes defined on a strategic level and executed models. It describes a solution which supports rapid prototyping by combining a model-driven framework for cross-organisational business processes with an agent-based approach for flexible process execution. It then shows how the W3C recommendation for Semantic Web service descriptions can be combined with the model-driven approach for rapid service integration.

Chapter 5 reviews several Semantic Web services frameworks that intend to bring semantics to Web services. The chapter depicts some ideas from SOA and Semantic Web services and their application to enterprise application integration. It illustrates an example of logic-based semantic matching between consumer services and provided services, which are described in ontologies.
Chapter 6 introduces an approach for developing service registries building on the UDDI standard and offering semantically-enhanced publication and discovery capabilities in order to overcome some of the known limitations of conventional service registries. The approach aspires to promote efficiency in EAI in a number of ways, but primarily by automating the task of evaluating service integrability on the basis of the input and output messages that are defined in the Web service’s interface. The presented solution combines the use of three technology standards to meet its objectives: OWL-DL, for modelling service characteristics and performing fine-grained service matchmaking via DL reasoning, SAWSDL, for creating semantically annotated descriptions of service interfaces, and UDDI, for storing and retrieving syntactic and semantic information about services and service providers.

Chapter 7 considers semantics in two major integration tasks. First, semantically corresponding data types that can be used for communication between components need to be identified. Second, natural language documentation needs to be studied in order to understand component behaviour, that is, dependencies between operation invocations and how semantically different outcomes of operation calls are represented in the technical output format. The approach presented in this chapter supports the two tasks as follows. First, closed frequent itemset mining (CFIM) is employed to help identifying semantically corresponding data types. Second, a formal representation for component behaviour is introduced. However, as component behaviour is specified during component development, but used during integration—two distinct phases involving distinct teams—the chapter provides model transformations to ensure the consistent transfer of generic behavioural information to specific integration constraints before automated integration techniques can be applied.

Chapter 8 describes a solution for service discovery and selection at the business level, that is, at the level of offered business capability of a service provider and the ability to serve a concrete requested business transaction. The proposed solution is based on semantic interpretation of offered service capabilities, contractual restrictions, business rules of the requestor specifying selection preferences, and the parameters of the run-time service request. The applicability of the proposed solution is demonstrated on a shipper-carrier integration scenario.

Chapter 9 is an application in the tourism industry. Currently in the travel domain most of the travel products are sold through global distribution systems (GDSs). Since only major airline companies or hotel chains can afford to join GDSs, it is difficult for small and medium enterprises to market their travel products. This chapter describe a middleware, called SATINE, to address this problem. In the SATINE middleware, existing travel applications are wrapped as Web services. Web services, as such, is of limited use because the service consumer must know all the details of the Web service like the functionality of the Web service (what it does) and the content and the structure of input and output messages. Therefore, the chapter proposes the annotation of both the service functionality and the service messages with Web ontology language (OWL) ontologies. Through the SATINE middleware, the travel parties can expose their existing applications as semantic Web services either to their Web site or to Web service registries they maintain. SATINE middleware facilitates the discovery and execution of these services seamlessly to the user.

Chapter 10 presents the FUSION approach for supporting collaboration and interconnection between enterprises with technologies that allow for the semantic fusion of heterogeneous service-oriented business applications. The chapter provides an overview on the FUSION approach and summarises integration experiences with the application of the approach and tools during the implementation in the case of career and human resource management services.

Chapter 11 extends a professional business process management (BPM) tool to allow semantic business process modelling using the event-driven process chains (EPC) notation. The chapter presents an adaptation of the tool’s EPC to business process execution language (BPEL) transformation to preserve
the semantic annotations. By introducing a proxy service, it is possible to perform Semantic Web service discovery on a standard BPEL engine. The chapter presents the evaluation of the approach in an empirical case study, which was replicated 13 times by 17 participants from 8 different organisations. Valuable feedback was received, which is interesting for researchers and practitioners trying to bring semantic technologies to end-users with limited background knowledge about semantics.