A Top-Down Methodology for Building Semantic-Rich Service-Oriented Collaborative Virtual Enterprise (CVE)

Gang Chen¹, Wei Ren¹, David Chen², Jing Bing Zhang³, Chengzheng Sun⁴, Zhonghua Yang¹, Chor Ping Low¹ and Liqun Zhuang⁰

¹School of Electrical and Electronics Engineering, ⁴School of Computer Engineering, Nanyang Technological University, Singapore 639798
²School of Computing & Information Technology, Griffith University, Australia 4111
³Singapore Institute of Manufacturing Technology (SIMTech), Singapore 638075

Contact email: WeiRen@ntu.edu.sg

Abstract—Collaborative Virtual Enterprise (CVE) is a new paradigm for industry and it imposes significant challenges on effective operation. In this paper, we formalized a top-down methodology for designing and developing service-oriented CVE with rich semantic support. This methodology facilitates automatic execution of virtual business process via service composition. The semantic notion is emphasized as semantic web services technologies (i.e. OWL-S) have become an integrated and essential part of our top-down methodology. The methodology has been applied to manufacturing sector for a virtual enterprise which has the business proposition of effective and efficient business process. To our knowledge, this is among the first attempts to use Semantic Web and Web services technologies in such a scale and sophistication.

I. INTRODUCTION

The emergence of Web services and Semantic Web services opens the new opportunities for enterprise integration and provides great potentials for automated integration. A Collaborative Virtual Enterprise (CVE) is a temporary alliance of enterprises that share skills or core competencies and resources in order to fulfill common business goals in a more collaborative rather than competitive manner.

Web service-oriented approach provides a standard-based higher level of abstraction than conventional message oriented approach to facilitate CVE integration. It standardizes various aspects of message exchange, including message format, message exchange pattern, and message transport protocol. Specifically, the whole enterprise system is virtualized through a network of services (functionalties). Each service exposes its functionality in terms of a group of interfaces, publicly described and discovered based on Web service standard.

Within a service oriented environment, the enterprise integration problem can be further viewed as a problem of service integration or service composition. In the literature, many service composers have been developed in an attempt to achieve flexible and automatic services composition. Their research results suggest that in order to realize effective integration at the enterprise level, each service in the system should be annotated with rich-semantic information. This imposes a new challenge of building a semantic-rich service-oriented environment.

Under the general assumption that the enterprise integration requirement is given in the form of business process model and data model, we have formalized a top-down methodology for designing and developing service-oriented CVE with rich semantic support. This methodology facilitates automatic execution of virtual business process via service composition and comprises six major steps. It is convenient to view these six steps as forming a chain of operations that systematically transfer a business model to the final design and implementation of a web services oriented CVE system. The semantic notion is emphasized as semantic web services technologies (i.e. OWL-S) have become an integrated and essential part of our top-down methodology. Particular attention has been given to the use of OWL-S for semantic markup of manufacturing web services and for ontology design.

This paper is organized as follows. Section II provides an overview of web services, particularly semantic web services technology from the perspective of enterprise integration. Section III introduces the service-oriented integration architecture and core service composition notions. A top down approach to realize the CVE is presented in Section IV. Its application in an enterprise manufacturing system is discussed in Section V. Finally Section VI concludes the paper.

II. WEB SERVICE TECHNOLOGIES FOR ENTERPRISE INTEGRATION

As an industry analysis shows, 95% of difficulties in integrations require to solve semantic issues which surfaces in many aspects such as data formats, transformation rules,
service level agreements, etc. Hopefully, these issues can be circumvented with rich semantics using semantic web technologies [1]. Two core technologies are crucial for creating semantic rich environments: ontologies and standard languages for developing ontologies. A web standard-based way for representing ontology is the use of Web Ontology Language (OWL) [2]. Built on top of XML, XML Schema, RDF, and RDFS schema, OWL adds more vocabulary for describing properties and classes, relations between classes, cardinality, equality, richer typing of properties, characteristics of properties, and enumerated classes. OWL benefits from many years of research in description logic. Practically, there are several known highly-optimized reasoning algorithms and implemented systems for OWL [3] [4]. OWL has emerged as a web standard language for ontology.

The landscape of technologies for bringing semantics to web services is a bit blurred. There are currently three W3C submissions for standardization: OWL-S, WEMO, and WSDL-S [5] [6] [7]. OWL-S is a collection of loose upper ontologies and is description oriented. It focuses on capabilities, functional properties, and other non-functional properties (such as QoS). The behavior of a web service is modeled with a process/action paradigm rooted in situation calculus. Comparing with other competing standards, OWL-S is well-developed with a broad user base. In this paper, we adopt OWL-S, an OWL-based web service upper ontology, and domain ontologies for the purpose of associating rich semantic information with each web service.

In this paper, we rely on a semantic driven services composer to achieve flexible and automatic enterprise integration. The main concept behind semantic driven services composer is not new in computer science. A composer of this kind typically operates on top of a semantic web service discovery component, which describes how to evaluate a degree of similarity between a service requirement and an actual service instance by measuring the syntactical, operational, and semantic similarity [8]. Many successful discovery (services matching) systems have been reported recently. Most of the matchmaking algorithms based on OWL-S ontologies use the service profile [9]. Flexibility is provided as the matching based on the service profile allows match to be performed through conceptual reasoning over the relationships between service inputs and outputs.

The semantic web service technologies such as OWL-S have attracted numerous research interests in recent years. In industrial applications, however, the Web Services Business Process Execution Language (WS-BPEL) has become a widely-accepted standard. WS-BPEL provides a language and an execution framework for the formal specification of business processes and business interaction protocols [10]. It facilitates the expansion of automated process integration in both the intra-corporate and the business-to-business spaces. However, WSBPEL lacks sufficient semantics required for business process automation. As a result, service composition using WSBPEL is commonly approached from pre-defined workflows. Although OWL-S also support composite web services through several control language constructs competitive in functionality to those provided by WSBPEL, OWL-S is to realize automatic services composition with its rich semantics. This is the main focus of this paper.

III. SERVICE-ORIENTED INTEGRATION ARCHITECTURE

Our top-down methodology for building service-oriented CVE pre-assumes a general and scalable integration architecture [11] as shown in Fig. 1, which will be briefly introduced in this Section. Fundamental to the service-oriented integration is the virtualization layer. Via virtualization, all systems, applications, services and equipments to be integrated in the collaborative virtual enterprise are exposed as web services, and they can offer their services and functionality in a standard web service environment. The basic web service infrastructure (SOAP-WSDL-UDDI) is augmented with business process execution service to support dynamic business process execution. It forms the services infrastructure layer. All the services newly developed or those obtained from the virtualization will be enhanced with domain specific semantic information. As stated above, the emerging semantic web standards provide the web services infrastructure with the semantic interoperability that integration needs. The semantic based services description is unambiguously computer interpretable and facilitates maximal dynamism in web service discovery, selection, composition, and invocation. In a way, semantic web transforms the web service infrastructure into repository of computer manipulatable data with rich semantics. Architecturally, this is achieved in the semantic-enhanced web services layer.

<table>
<thead>
<tr>
<th>Business process execution, service orchestration and choreography</th>
</tr>
</thead>
<tbody>
<tr>
<td>Business process models</td>
</tr>
<tr>
<td>Semantic Enhanced Web Services</td>
</tr>
<tr>
<td>Manufacturing Web Services</td>
</tr>
<tr>
<td>Planning</td>
</tr>
<tr>
<td>Web Service Infrastructure</td>
</tr>
<tr>
<td>Virtualization for Service-Orientiation</td>
</tr>
<tr>
<td>(equipments/machine/devices, applications, data sources)</td>
</tr>
</tbody>
</table>

Fig. 1. The service-oriented architecture for CVE.

The essence of business process modeling is to create a coherent, integrated (and interoperated) business process for achieving enterprise’s ultimate strategic business goals. In a sense, the coherent business processes provide a process perspective of the collaborative virtualized enterprise. Those parts of business processes that are IT-enabled will be realized using web services processes via service orchestration and choreography.
Based on the CVE architecture as shown in Figure 1, in this paper, the top-down system design starts from a business process model. Constituent components (or procedures) within this process model are then identified and virtualized to become self-contained web services. Finally ontologies will be designed and applied to annotate each web service. Upon receiving users’ requests, a services composer that utilizes such semantic information will dynamically forge a business process, which is consistent with the business process model that serves as the design input. Our top down methodology therefore satisfies the design requirement together with the flexibility and dynamicity desirable for enterprise integration. The details will be highlighted in the next section.

IV. A TOP-DOWN METHODOLOGY FOR BUILDING SEMANTIC-RICH SERVICE-ORIENTED CVE

In many situations, usually the requirements of enterprise integration are initially modeled as business process models. The essence of business process modeling is to create a virtualized enterprise (VE) which represents a coherent, integrated and interoperated business process for achieving enterprise’s ultimate strategic business goals. The integrated business process is enabled and realized through IT technology and infrastructure. Using the service-oriented paradigm, it is necessary to transform the business process models into service-oriented process model, resulting in a service-oriented requirement (or goals).

However, mapping from business process to web services, especially semantic web services is a very complex task. It involves a lot of design decisions in business modeling as well as domain knowledge. To effectively handle this complexity, one commonly used strategy is to device a top-down approach to design and realize a semantic-based web service system for enterprise collaboration. We begin with the top-level business model (i.e. process and data model), which is broken into a group of web services by identifying the service interfaces. Afterwards each service interface is refined by adding XML Schema and ontologies, which are derived from a given business data model. Web services at the lowest level are implemented and deployed after getting complete WSDL descriptions. Finally, dynamic enterprise integration is realized via semantic-driven services composition. In our top-down methodology, six major steps as summarized in Table I have been identified to realize the above process. In the remaining of this section, each step will be discussed in some details.

<table>
<thead>
<tr>
<th>Step</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Service Interface Identification</td>
</tr>
</tbody>
</table>

A. Service Interface Identification

We adopt IDEFO for identifying service interface (especially inputs and outputs) of activities. IDEFO suits business, data processing and manufacturing systems because these systems generally consist of a network of inter-connected activities linked by the flow of control, flow of materials and flow of information. Use of IDEFO permits the construction of models comprising system functions (activities, actions, processes, and operations), functional relationships, and data (information or objects) that support systems integration [14]. Over the years, IDEF have been extensively applied in industrial engineering projects [15]. In this paper, we will mainly focus on the flow of information. The execution of a process is triggered by some events such as arrival of a Customer Request, and ended up with some observable result such as a Purchase Order. Most processes have one or more suppliers providing inputs and produce outputs consumed by the following process. Therefore, each constituent process can be characterized by its (1) input flow, (2) the business function that processes the input flow, and (3) output flow as the processing result.

The key of a service-oriented CVE implementation is to identify a set of business services whose execution satisfies a specified business goal. Services provide the highest level of abstraction with well-defined interfaces (input, output and operation which can transform input into output). It is intuitive and convenient to view each business process as a separate web service. Moreover with IDEFO, it is straightforward to establish the mapping from business process to web service. Particularly the input flow of a business process can be mapped to the input message of a corresponding web service. The
output flow can be mapped to the output message and the business function can be mapped to the service operation.

B. UML Data Modeling

While the business data model can be described using IDEF1x to produce information model which represents the structure and semantics of information within an environment or system, in order to have a better bridge between business model and Web services technology, we choose UML to model data structures exchanged between web services. The entity, the attribute and the relationship can be conveniently modeled using UML class diagrams [12].

C. XML Schema Design and WSDL Finalization

When a business process is virtualized as a self-contained web service, the input message as well as output message of this business process should obey rigorously the SOAP protocol. Specifically, the data to be exchanged with the business process, which have been identified and modeled in Subsections A and B, have to be wrapped within a SOAP envelope, which is essentially an XML document. In order to define the structures of those data types within an XML document, this Subsection focuses on the XML schema design. The design starts from the UML class diagram, which is chosen by us for data modeling. Detailed designing steps are listed as follows:

1). For each class in the class diagram, a separate XML element is defined within an XML schema file using the XML schema language (XSD). In this manner, elements actually serve as tags to differentiate various data types.

2). If a class C1 is associated with another class C2, a reference to the element representing C2 will be added to the element definition associated with C1. Depending on the multiplicity of the association, a reference to certain element can be modified with attributes such as “minOccurs” and “maxOccurs”.

3). For each attribute A1 defined within a class C1, an XML attribute with the same name as A1 will be incorporated into definition of the XML element of C1.

4). If class C1 generalizes class C2, a reference to the XML element representing C2 will be added to the XML element of C1. With this treatment, we reduce the burden of re-defining every attribute that appears in C2.

After defining the XML schema for all modeled data types, the schema is further added to every WSDL service description created in Subsection A. With every input message and output message of a web service being annotated with proper schema information, the design of the web service is complete. The corresponding WSDL service description is therefore fully operational and will drive the consequent service implementation to be introduced in the next subsection.

D. Web Service Implementation and Deployment

Model driven tools such as WSDL2Java offers convenient support for implementing and deploying web services from WSDL service descriptions. In a nut-shell, WSDL2Java produces the Java code necessary for building stubs, skeletons, and data types. As these data types are derived from the XML schema defined in Subsection C, they are further compliant with the UML data model obtained in Subsection B. The only thing service developers need to do is to extend skeleton Java classes by adding specific code in order to implement the business logic. The underlying technical details of web services platform and engine are transparent during the implementation process. Web services platform such as Apache AXIS2 provides a convenient mechanism for developed Web services.

The code-first approach can also be used for the cases where the Java codes, especially the service interfaces are already available or want to be first developed. In this case, Java2WSDL tool is applied to generate the corresponding WSDL descriptions. This approach enables a convenient integration of legacy systems. It should be noticed that both approaches rely on the identification of service interfaces. We adopt a contract-first approach and design services interface using WSDL as it provides a standard and platform-independent way of building web services.

E. Domain Ontology and Service Ontology Design

Due to the similarities between the two formalisms, UML class and OWL ontology, there exists research efforts in an attempt to transform UML classes to OWL ontology [13]. However, it is difficult to conduct full transformation from UML to OWL ontology. In this paper, we decide to take the simplest transformation strategy. For each class C1 in UML, a corresponding OWL concept O1 will be created. If class C1 generalizes class C2 in a UML class diagram, O2 will be described as a subclass of O1 in OWL ontology. Moreover for each attribute in a UML class, a datatype property of the corresponding OWL concept will be created. Also for each association relation defined in UML, a corresponding object property will be created. As ontologies typically do not specify any behavior information, the operations as listed within any UML class will be omitted.

Once OWL ontology is available, WSDL description can be further ex-extended to get OWL-S semantic web service descriptions (i.e. service ontology). With help of WSDL2OWLS tools which is developed by University of Maryland, each WSDL description can be transformed directly to OWL-S description. Afterwards, the Inputs, Outputs, Preconditions and Effects of this OWL-S description (known as IOPE) are annotated with semantics (OWL ontology). In order to access web service using OWL-S API, an XSLT script is also required to transform OWL-S parameters (inputs and outputs) to and from XML Schemas (Ref. step 3).

F. Dynamic Service Composition

In order to achieve dynamic integration in CVE, it is necessary to realize the business goals in the form of business process through service-oriented process models. This service-oriented process is a high-level description of what to be done in terms of a sequence of services, that is, a service process. The service process will be realized via service composition. In
In this paper, the service process is conveniently described in OWL-S with IOPE. The service composer implementation involves service discovery (matchmaking), the service selection is governed by the composeability characterization, and finally composition of web services that realize the service process.

The input to a services composer (SC) comprises (1) the requested service capability from the service consumer and (2) all advertised services imported from the OWL-S repository. In our current implementation of SC, the requested service capability refers primarily to the required outputs of a service given the available inputs. The OWL-S repository stores the services advertised by Service Providers. To import the services from OWL-S repository into SC, the OWL-S API library is used which is also developed by University of Maryland. The output from SC is a composition sequence which consists of one or more component services. The component services will be executed based on the control flow so as to achieve the requested capability. Evidently after employing a SC component in a web service-oriented CVE, our top-down methodology realizes the business process model together with the flexibility and dynamicity desirable for enterprise integration.

V. BUILDING A SEMANTIC-ENHANCED WEB SERVICE-ORIENTED ENTERPRISE MANUFACTURING SYSTEM

In this Section, the top-down methodology discussed in the previous Section is explored in order to build an enterprise manufacturing system. Due to the space limitation, we will not go through every detail of the six major design/implementation steps (Ref. Table I). Instead, three important activities as list below are highlighted from the perspective of enterprise integration:

1). Service virtualization: CVE is virtualized as a network of web services with WSDL service interfaces.

2). Enrich web service with semantics: This step is to extend WSDL service descriptions via creating domain ontology and service ontology.

3). Service composition and execution: The dynamic composition of services based on semantic service descriptions, and executing these services to achieve the requested capability.

A. Service Virtualization

The ultimate goal of our enterprise manufacturing system is to achieve a business process model, the simplified version of which is shown in Fig. 2. As evidenced in Fig. 2, effective collaboration of various business functionalities (e.g. scheduling, engineering, material management, manufacturing, inventory, and shipping) is essential to the implementation of this business process. Each of the functionality is virtualized through one or multiple web services. For example, the engineering task is embodied by the Product Engineering web service (E1). Other system components can directly assess the engineering functionality through invoking E1. Driven by our goal, multiple services are linked together in Fig. 2, to form a service process. For instance, the Issue Purchase Order service (C1) will be invoked first, which is followed by the process product purchase order service (C2), as the two services are connected with a directed edge. It is highly likely that several organizations have contributed to this service process. As an example, E1 can be provided by a product design firm, which is different from the company that actually manufactures the product. In this manner, the execution of the service process will lead to an effective collaboration among numerous organizations, and a CVE is therefore formed.

B. Enrich Web Service with Semantics

The inputs and outputs of each service are highlighted in Fig. 2. For example, the input of the Process Product Purchase Order service (C2) is a Product Purchase Order. After executing C2, a Manufacture Order (the output) is returned and will be later feed into another service (i.e. Schedule Production service P1). To facilitate semantic based services discovery and composition, OWL ontology have been designed in order to capture the relationships between various types of inputs and outputs. For illustration purpose, the OWL ontology to model product purchase order and manufacture order has been depicted in Fig. 3 using UML formalities.

Fig. 2. The simplified business process model and the corresponding web services.
C. Service Composition and Execution

Based on the service virtualization in Fig. 2 and ontology design in Fig. 3, the semantic-driven services composer is able to operate with any given set of inputs. We have implemented a services composer component within our enterprise manufacturing system using the forward-chaining algorithm. Given any inputs, the forward-chaining process continues until all the required outputs have been identified. Suppose that the available input to the composer is the manufacturing order, and the required output is the daily production schedule. A composite service execution sequence as shown in Fig. 4 is finally obtained. Consider further that a web service P’, which is able to produce daily schedule upon given a manufacture order, is newly created. When P’ is added to the OWL-S service repository, the composer is able to produce a much simpler composite service sequence, containing only a single service P’. It shows that the composer can cope with the dynamism required by enterprise integration.

Fig. 3. Partial view of the OWL ontology design.

![Diagram]

Fig. 4. The composite service sequence obtained through the services composer.

VI. CONCLUSION

Collaborative Virtual Enterprise (CVE) is a new paradigm for industry and it imposes significant challenges on effective operation. In this paper, we formalized a top-down methodology for designing and developing service-oriented CVE with rich semantic support. This methodology facilitates automatic execution of virtual business process via service composition and is comprised of six major steps. It is convenient to view these six steps as forming a chain of operations that systematically transfer a business model to the final design and implementation of a web services oriented CVE system. The semantic notion is emphasized as semantic web services technologies (i.e. OWL-S) have become an integrated and essential part of our top-down methodology.

The methodology has been applied to manufacturing sector for a virtual enterprise which has the business proposition of effective and efficient business process in order to market and deliver new manufacturing product to customer quickly. To our knowledge, this is among the first attempts to use Semantic Web and Web services technologies in such a scale and sophistication.

ACKNOWLEDGMENT

This work was supported in part by the Agency for Science, Technology, and Research (A*STAR) of Singapore under SERC TSRP on IMSS Grants P0520094 and P0520104.

REFERENCES

[14] NIST, "Integration Definition for Function Modeling (IDF0)," Federal Information Processing Standards FIPS 183, USA.