International Journal of Business, Human and Social Sciences
ISSN: 2517-9411
Vol:5, No:2, 2011

Multi-Agent Model for Automation of Business Process Management System Based on Service Oriented Architecture

Soe Winn, May Thwe Oo

***Abstract*—**Business process automation is an important task in an enterprise business environment software development. The requirements of processing acceleration and automation level of enterprises are inherently different from one organization to another. We present a methodology and system for automation of business process management system architecture by multi-agent collaboration based on SOA. Design layer processes are modeled in semantic markup language for web services application. At the core of our system is considering certain types of human tasks to their further automation across over multiple platform environments. An improved abnormality processing with model for automation of BPMS architecture by multi-agent collaboration based on SOA is introduced. Validating system for efficiency of process automation, an application for educational knowledge base instance would also be described.

***Keywords*—**Business process management system, business process automation, multi-agent collaboration, Service Oriented Architecture, extensible service application

I. Introduction

THERE are several requirements and changes ever occurring in various enterprise business environment. Those requirements and changes make business process enterprise required to be sufficient efficiency, seamless effectiveness and loosely coupled comprehensively.

A SOA based BPMS, normally, is that services providers provide satisfiable services to service requesters. BPMS should be perfectly arranged several services with different task for various service requesters. WS-BPEL is predominant one for orchestration of web service implementation for BPMS, which could automate tasks or enable integration for required business process. For optimizing business process, BPMS use incremental manner and reorganizes basic services. Therefore, the BPMS would provide converting rigid and isolated applications and data into flexible and deployable component for interactive collaborating between business processes and main system [1]-[2].

Agent-Oriented Programming (AOP) is a relatively new software paradigm that brings concepts from the theories of artificial intelligence into the mainstream realm of distributed systems. AOP essentially models an application as a collection of components called agents that are characterized by, among other things, autonomy, proactivity and ability for distributed communication.

JADE (Java Agent DEvelopment Framework) is a software framework of fully implemented in Java programming language. It simplifies the complex implementation of multi-agent systems through a middle-ware that through a set of graphical tools that supports the debugging and deployment phases. The agent platform can be distributed across machines (without sharing the same OS) and the configuration can be controlled via a remote GUI. The configuration can be even changed at run-time by moving agents from one machine to another one, as and when required [4].

We use ontology as a means of enabling human task support and automation in model. Enabling communication and knowledge sharing by capturing a shared understanding of terms that can be both by humans and by programs would by defined by ontology [8], as DAML, CGs, OIL, DAML+OIL, and OWL [7]-[9]. Multi-attribute decision-making is also important topic for BPMS since decades [10]. Different human task owners are represented by agents and a multi-agent system (MAS) is a system composed of cooperative or competitive agents that interact with one another in order to achieve individual or common goals [11], Implementation of Web services with agent technology, in order to realize complex interaction and coordination of services [12]-[13].

Multi-agent collaborating for making enterprise business processes more efficient and enterprise IT framework more nimble, we focused our business process automation model by configuring our paper with the following parts, abstract of paper, brief introduction of technically requirements and related works, fundamental aspect of SOA based BPMS [6], internal infrastructure and overview of model, an application of model in technically and process automation aspect, conclusion for this paper and references, which we prepared for this paper.

II. SOA Based BPMS

Process automation, faster transition and well-behaved environment are fundamental intension of BPMS. It does not code itself for business process information and rules into application directly, but separates them from application systems and places it under the control of enterprise system [3]. It would also a great advantageous for any environment either familiar with technology as developer or graphical users to create, manage, deploy and optimize process through the back-end functions supported by SOA. There are four parts in the model for interacting, processing, managing and exception handling for the system.

A. Interactive Layer

The layer for interconnection of the whole system, which

describe product reviewing, resources exploring, online shopping, business directory searching, process monitoring, performance managing and personal information identifying, and so on.

B. Processing Layer

Mandatory layer of the business process management system and it implement all require processes for workflow process management, process querying, level investigating, transmission controlling and task assigning.

C. Data Management Layer

Database resources for require libraries are included in data management layer. It undertakes resources managing, exception library handling and updating unpredictable exception after exception description.

D. Process of Abnormality Processing

To improve the exception occurrence events of system, exception-handling mechanism based on processing abnormal library would be integrated for the lacks of the ability of security accuracy and reliability standard.

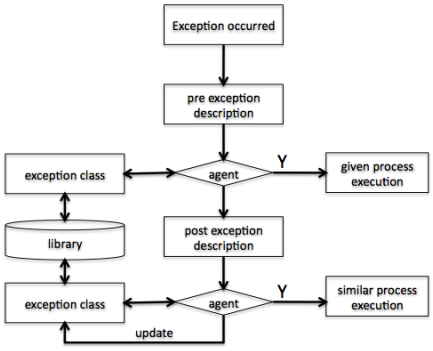

Fig. 1 process of abnormality processing

Most of BPMS that handle simultaneous processes and processing failure in distribution and heterogeneous environment have being experienced vulnerability of seamless processing. Principle of similarity match and reusing past experience action of multi-agent model would improve adjusting the execution process of the system.

III. Internal Infrastructure and Overview of Proposed Model

The internal infrastructure of the proposed model was included four mandatory parts. They are:

A. Task description and managing,
B. Business process controller,
C. Rule definition and verification, and
D. Updating response and exception.

A. Task Description and Managing

Task description is for instantiating process and performing specific tasks of the process. BPEL processor perform the require actions for tasks and module creation, executing and process of BPEL workflow instances. All of the task description and managing in the proposed model are handled in workflow engine which provide either executing environment for workflow processes or scheduling and allocating internal or external enterprise resources during the operation of business process executing. Therefore the sufficient efficiency and execution performance of BPMS are being improved by the effective efficiency and reliability of workflow engine.

B. Business Process Controller

Workflow models need to be in line with business process models that capture the operational business processes [14]. Engine controller is the control center and core part of business process model. The business process model controller make the explanation of the definition of process, creation process instance and controls its executing, scheduling various types activities, adds work item into worksheet for customer and calls application through application programming interface and those stand for its functions.

The procedure of business process model are receiving request of controlling business process from external interface and transfer different types of requests to corresponding modules and return results by schedule centre while task management is responsible for task creating, task state transforming and data maintaining under the control of it. Task management will be triggered to construct a new pended instance for subsequent activity when a task is finished. At the same time, other external request can also use task management module to switch tasks' state. Task assignment selects a group of staff who can execute the task accord to the basis of assign principle firstly and then make sure which individual in the group can execute the task and marked it according to assign plan. At last the marked individual will be recorded in the field of corresponding record. Dependence examine module examines pre-dependence of activities. Schedule centre will examine pre-dependence before a task switched into ready state, and only those tasks satisfying the examining condition can be switched. Transmit control module works according to back-regulation, which defines relationship between current activities and follow-up activities defined in business process model. Start control module control the starting of a task and monitor the tasks' state.

C. Rule Definition and Verification

Rule engine activates business rule and executes proper operation according to the logic in business rule. The operation of rule controller is as follow:

1. Rule Definition Process

TABLE I
State Of Rule Verification

State	Verification	Testing	Result
0	unverified	no tested	
1	verified	no tested	
2	verified	tested	pass

The initial state is unverified and not tested when a new business rule is defined and set ‘0’ rule state. Rule state set as

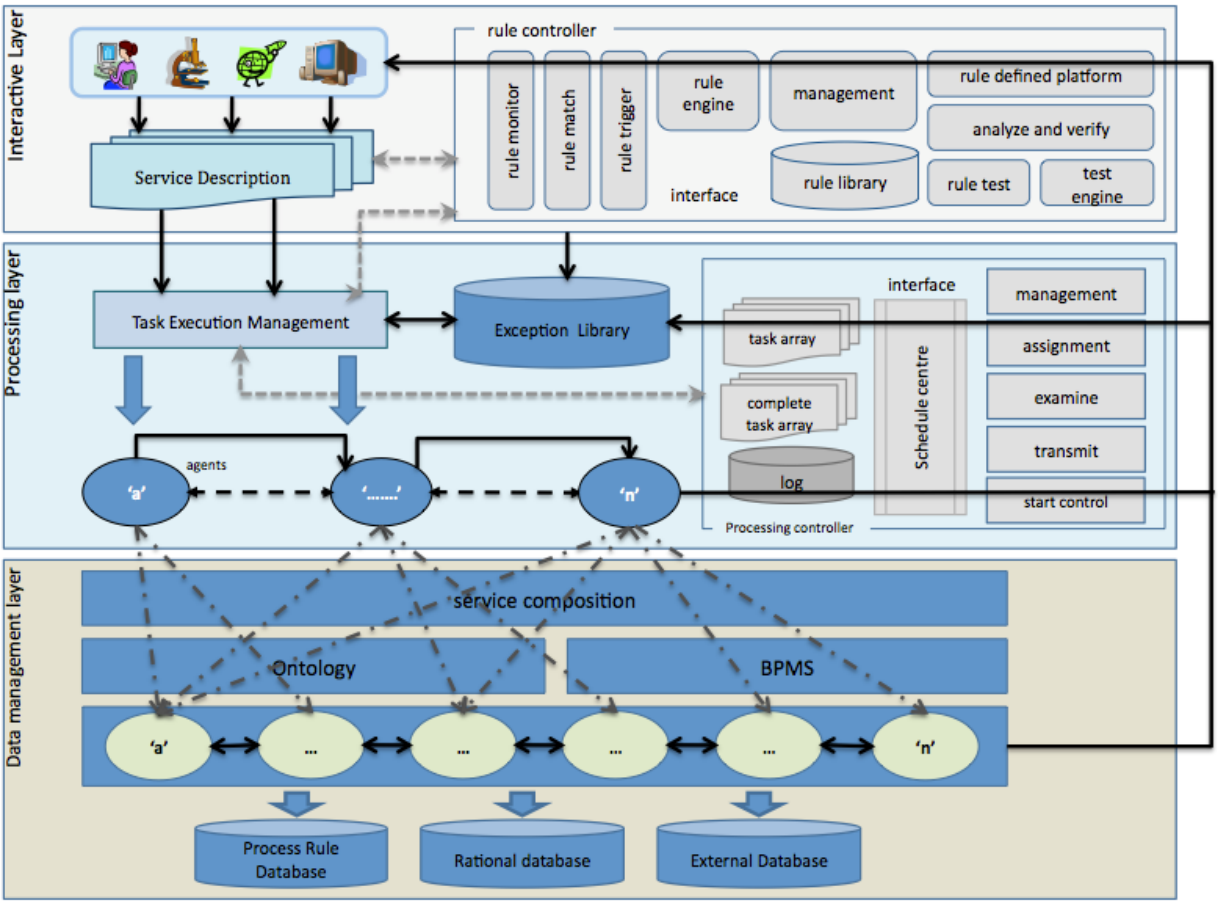

Fig. 2 overview of the model

'1' when rules are set verified and not tested to business rule management system by rule analyze and verify module if validated. Validated rules are set as '2' after tested by rule test module at rule test engine and saved into XML rule library.

2. Process of rule triggering

BPMS encapsulate requiring date of business rule into validated format and transport it into business rule engine through data interface. Business rule matches data and put them into executing array of rule engine if satisfy. The rule was activated from executing array by executing engine and could be processed its features through the rule monitor.

3. Authority setting

Hence the security of business rule is one of the significant components of any control system of enterprise business, the different level of authority would be set. Users define a new business rule or implement current business rule by using authority of business rule defining and realizing for entering the visualization of rule-defined platform. Authority of triggering a business rule was examined by triggering node respectively.

D.Updating Response and Exception

Most of BPMS at heterogeneous environment have security vulnerability for guarantee accuracy and reliability when operating simultaneous process or processing failure. Therefore, an exception handling mechanism based on processing abnormal library was presented in the model.

When the system encounters an exception, exception class would be automatically defined by exception handling mechanism based on processing abnormal library. Exception class was put forward according to case-based reasoning for exception verification. Post exception class would be defined unless verification would support properly process execution in accordance with the principle of similarity match by past experience and predefined exception. Otherwise, a new rule of the process will be defined to handle the abnormal exception based on the exception handling mechanism by operational way of defining similarity. Therefore, process execution procedure according to exception library of abnormality processing system would be preferred and exception handling mechanism update the exception class to process abnormal library for making further exception handling smoothness of better intelligent BPMS.

At business scenario, internal or external clients would initialize the services, which are described by automatic service description. Task execution management would prepare the services, which are put forward by service description for matching and ruling process in rule engine and business process controller. The agents implement process,

which was verified and tested by rule definition, and make execution of the process. The process the agents created would be interactive layer again and human task or automation task of decision-making is directly going to response to business process management system. The process would define for further process and library from database management layer will update automatically. The new process with new rule would be implemented by agents and described for brokers or agents, which have more available resources or environment. Interactive layer was responded several levels of resources by business process controller and library of resources are updated for references for further processing. During the process, exception library is occasionally updated by process of abnormality and process would be forwarded for seamless execution.

IV. An Extensible Dictionary Application of Model

We would like to consider an instance scenario. Several dictionary applications are available around as. Most of them are just simple dictionaries and for every day use only. Another difficulty is we have to wait until new version release for what we could not find in old version, but it is also not the solution what difficulty we are facing for. Moreover, we could rarely see perfect technical dictionary or computer dictionary in specific language with advantageous of extensible, updatable or so. Literally, creating dictionary thing is really time consuming and several discussion needed work. Extensible dictionary is one that you can extend easily without modifying its original code base. We could enhance its functionality with new plug-ins or modules.

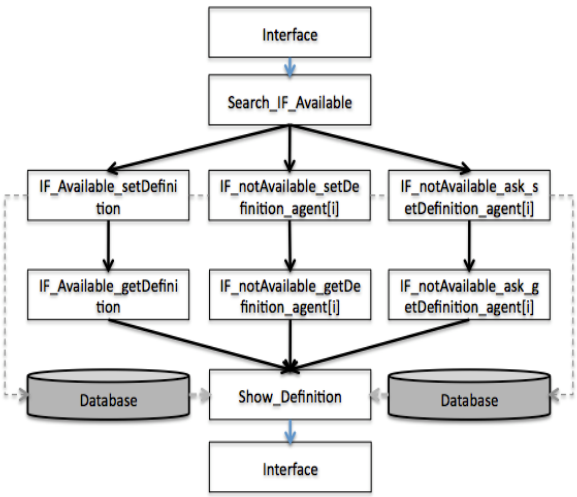

Fig. 3 process management flow layout

Developers, software vendors, and even customers can add new functionality or application programming interfaces (APIs) by simply adding new extension application onto the application-specific extension directory. By designing an extensible dictionary application, which allows you or others to provide service implementations that require no modifications to the original application, we provide an easy way to upgrade or enhance specific parts of a dictionary without changing the core application. Therefore, the general purpose of the extensible dictionary is that making available any specific dictionary of any specific language by providing open source extensible process management framework for every environment.

For this purpose the overall process of extensible dictionary application process management system is show in figure 3 and the implementation program of the workflow can be configure as follow.

```
<process:CompositeProcess rdf:ID="Extensible_Dictionary">
  <process:composedOf>
    <process:sequence>
      <process:components rdf:parseType="Collection">
      <process:AtomicProcess rdf:about="#Search_IF_Available"/>
      <process:AtomicProcess rdf:about="#Search_IF_NotAvailable"/>
      <process:AtomicProcess
          rdf:about="#Search_IF_NotAvailable_Ask"/>
      <process:AtomicProcess rdf:about="#Show_Definition"/>
      </process:components>
    </process:sequence>
  </process:composedOf>
</process:CompositeProcess>

<process:AtomicProcess rdf:ID="Search_IF_Available">
  <process:hasInput rdf:resource="#WordName"/>
  <process:hasInput rdf:resource="#WordType"/>
  <process:hadInput rdf:resource="#WordCategory"/>
  <process:hasInput rdf:resource="#DateOfUpdate"/>
  <process:hasInput rdf:resource="#ProcessLog"/>
</process:AtomicProcess>

<process:Input rdf:ID="#WordName">
  <process:parameterType rdf:resource="&concepts;#Name"/>
</process:Input>
<process:Input rdf:ID="#WordType">
  <process:parameterType rdf:resource="&concepts;#Type"/>
</process:Input>
<process:Input rdf:ID="#WordCategory">
  <process:parameterType rdf:resource="&concepts;#Category"/>
</process:Input>
<process:Input rdf:ID="#DateOfUpdate_In">
  <process:parameterType rdf:resource="&concepts;#Date"/>
</process:Input>
<process:UnConditionalEffect rdf:ID="#ProcessLog">
  <process:ceEffect rdf:resource="&concepts;#Log"/>
</process:UnConditionalEffect>

<process:AtomicProcess rdf:ID="Show_Definition">
  <process:hasInput rdf:resource="#setDefinition"/>
  <process:hadOutput rdf:resource="#getDefinition"/>
</process:AtomicProcess>

<process:Input rdf:ID="setDefinition">
  <process:parameterType rdf:resource="&concepts;#Definition"/>
</process:Input>
<process:Output rdf:ID="getDefinition">
  <process:parameterType rdf:resource="&concepts;#Definition"/>
</process:Output>
<process:UnConditionalEffect rdf:ID="ProcessLog">
  <process:ceEffect rdf:resource="&concepts;#Log"/>
</process:UnConditionalEffect>
```

The preceding program is showing process structure and a situation for normal searching. All of the processes have their own ID like “Search_IF_Available” for normal searching situation, “Search_IF_NotAvailable” for retriving available similar resources and “Search_IF_NotAvailable _Ask” for updating by end user.

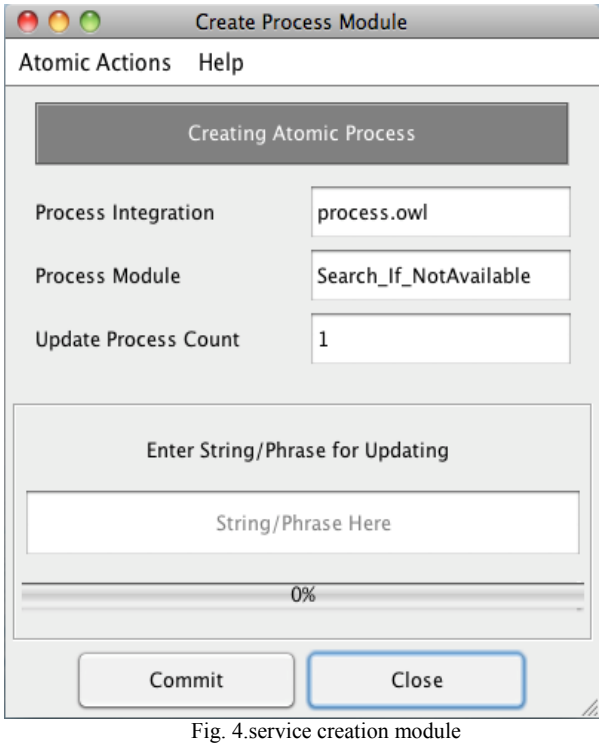

Fig. 4.service creation module

Let's assume we have a process for normal searching, all of the service on process was instantiated and retrieved available result from relational database. The process would be update for further new process at the end of the process. Process Log would be thrown if the process occurs exceptions and system will update again.

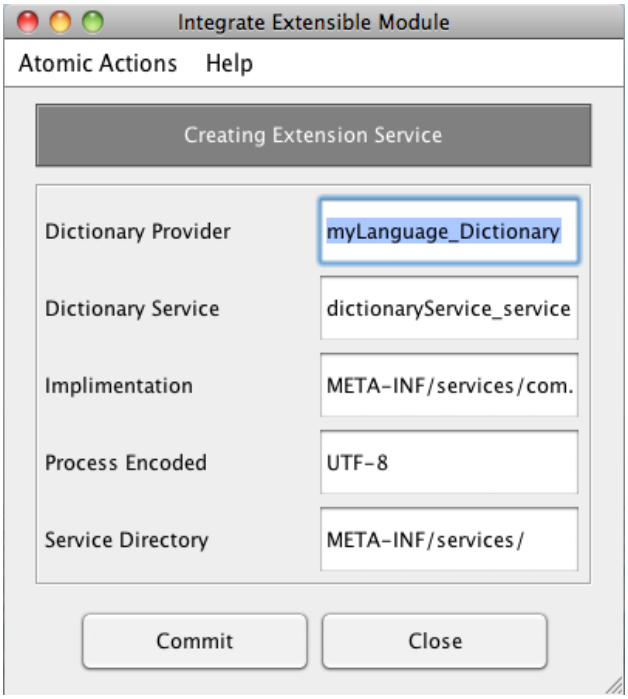

Fig. 5 Integration of Extensible Module

Therefore, the process structures for another two "NotAvailable" situation would be like the following program.

```
<process:CompositeProcess rdf:ID="Extensible_Dictionary">
   <process:composedOf>
      <process:Sequence>
         <process:components rdf:parseType="Collection">
            <process:AtomicProcess rdf:about="#Search_IF_Available"/>
            <process:AtomicProcess
                rdf:about="#Search_IF_NotAvailable"/>
            <process:AtomicProcess
                rdf:about="#Search_IF_NotAvailable_Ask"/>
            <process:AtomicProcess rdf:about="#Show_Definition"/>
         </process:components>
      </process:Sequence>
   </process:composedOf>
</process:CompositeProcess>

<process:AtomicProcess rdf:ID="Search_IF_NotAvailable">
   <process:hasInput rdf:resource="#WordName_Get"/>
   <process:hasInput rdf:resource="#WordType_Get"/>
   <process:hasInput rdf:resource="#WordCategory_Get"/>
   <process:hasInput rdf:resource="#DateOfUpdate"/>
   <process:hasInput rdf:resource="#ProcessLog"/>
</process:AtomicProcess>

<process:Input rdf:ID="WordName_Get">
   <process:parameterType rdf:resource="&concepts;#getName"/>
</process:Input>
<process:Input rdf:ID="WordType_Get">

   <process:parameterType rdf:resource="&concepts;#getType"/>
</process:Input>
<process:Input rdf:ID="WordCategory_Get">
   <process:parameterType rdf:resource="&concepts;#getCategory"/>
</process:Input>
<process:Input rdf:ID="DateOfUpdate_Get">
   <process:parameterType rdf:resource="&concepts;#Date"/>
</process:Input>

<process:UnConditionalEffect rdf:ID="ProcessLog">
   <process:ceEffect rdf:resource="&concepts;#Log"/>
</process:UnConditionalEffect>

<process:AtomicProcess rdf:ID="Search_IF_NotAvailable_Ask">
   <process:hasInput rdf:resource="#WordName_In"/>
   <process:hasInput rdf:resource="#WordType_In"/>
   <process:hasInput rdf:resource="#WordCategory_In"/>
   <process:hasInput rdf:resource="#DateOfUpdate"/>
   <process:hasInput rdf:resource="#ProcessLog"/>
</process:AtomicProcess>

<process:Input rdf:ID="WordName_In">
   <process:parameterType rdf:resource="&concepts;#InName"/>
</process:Input>
<process:Input rdf:ID="WordType_In">
   <process:parameterType rdf:resource="&concepts;#InType"/>
</process:Input>
<process:Input rdf:ID="WordCategory_In">
   <process:parameterType rdf:resource="&concepts;#InCategory"/>
</process:Input>
<process:Input rdf:ID="DateOfUpdate_In">
   <process:parameterType rdf:resource="&concepts;#InDate"/>
</process:Input>

<process:UnConditionalEffect rdf:ID="ProcessLog">
   <process:ceEffect rdf:resource="&concepts;#Log"/>
</process:UnConditionalEffect>

<process:AtomicProcess rdf:ID="Show_Definition">
```

```
    <process:hasInput rdf:resource="#setDefinition"/>
    <process:hasOutput rdf:resource="#getDefinition"/>
</process:AtomicProcess>

<process:Input rdf:ID="setDefinition">
    <process:parameterType rdf:resource="&concepts;#Definition"/>
</process:Input>
<process:UnConditionalEffect rdf:ID="getDefinition">
    <process:ceEfect rdf:resource="&concepts;#Definition"/>
    </process:UnConditionalEffect>
```

As the preceding programs, the new process is further mapped at the execution level once a process has done. New web service for "NotAvailable", means in relation database, would be available. Once the translation of the process is completed the process will again be ready and available for execution and hence serving the client requests. It is the way of our system maintains to keep loosely coupled environment. There are also service loader class and dictionary provider implementation sections we had left because out of scope of the publication. However we got strong enough flow level description and could make extension for any specific dictionary of any specific language without modifying original service and application.

V. Conclusion

In the era of automatic business transaction and IT based enterprise solution exploring, many educational organizations in industry and civilian government start deploying business process management technology and systems with expecting the dramatic operational efficiency improvement [5] on their business and knowledge-based administrative environments. SOA provides a good solution for business process management and heterogeneous distributed environments. It describes and deploys application by standard protocol and interfaces. It makes enterprise application based on uniform norm which conducive to the interaction between different applications. BPMS is an enormous system, which concerns the whole enterprise development strategy, operations and the IT infrastructure, and it depends on enterprise and every IT staff to build jointly. Although we tried at a good concentration of BPMS and enterprise solution, we also should a good care of security level of every transaction we make. Therefore tightly security integration would be considered for further BPMS integration arena.

References

[1] MOORE J., "*Business Process Management*", Chemical Engineering Process, American, 2003, pp. 22-23.
[2] Gartner, Growing IT's Contribution: The 2006 CIO Agenda, Garther EXP, American, 2006.
[3] Dirk Krafzig, Karl Banke, *Enterprise SOA -- The Best Practices of Orienting Service Architecture*, Tsinghua University Press, Beijing, 2006.
[4] Fabio Bellifemine, Giovanni Caire and Dominic Greenwood, "Developing multi-agent systems with JADE", John Wiley & Sons, Ltd, 2007.
[5] Haeng-Kon Kim, Roger Y. Lee, Hae-Sool Yang, "Frameworks for secured business process management systems", Proceedings of the Fourth International Conference on Software Engineering Research, Management and Applications (SERA'06)
[6] Dan Luo, Jianghua Jiang, Buyun Sheng, Mingzhong Yang, "Research on Business Process Management System Based on Service Oriented Architecture", 2008 IEEE.
[7] Ana Sasa, "A Model for Business Process Automation in Service Oriented Systems with Knowledge Management Technologies", 2010 IEEE 6th World Congress on Services
[8] L. F. Lai, "A knowledge engineering approach to knowledge management," *Information Sciences*, Vol. 177, No. 19, Oct. 2007, pp. 4072-4094.
[9] (2009) W3C, OWL 2 Web Ontology Language Structural Specification and Functional-Style Syntax, http://www.w3.org/TR/owl2-syntax.
[10] M. Bohanec and V. Rajkovič, "Multi-Attribute Decision Modeling: Industrial Applications of DEX," *Informatica*, Vol. 23, No. 4, Oct. 1999, pp. 487-491.
[11] B. Henderson-Sellers and P. Giorgini (Eds), Agent-oriented Methodologies, Idea Group Inc., Hershey, PA, 2005, ch. P. Giorgini and B. Henderson-Sellers, "Agent-Oriented Methodologies: An Introduction."
[12] Y. Li, K-M Chao, M. Younas, Y. Huang, and X. Lu, "Modeling emarketplaces with multi-agents Web services," In *Proc. 11th Int. Conf. on Parallel and Distributed Systems, Fukuoka, Japan, 2005*, pp. 175-181.
[13] T.I. Zhang and H. Jiang, "A Framework of Incorporating Software Agents into SOA," In Proc. Artificial Intelligence and Soft Computing (ASC 2005), Benidorm, Spain, 2005.
[14] "Business Process Management Architecture" pp. 305-343.