Service Model for Collaborating Distributed Design and Manufacturing

Moon Jung Chung[1], Woongsup Kim[1], Ravi Gopalan[1], Hong Suk Jung[1], Hyun Kim[2]
[1]*Computer Science and Engineering, Michigan State University, East Lansing MI 48824,*
{chung, kimwoong,gopalanr jungho}@cse.msu.edu
[2]*ETRI,Taejon, Korea , hyunkim@etri.re.kr*

Abstract

This paper presents a Service-Oriented Process Model (SOM) to build a web-services based process management system, called MIDAS that would support distributed Design and Manufacturing process. SOM uses OWL specification, and describes the semantics of the design process with an underlying mathematical mode, Process Grammar, which supports dynamic process creation and enactment. The rich semantics provided by SOM offers a unique functionality of web service-based process composition and iterative process enactment that may be used for any design and manufacturing process, and hence makes it appropriate for usage in scenarios involving distributed and collaborative design and manufacturing process. The use of OWL and Process Grammar makes SOM very suitable for coordination of processes in loosely coupled collaborative environment. We also introduce a framework, MIDAS, which is built upon SOM. Through semantics of SOM, the framework provides facilities for business/technological entities to discover services that other collaborative entities provide, monitor dynamic process environments, and reconfigure process logic by iteration.

1. Introduction

The advent of dynamic markets, changes in production economics coupled with rapid advancements in information technology have set a new stage for manufacturing practices in the fiercely competitive global industry. To stay competitive, manufacturers must be able to 1) manage increasing product complexity and product innovation driven by market demands, 2) have faster and flexible product development cycle, and 3) control globally distributed outsourcing operations. Collaborating Distributed Design and manufacture enables manufacturing organizations in maintaining competitiveness by creating products with lesser turnaround time, lesser cost, and with fewer defects. The distributed design solution unifies the product life cycle by enabling the sharing of product knowledge and incumbent manufacturing applications [1].

A variety of frameworks and specifications have been proposed to manage manufacturing processes in heterogeneous business environment [4]. However, these frameworks mostly focus on the system integration in a closely coupled design and manufacturing environment. These systems therefore have inherent weaknesses in terms of scalability and extensibility in loosely coupled application integration. Moreover, they do not address run-time process reconfiguration sufficiently, which is necessary to reflect dynamic manufacturing environments. Most systems react to dynamic changes based only on pre-defined routines composed at the process definition stage.

In this paper, we present a semantic model, which a Web service framework may use for Distributed Design and Manufacture. The Semantic model called Service Oriented process Model (SOM) is the underlying foundation for the framework, the Manufacturing Integration and Design Automation System (MIDAS) that supports distributed design processes to integrate design engineering, process engineering, business plan and assembly operations in a loosely coupled environment. The system uses the semantic service model to enable a machine agent in actively locating a Web service and choreographing collaborative services into an optimized process workflow. Through such operations, a user in distributed design process will get more sophisticated intellectual aid on process design and management. The framework provides collaborative dynamic process management using web service composition. The MIDAS framework provides a truly distributed architecture for management of manufacturing process composition and inter-operation in two aspects. 1) Process integration is described in terms of choreography of web services, which is achieved by using the Process Grammar, which is a process model helps MIDAS to configure processes dynamically. 2) Also we developed a service-oriented process model, which specifies semantics on manufacturing process. It provides standardized facilities for companies to integrate processes on distributed design environment using web services.

This paper is organized as follows. Section 2 provides a background on distributed design and web-service based process management. Section 3 describes the Process Grammar used for dynamic process configuration. Section 4 introduces the semantic model, which consists of process definition model, process enactment model, and process monitor model. Section 5 shows a general MIDAS architecture and illustrates how distributed design works at the MIDAS framework.

2. Background

Significant research to automate and coordinate design and manufacturing from the perspective of process management [5, 12, 16]. The distributed design process should be easily reconfigured when changes in user requirements occur or when the results may not conform to the constraints and companies should be able to execute their own processes concurrently with others during collaboration [22]. Managing processes in distributed design is highly dynamic and poses challenges completely different from conventional workflow management where workflows are static. In [6] they call such type of process as "enacted processes". These involve sub-processes which are designed "on the fly", by the participants, as part of the main process that is being executed. These characteristics pose challenging problems.

To support inter-operability of business processes, a variety of standards and languages have been proposed. WSFL [19] is a workflow language that provides recursive composition of web services. BPEL4WS allows a composer to aggregate two or more web services into processes which may be abstract for a high-level business transaction or executable as a compiled process [17, 18]. BPML [15, 17] specifies web services orchestration and choreography. Orchestration in this context refers to an executable business process that can interact with both internal and external Web services, while choreography describes relationship and process flow among multi parties or multi organizations. WSCI [2] was proposed as a guideline for web service choreography to define and represent complex behaviour of the set of collaborating services. Reliable and large-scale interoperation among trading partners is being attempted by creating a semantic web for each trading partner's service whose properties, capabilities, and interfaces are encoded in an unambiguous, computer-understandable form [8, 9, 10]. The most noticeable result from these efforts is OWL-S specification, a language for ontology definition, manipulation, and reasoning [11, 14]. OWL-S provides a mechanism to allow web service autonomy for identifying operational metrics at the design stage and hence facilitates heterogeneous web services discovery and integration. A number of Business Process Modelling methods like SAP's Event Process Chain (EPC), IMG AG's Promet and the Communication Structure Analysis (CSA) known from Bonapart have also been proposed. Each of these methods graphically represents the process flow but are coupled with the underlying platform.

These frameworks are working their way to combine Web services to create higher level and cross-organizational business processes that requires standards to model the interactions. But, there is an important missing piece to realize truly reliable and scalable processes of design and manufacturing over the web. In such a scenario choreography cannot be realized unless the cooperating entities do not share similar semantic views of the processes involved. Existing frameworks are closely tied to a specific domain and platform. Therefore there is a need for a semantic model that would enable frameworks to manage and support dynamic manufacturing process design through collaborative web services.

3. Process Grammar

Our semantic model uses the terms, process grammar for process representation and task decomposition. Process grammar [3, 6] has been proposed to represent design and manufacturing process and to generate process flow dynamically. Process flow graphs describe the information flow of a design methodology, and process grammars provide the means for transforming high-level task into progressively a more detailed set of tasks as well as selecting a method among many alternatives for a task.

In process grammar, the process flow graph consists of two types of entities: tasks and data specifications. A task is a single unit of design activity as defined with the process flow diagram. The flow diagram shows how to compose a task and the input and output specifications of the task. Data specifications are design data, where the output specification produced by a task can be consumed by another task as an input specification. There are two types of tasks, a logical task and an atomic task. A logical task can be decomposed into a set of subtasks. An atomic task is the simplest form of the task, which cannot be decomposed any further. Invoking an atomic task represents the execution of an application program or a specific tool. In essence, an atomic task is defined as an encapsulated tool.

The process grammar provides an abstraction mechanism so that designers are not overly burdened with details. It allows a user to represent and manipulate a small number of abstract, higher-level tasks that can be expanded into detailed, executable alternatives. This can be especially valuable when engineers from different disciplines are working together on a project. Process grammar can also support the dynamic, iterative nature of the design process. During the execution of a process, if the execution of certain task does not meet the requirement, a roll back can occur to an appropriate point and a new production can be applied to generate alternative process flow dynamically.

4. Description of SOM

The Service-Oriented Process Model (SOM) of MIDAS is the key semantic model that enables a contract initiator in actively locating a distributed design process flow, collaborating on discovered process flow, and generating an optimized collaborative workflow. The ultimate goal of SOM is to provide a standardized way to understand distributed workflows and its executions among heterogeneous systems. Effective representation of semantics of the participating process and components are required to realise the above-mentioned goal. To realize such a goal, SOM specifies a semantic definition of distributed design process and process flow execution in terms of service flow and service flow execution.

SOM is written with OWL (Web Ontology Language), which is a standard language to describe semantics for Web resources [14, 20]. OWL-based Web service ontology, supplies Web service providers with a core set of mark-up language constructs for describing the properties and capabilities of Web services in unambiguous, computer-interpretable form so that the participating entities could have comprehensive information about each other's capabilities. From the semantic representation of process, it is possible that each collaborating unit understands each other partner's heterogeneous process representations. SOM consists of three sub models: Process Definition Model, Process Enactment Model, and Execution Monitor Model.

4.1 Process Definition Model

Generically, a model describes the way in which a service works and what happens when it is invoked. Process definition model defines the semantics for a service provider's process flow in the context of a service flow. The process definition model regards each task as a *Service*. Figure 1 shows our semantic model, SOM.

SOM follows syntactic specification of process grammar. So, Service can be classified into two types namely, *Atomic Service* and *Logical Service*, which are analogues of *Atomic Task* and *Logical Task* of process grammar. A service model should also define the pre conditions and post conditions along with the expected inputs and outputs of a service. Service includes *Input* and *output Specification* and *Pre-* and *Post-Condition*. MIDAS framework had earlier utilized such marks when a service provider posts the service flow at the registry, and a service requester selects a service that meets requirements from the registry.

Process definition model has the Service Composite as a placeholder for service provider's service flow. Service composite consists of a set of component services along with the task dependencies between component services. The dependencies between component services are captured by *linkTo* and *linkFrom* properties of component services. Logical service has alternative choices of service composites. The Alternative Choice encapsulates multiple service composites inside. Each dependent component service links each other by linkTo/linkFrom. Services in a Service Composite share semantic marks of service. Figure 1 (a) shows ontology graph of process definition model.

4.2 Process Enactment Model

Process enactment model provides standardized view of processes to be called subsequent state changes to be made upon calling. Process enactment model stipulates seven standard operations, which are essential to gear up operations of heterogeneous distributed design system. These operations are described as follows.

- *Provide input* delivers input data to a service.
- *Invoke enactment* brings a cue to start process enact ment
- *Retrieve workflow graph* generates workflow graph from a service to a viewer.
- *Retrieve output* transports output data from a service to a viewer.
- *Execute task* carries out applying of production even t or tool execution event.
- *Rollback* instantiates rollback event by user to a serv ice.
- *Enforced rollback* delivers abort event by user to a s ervice.

Process enactment model also defines five basic exe cution states. Figure 1 (b) illustrates the relationship am ong standard operations and execution states.

- *un-initialized* indicates that nothing has been initialized in a service.
- *ready* shows that input data has been bound to a service, but service execution is not invoked yet.
- *proceeding* points out that execution of a task has been invoked and keeps on going.
- *finished* is the state that execution of a service has been finished. Two sub states are *success* and *fail*.
- *exception* indicates that unexpected event has occurred during proceeding state.

4.3 Process Monitor Model

Process monitor model provides standardized view of how to capture and deliver the distributed design process execution to viewers. Users of MIDAS can monitor a process by capturing traces of task executions and data binding through Execution Monitor Model. Figure 1 (c) shows ontology graph of process monitor model. Process monitor model follows the process enactment logic of Process Grammar. A logical task will be accompanied by a service composite as a mark of execution. A tool will do an atomic task, in the same manner. According to the mathematical model of Process Grammar, the execution of logical task means an applying of a production to the logical task, while the execution of atomic task means execution of a given tool for the atomic task. Input specifications are bound to input data before task execution, and output specifications are bound to output data after task execution.

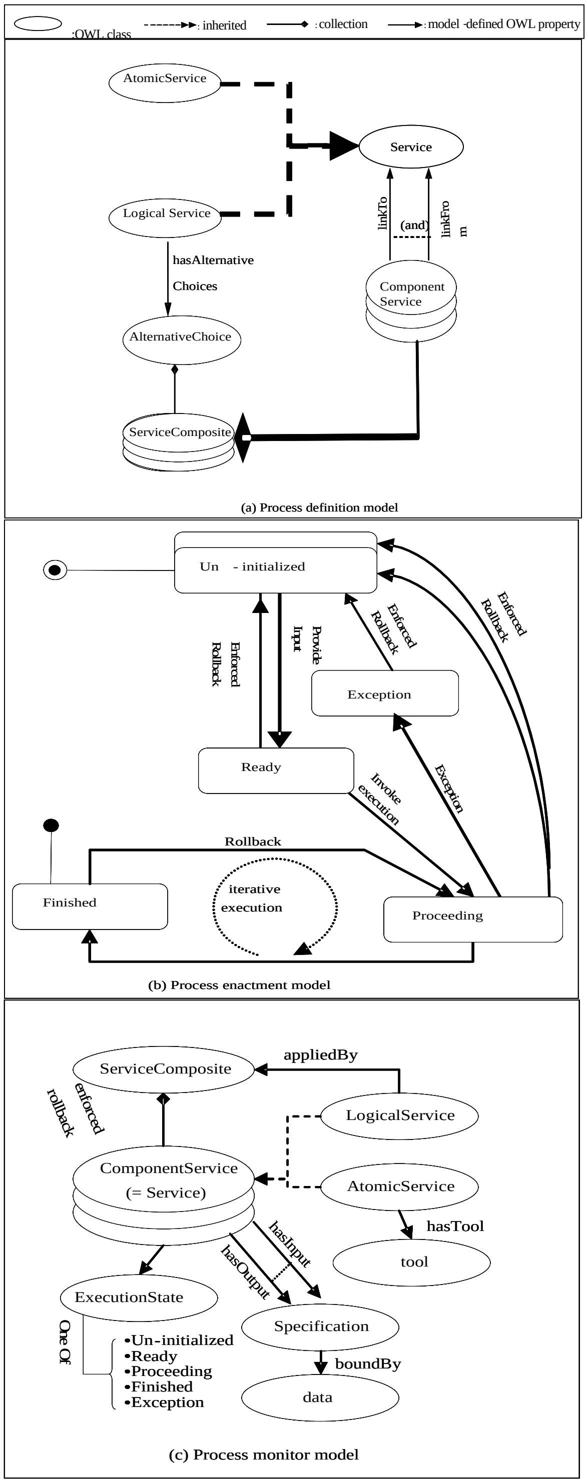

Figure 1: Three sub models of SOM

5. Distributed Design with MIDAS

The MIDAS framework is a collaborative engineering framework that coordinates various tasks in design and manufacturing using web services. It provides a means 1) to locate manufacturers dynamically, 2) to select and make contracts with particular manufacturer in agreement with requirements, 3) to create the collaborative process by incorporating distributed services among manufacturers, and 4) to provide a flexible and interoperable execution environment for the collaborative process. The distinct features of MIDAS are:

- Separation of process specification from the execution environment. Syntactic structures, such as dependency among tasks and input output requirement, together with alternatives are specified using the process grammar. Execution details and constraints are encoded as a part of execution environment.
- A task is a unit of an activity in a distributed design process. A task can be accomplished by service providers, or by someone in the same organization. There may be several different alternatives of accomplishing the task, with each service provider its own alternative. The service provider can be located within the organization itself or a company where the task can be outsourced.
- Guiding the designer to select appropriate processes and service providers. Through user interaction, the framework generates a process configuration that provides an optimal solution within a given set of constraints.
- Process flow generated is modulated and archived for future use. Archived process flows and alternatives can be retrieved, revised and reused.

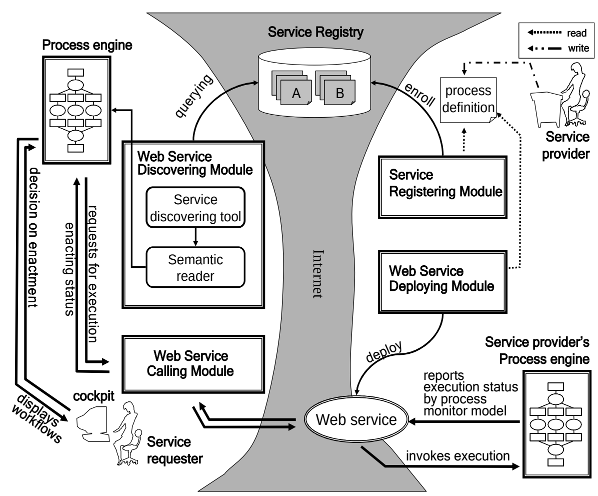

Figure 2: MIDAS framework

We have integrated service-oriented distributed design functionality into MIDAS framework. MIDAS has been developed for collaborative entities to work together to complete a manufacturing design process [6, 7]. Figure 1 illustrates general MIDAS framework. The system consists of four components as follows.

- *Cockpit*: Cockpit in MIDAS provides the following main functionalities: Process Editing, Display and maintenance of process information archive, Operations of process design.

- *Process Engine*: They provide services to distribute the process design execution snapshot, through process library and tool library.
- *Process/Tool Library*: Process/Tool information is organized in production libraries.
- *Web Service Module*: It provides facilities such as browsing service semantics, and calling Web service to the engine. MIDAS web service module is composed of Web Service Discovering Module, Service Registering Module, Web Service Deploying Module, and Web Service Calling Module.

Distributed design workflow generation by MIDAS proceeds in interaction between manufacturing companies. For each alternative for sub process generation, there is a manufacturing service provider who can be outsourced. Service providers offer their manufacturing process as a service flow, and MIDAS guides a designer to select an appropriate service provider and his service flow. The figure 4 illustrates how companies use MIDAS to outsource a distributed design tasks. The contract initiating company performs a referencing for distributed design task as shown in the figure 3. After selecting one service provider, the two companies negotiate, and reach an agreement on cooperation. Incorporating subcontractor's process and execution of merged process flow will be simply done by calling subcontractor's Web services.

Distributed design by MIDAS framework proceeds in two major steps: (1) Service deployment and registration by geologically distributed manufacturers. (2) Iterative workflow configuration among distributed manufacturers.

5.1 Service Deployment and Registration

Service provider writes process definition in OWL following the process definition model schema. Process definition includes product description, manufacturing process for the product, required input and output specification, and etc. Based on this definition, the service-deploying module of service provider automatically generates Web service code and deploys it on the Web. Then, the service-registering module enrols deployed Web service on the Web service registry. Service registry registers not only the Web service, but also semantic marks of manufacturing process that the Web service provides.

5.2 Iterative Workflow Configuration

In order to reflect dynamic nature of process management, MIDAS provides support for iterative process configuration including service discovery and process enactment. MIDAS process enactment consists of four steps: (1) Locate and select service provider and his process. (2) Expand initial process with selected process. (3) Execute expanded process. (4) Check if expanded process meets constraints, and if not, rollback to step 1. Figure 3 represents each of the steps mentioned above.

In step 1, service requester checks the semantic marks of processes registered in the service registry, and selects one that meets the requester's requirements. Through step 2 to step 4, all operations are done by collaboration among service requester and provider. This collaboration follows the enactment model of SOM. In step 2, importing service provider's OWL process definition expands service requester's process. In step 3, the service requester invokes process execution by calling *invoke enactment* operation of service provider's Web service. During the execution, the service provider finishes distributed design tasks in his process and generates output data. MIDAS allows service requester to participate service provider's process execution. The service provider can participate the process execution by calling *execute task* and *rollback* operations. In step 4, the service requester validates finished process and result data. The service provider can get the finished process by calling *retrieve workflow graph*, and can get the result data by calling *retrieve output data*. If the finished process doesn't meet given distributed design requirement, the service provider can reinitiate another process configuration with new service provider. During the iterative workflow configuration, the service requester can check the execution status by calling retrieve workflow graph operation. This operation returns execution state of each element of service provider's process in format of OWL document defined in the process monitor model of SOM.

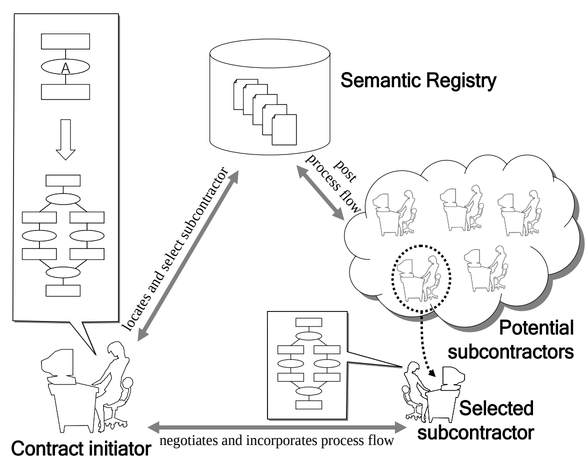

Figure 3: task outsourcing on MIDAS

6. Conclusion

This paper presents an all purpose semantic model and a framework architecture of a web-services based process management system for collaborative design and manufacturing. Our service-oriented process model provides a unique functionality of web service based process composition and iterative process enactment. Processes are designed with modular structure. Process logic is constructed by compositing modular process logics through choreography of Web services. We have implemented a prototype using Java and OWL. The collaborative dynamic process design and management of MIDAS is purely task-oriented, which gives Web services choreography capability to each task, not to a central process management unit in each cooperative organization. The main advantage of the framework is that it is truly distributed architecture, which enables it to exploit loosely coupled heterogeneous networks, and hence the framework facilitates distributed design between totally different types of partners. In addition, SOM provides generic collaborative dynamic process management and the same has been demonstrated using MIDAS. SOM helps MIDAS in effectively monitoring dynamic process environment and reconfiguring process logic by iteration.

7. References

[1] AberdeenGroup, "Beating the competition with collaborative product commerce," June 2000.

[2] Arkin, A., Askary, S., Fordin, S., Jekeli, W., Kawaguchi, K., Orchard, D., Pogliani, S., Riemer, K., Struble, S., TakacsiNagy, P., Trickovic, I., and Zimek, S., "WSCI", http://www.w3.org/TR/2002/NOTE-wsci-20020808

[3] Baldwin, R. and Chung, M.J. "Design Methodology Management: A Formal Approach," *IEEE Computer*, pp. 54-63, February 1995.

[4] Bourke, R. "Software Survey: Collaborative Product Commerce, " *Midrange Enterprise*, November 2000.

[5] Chung, M.J., and Kwon, P. "A Web-based Framework for Design and Manufacturing a Mechanical System," *DETC*, Atlanta, Georgia. Sep. 1998.

[6] Chung, M.J., Kwon, P. and Pentland, B. "Making Process Visible: A Grammartical Approach to Managing Design Proc esses," *ASME Transaction, Journal of Mechanical Design*. vol . 124, 364-374, 2002.

[7] Curbera, F., et al., "The Next Step in Web Services," *Communication of the ACM*. 46(10): p. 29-34. 2003.

[8] Ding, Y., Fensel, D., Klein, M., and Omelayenko, B., "The semantic web: yet another hip?," *Data & Knowledge Engineering*, Vol. 41, No. 2, pp. 205-228, 2002.

[9] Fensel, D., Horrocks, I., Harmelen, F., McGuinness, D. L., and Patel-Schneider, P. F., "The semantic web OIL: an ontology infrastructure for the semantic web", *IEEE Intelligent Systems & Their Applications*, Vol. 16, No. 2, pp. pp. 38-45, 2001.

[10] Hendler, J., "The Semantic Web - Agents and the Semantic Web," *IEEE Intelligent Systems & Their Applications*. Vol. 16, No. 2, pp. 30-37, 2001.

[11] IBM, "Using Service-Oriented Architecture and Component-Based Development to Build Web Service Applications", *Rational Whitepaper*, 2003.

[12] Lavana, H., Khetawat, A., Brglez, F., and Kozminski, K., "Executable Workflows: A Paradigm or Collaborative Design on the Internet", *Proceedings of the 34th ACM/IEEE Design Automation Conference*, June 1997.

[13] Leymann, F., "WSFL 1.0", http://www-3.ibm.com/software/solutions/webservices/pdf/WSFL.pdf

[14] Paolucci, M., Srinivasan, N., Sycara, K., Solanki, M., Lassila, O., McGuinness, D., Denker, G., Martin, D., Parsia, B., Sirin, E., Payne, T., McIlraith, S., Hobbs, J., Sabou, M., and McDermott, D., "OWL-S", http://www.daml.org/services/owl-s/1.0/owl-s.pdf

[15] Peltz, C. "Web Services Orchestration and Choreography," *IEEE Computer (October)*, pp 46-52, October 2003.

[16] Schey, J. A., (1987), *Introduction to Manufacturing Processes*, 2nd edition, McGraw-Hill, New York, NY

[17] Shapiro, R., "A Comparison of XPDL, BPML, and BPEL4WS." xml.coverpages.org/Shapiro-XPDL.pdf, 2002.

[18] Weerawarana, S. and Francisco, C., "Business Process with BPEL4WS: Understanding BPEL4WS", Part1", http://www-106.ibm.com/developerworks/webservices/library /ws-bpelcol1/ 2002.

[19] WSFL. http://www-3.ibm.com/software/solutions/webservices/pdf/WSFL.pdf

[20] OWL www.daml.org

[21] Christian Fillies, Frauke Weichhardt, Semtalk, "Towards the Corporate Semantic Process Web," http://www.semtalk.com

[22] Burdick, D. "Collaborative Product Commerce: The Technology Vision," *Research Note Technology by Gartner Group*, January 4, 2000.