Multi Agent Model for Execution of Workflow Patterns

[1]Tarini Prasad Panigrahy, [2]Sankarsan Sahoo, [3]Prasanta Kumar Dash, [4]Chandan Patra, [5]Monoj Kumar Sahu

[1]Associate Professor, [2,3,4,5]Assistant Professor
Department of Computer Science,
Gandhi Institute for Technological Advancement, Bhubaneswar, India

Abstract: **Web service composition is the process of creating a new service by combining a set of available services in the system. Service compositions are mainly two types: static web service composition and dynamic web service composition. Static web service composition is done at the design time and dynamic composition is done at the run time based on user's request. To address the dynamic web service composition we use Service based Inter-organizational Workflow (SIW) and Agent Based Architecture which can dynamically integrate the workflow and compose a workflow execution community customized to different workflow specifications. In our work, each of the agent capability is used to serve a particular service request. In our approach, business processes are wrapped by service agents. Based on the user's need, the process agent contacts the service selection agents to find appropriate service agents, and then negotiates with the service agents about task execution.**

IndexTerms **- Service based inter-organizational Workflow, business process, agent based architecture.**

I. Introduction

Service composition is the process of creation of a new service by combining available services of the systems. It can be done mainly in two ways: Static web service composition is done during the design time by the developer. Dynamic web service composition is done at the runtime by agents. Dynamic service composition provides many benefits and flexibility as new services can be selected on the fly. Dynamic service composition and workflow technology is essential for effective B2B integration. "A workflow system automates a business process, in whole or in part, during which documents, information, or tasks are passed from one participant to another for action, according to a set of procedural rules" [1].

So in order to support workflow interaction when an organization incorporates the services of others within its own processes we introduce Service based Inter-Organizational Workflow (SIW) and is similar to the notion of virtual enterprises [2][3]. A fundamental problem to be addressed in SIW is the coordination of the different business processes involved in it. By coordination, we indicate all the work required to put all these processes collectively in order to provide the global common goal in a proficient manner. The use of web services [4] for functional specification and interactions has achieved a great deal of attention at present. The SOAP[5] is XML based communication protocol to access the web services over the network using transport protocol like HTTP. WSDL [6] specifies the mechanisms to access a Web service over the network. WSDL files are stored in Universal Description Discovery and Integration (UDDI) as a registry so that a web service requester can locates them. It is an XML based language which provides syntactic description of web services. UDDI [7] contain description of web services and mechanism for requester to get access to the services publication and description. It contains data with mixture of white, green and yellow pages. White pages enclose information like company name, address, contact information etc, yellow pages enclose information like business category, business type etc; while green pages contains which kind of services the business presents. Since these UDDI registries have relatively straight-forward access methods (i.e. the locate _service and acquire _service methods in their specification) so, how to automatically maintain the coordination of different business processes involved in an SIW? How to face the unavailability at run-time or the not knowledge at design-time of an organization responsible for the execution of a business process involved in an SIW?

Automatic coordination in dynamic SIW raises many specific problems as given below:

• Finding partners, that consists in, for a requester organization, choosing one or Several provider organizations ready to execute a requested business process.

• Negotiation of a business process between the requester organization and therefore the previous chosen provider organizations. Negotiation criteria may be as varied as due time, quality of the business process, price, and visibility of its evolution and method of executing it. The results of this step is that the identification of the provider organization responsible of the requested business process.

•Negotiation between the requester and therefore the selected provider aiming at defining execution rules of the requested business process.

• Distributed execution of the business process and its monitoring.

Regarding connected literature, we can find some interesting propositions handling dynamic coordinatation [8][9] in SIW. Unfortunately, these propositions solely provide coordination architecture while not specifying the interaction rules or protocols between the various elements of the architecture. In case of SIW where partners are not necessarily known before or during the execution either for the reason that they are unknown at design-time or no more available during the execution time. Therefore it is necessary that the different Workflow Management Systems (WFMS) involved in a dynamic SIW support specialized tools and interface infrastructure to face this highly dynamic environment. Agent technologies represent a promising solution to realize the workflow composition and management of services in a dynamic environment. Software entities have been defined as agents when they possess certain characteristics. The grouping of these characteristics has been defined as levels of agency [10]. So to address this we use our Agent Based Architecture [11] as middleware architecture to implement the SCW framework. This MAS architecture under the control of a workflow management system, used to effectively integrate cross-organization workflow. This MAS consists of a team of software agents, collectively performing a task which could not be performed by any individual agent and further do the negotiation to choose an effective team of agents to complete the tasks of the workflow schema [12][13].So, having an agent view of the coordination issue in dynamic SIW, we could inherit from numerous concrete solutions projected in this area to deal with agent coordination[14] .

In our Agent Based Architecture we use five types of agents namely interface agent, service agent, service selection agent, process agent and monitor agent. For each workflow instance these agents form an agent community to execute the workflow.

Interface Agent provides the interface through which the user can interact with the Agent Based Architecture system. It submits the XML document for the workflow schema to the process agent to execute the workflow. Service agents are used to abstract the business processes from their physical organizations. Service selection Agent is used to search a specific service in web-based environment which is large and highly dynamic. The service agents advertise and hence register their offerings like identity, capabilities and constraints in a meta data repository (UDDI). The meta data is used to locate the service agents. The main task of the process agent is to transform a workflow specification into a workflow instance. It gets the workflow specification from the interface agent and makes a query to the service selection Agent to find the suitable service agents and then integrates the service agents to execute the workflow. A monitor agent monitors the actual execution of a given task at the site of the service provider.

In this paper we bring together the dynamic service composition for a set of distributed web services and static composition for the system that do not change business requirement ,usually for application with in a company

The SIW environment described in this paper incorporates the interoperability of general web services.Fig.1, illustrates the SIW environment for a supply chain Management (SCM).

In a SCM process where the manufacturer initiates the business process once it receives the order from the customer. The manufacturer has internal services for managing price calculation, Logistics arrangements and production scheduling after it receives the raw materials. However, manufacturer depends upon the other third-party Suppliers (here supplier1, supplier2 and supplier3) for the raw materials in order to manufacture the Item as required by the customer. The suppliers for supplying item2 and item3 register their offerings as web services in a distributed registry, such as a UDDI registry. The manufacturer uses these registry services as a part of its internal workflow. In this case, the manufacturer has a static connection with the partner organizations and is able to access services directly. Thus there are two elementary service types considered in our approach, invocation-based and third party services. The third party webservices are accessed through the middleware Agent Based Architecture.

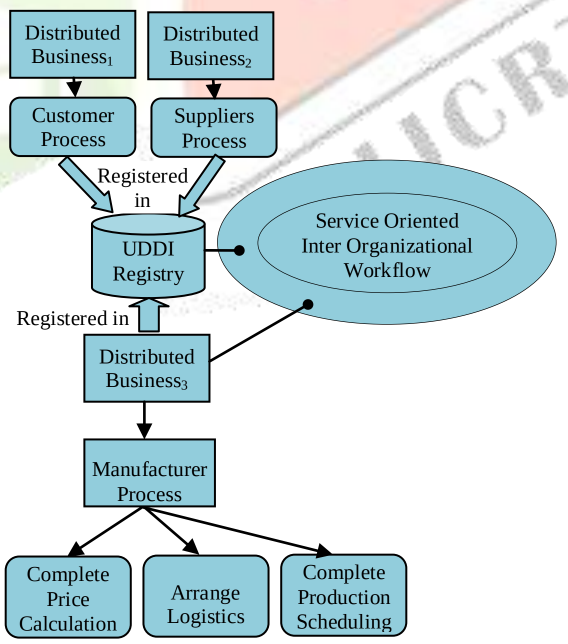

Fig 1: An Example of SIW environment

The invocation types of services, at this point, are accessed using WSBPEL In this work, several questions must be answered to support the formation of the SIW environment:

1. How business process are described through workflow patterns
2. Based on users demand how to integrate the dynamic and constantly changing business processes?

3. In web service environment, how to run a process successfully when there are predefined service providers that are no longer available to response to service requester

In order to answer these above questions, we use Agent Based Architecture as an agent realization mechanism. This agent realization determines the best configuration of agents for the specific cross-organizational workflow process.

The paper is organized as follows. Section 2 describes Multi agent system (MAS) Based SIW Composition Model. The paper continues in Section 3 by describing a SCM process workflow and a composition pattern for our agent based infrastructure. Section 4 discusses how the process agent coordinates with other agents to select a workflow execution plan. Lastly in Section 5 we compare our proposed model to other related works and then conclude the paper giving some guidelines for future works.

II. MAS Based SIW Composition Model

The Details of Agent based workflow Architecture is given in our previous work [11].The Fig.2 shows Web service composition in agent based Workflow Architecture

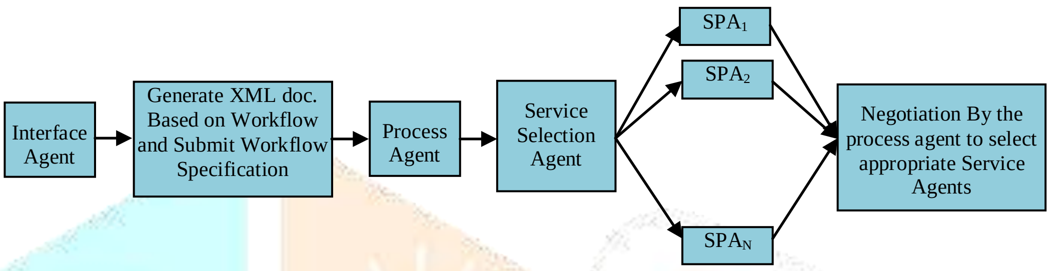
Interface Agent
Generate XML doc. Based on Workflow and Submit Workflow Specification
Process Agent
Service Selection Agent
SPA1
SPA2
SPAN
Negotiation By the process agent to select appropriate Service Agents

Fig 2: Web service composition in agent based Workflow Architecture

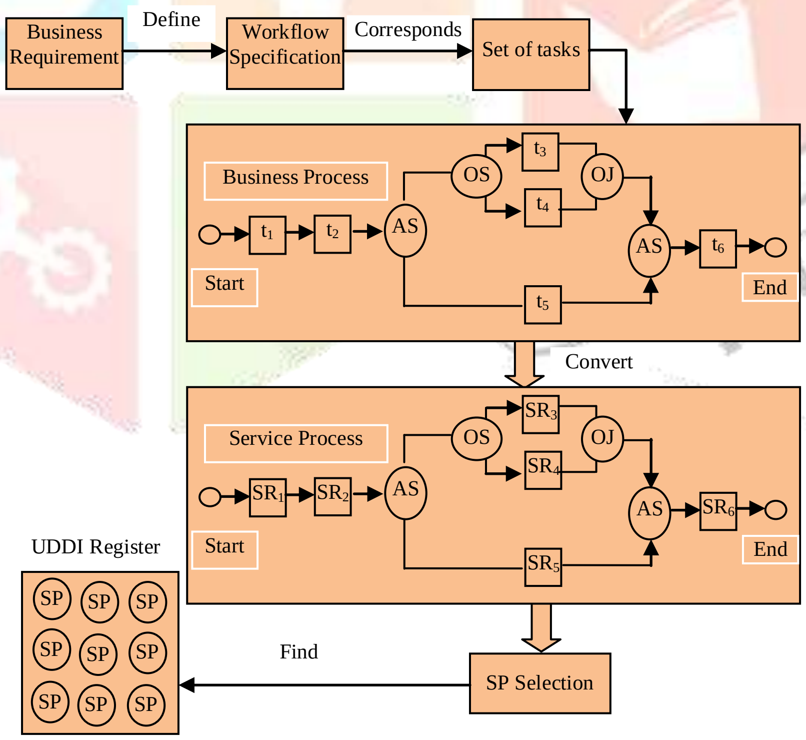

Fig 3: Step by Step dynamic web service composition.

It is through the interface agent the process composer and the end user access the system. The process composer defines the workflow schema and the end user is to create and start workflow instances, control and monitor the execution of workflow instances. The interface agent provides a tool to define the workflow schema, which is represented by UML state chart diagram. Once the process composer draws the UML state chart diagram for the workflow schema, the end user generates an XML document for it. Further the end user provides the parameters required to select the appropriate service agents to execute the workflow. The XML document is submitted to the process agent to execute the workflow.

The process agent is responsible for transforming the workflow specification to a workflow instance. Fig 3 below shows a hierarchical process of dynamic web service composition. Based on the workflow specification the composition process has several tasks or activities as Task1, Task2, Task3, … Taskn. These tasks are mapped to Service-Request activities(SR) in the Service Process. The Process agent makes a query to the service selection Agent to find the candidate service provider agents (SPA) for the different atomic tasks or activities. Then it negotiates with the service agents about task execution. When the process agent receives many choices to perform a particular task, it allows the user to do selection among the available service agents. Finally, it makes connections to that service agent and assigns the task specifications to it.

III. Business Processes Composed of Workflow Patterns

Workflow patterns have been used[15] as a baseline to evaluate process languages such as BPEL4WS, XLANG, WSCI, BPML.Van der Aalst [16] retains a repository of atomic workflow patterns which define common workflow operations required to implement business processes. We go together with these earlier assessment approaches by using workflow patterns to offer formal descriptions of common operational procedures of the underlying business process. Benatallah and others in 2002[17] discussed numerous workflow patterns for several agent-based workflow composition. These workflow patterns are parallel split, normal sequence, synchronization, exclusive choice and simple merge. In parallel split, many activities are executed concurrently. In the normal sequence pattern, modelers must specify the basic sequence of activities (i.e. order of service executions).In synchronization the different parallel threads must be synchronized or one path is chosen from many alternative choices (called as exclusive choice). Lastly, business process modelers can specify when two paths are merged (called as simple merge). It is not in the scope of this work to show all possible workflow patterns.

A. Business Processes & Workflow Patterns for our SIW

This business process shows the composition of services that perform a simple supply chain management scenario where a customer orders for an item to a manufacturer. In order to supply the ordered item, the manufacturer depends for raw materials that might be supplied by various suppliers like Supplier3 and either by Supplier1 or by Supplier2 .Once the manufacturer receives the raw material from Supplier3 and one form either Supplier1 or Supplier2 the other suppliers process(out of Supplier1 and Supplier2) is cancelled based on certain criteria(non-functional) and the manufacturer, manufactures the product and supplies to the customer. In this simple scenario, three workflow patterns can be extracted, these are: <<PARALLEL SPLIT>>, <<DEFERRED-CHOICE>>, and <<SIMPLE-MERGE>> stereotypes.

In this business process, the Customer Interface actor makes order-item1 to the manufacturer Interface actor. Succeeding services are dependent on the successful completion of this service (<<SYNCHRONIZATION>> pattern).The Manufacturer Interface actor makes a parallel split to enquire for the raw-material to various suppliers: Supplier1 Interface actor, Supplier2 Interface actor and Supplier3 Interface actor. Delivery-Raw-Material service is executed concurrently by each supplier. The Manufacturer Interface actor accepts the Raw-material from Supplier3 Interface and one of either Supplier1 or Supplier2 Interface actors (where some criteria is used) through a simple merge. The choice from Supplier1 or Supplier2 is made by a deferred choice pattern.

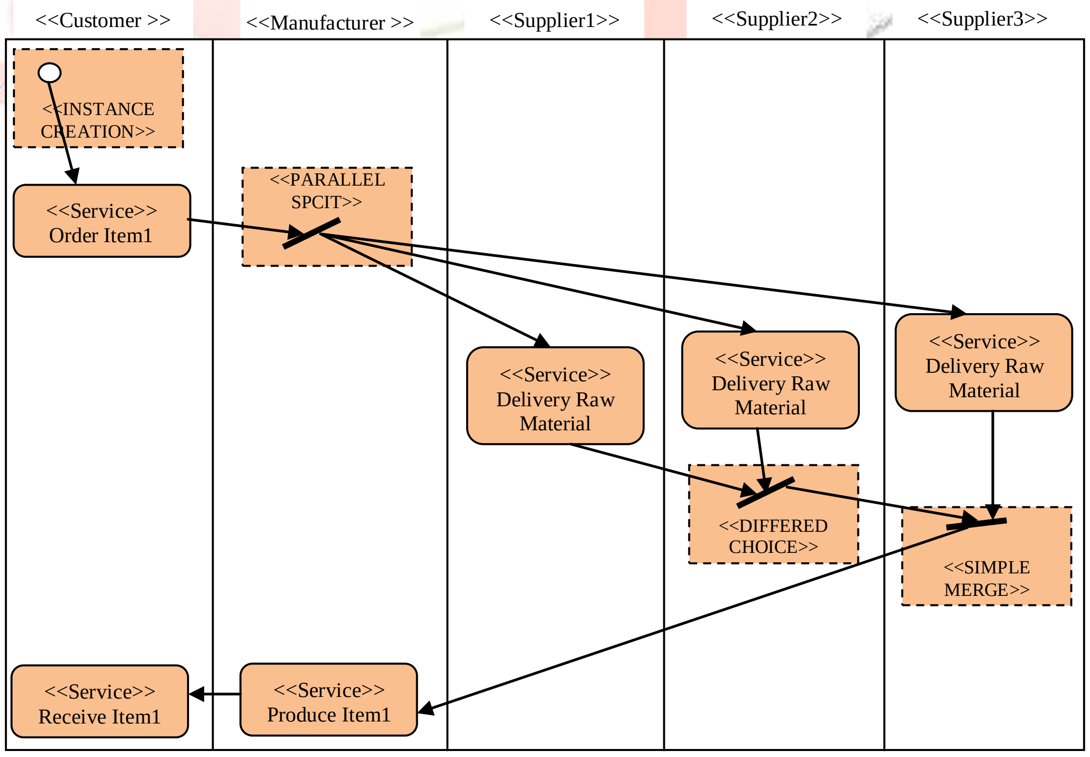

Fig 8. Supply Chain Business Process Composed of Workflow Patterns

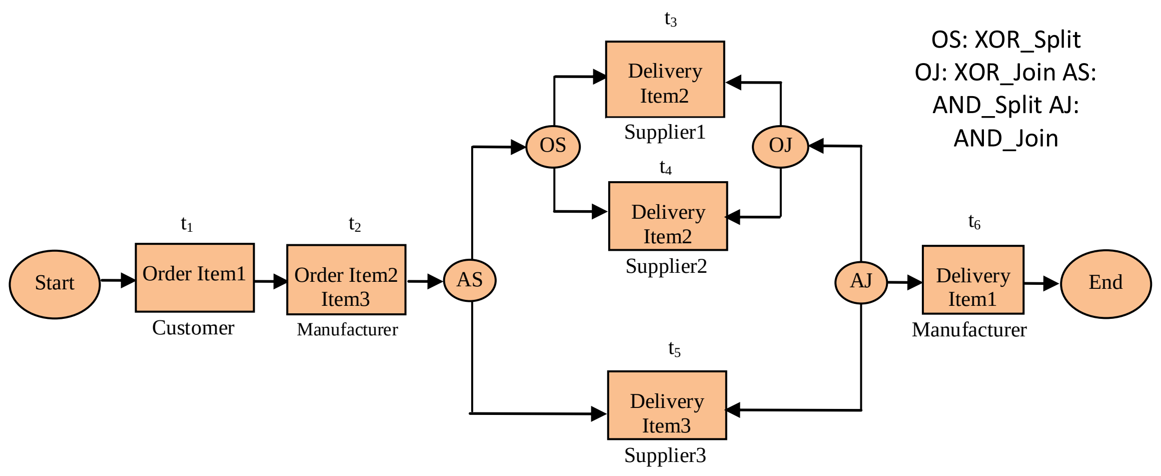

Fig 9: SCM Workflow pattern

Consider our SCM workflow application as shown in Fig 9. It involves inter-action among agents from various stakeholders. In order to find the most efficient design of an agent-based business process management system, we must analyze and evaluate the different workflow plan.

IV. Workflow Execution Planning

We start with interface agent submitting the workflow specification to process agent. Further the service agents wrap the business process from their respective organizations and have registered with the service selection agent. In this part we talk about how the process agent coordinates with other agents to pick a workflow execution plan.

A Parsing workflow specification based on Service Agents

As per the workflow specification of fig 7, the workflow pattern contains 6-tasks. The interface agent submits this workflow specification (W) to the process agent. Thus:

W= [t1, t2, t3, t4, t5, t6]

For each task ti the process agent finds the suitable service agents which can execute the task ti by querying through the service selection agent. The search result is a set of service agents SPAi. Once the process agent finishes the queries for each task ti in W the process agent gets a set of service agents for each task:

t1=SPA1= [spa1, spa2]
t2=SPA2= [spa2, spa3, spa4]
t3=SPA3= [spa3, spa5, spa6]
t4=SPA4= [spa4, spa7]
t5=SPA5= [spa5, spa8, spa9]
t6=SPA6= [spa1, spa7, spa10]

Here SPA= [SPA1, SPA2, SPA3, SPA4, SPA5, SPA6] …………………….. (1)
Or, SPA= [spa1, spa2…….spa10] ………………………………………… (2)

In (1) or (2) contains all the service agents spai those are responsible to execute certain task in the workflow W. For each spai in SPA, putting all the tasks that the service agent spai can execute, we can have

Spa1= [t1, t6]
Spa2 = [t1, t2]
Spa3 = [t2, t3]
Spa4 = [t4]
Spa5 = [t3, t5]
Spa6 = [t3]
Spa7 = [t4, t6]
Spa8 = [t5]
Spa9 = [t5]
Spa10 = [t6]

After grouping the service agents by the tasks they can execute, we can have

$\underbrace{[spa1, spa2, spa3, spa4]}_{SPA'1} = \underbrace{[t1, t2]}_{T1'}$ …………………. (1)

$\underbrace{[spa3, spa5, spa6, spa8, spa9]} = \underbrace{[t3, t5]}$ …………………. (2)

SPA'2 T2'

$$\underbrace{[spa1, spa4, spa7, spa10]}_{SPA'3} = \underbrace{[t4, t6]}_{T3'} \qquad \ldots\ldots\ldots\ldots\ldots\ldots\ldots (3)$$

B Negotiation between the process agent and service agents

For (3), (4) and (5) the process agent starts a negotiation session so as to negotiate with every service agent in SPA'i about executing the tasks T'i .Our adopted negotiation protocol constitutes 3-steps as described below:

Request-for-Bid: The negotiation session begins with the process agent sending a request-for-Bid to each service agent in SPA'i .The request-for-Bid includes specification of the tasks and deadline for responding to the request-for-Bid.

Respond-With-Bid: Upon receiving the Request-for-Bid from the process agent, the service agent will decide whether to respond with a bid depending upon the task specification. If the bid is for a set of tasks then the cost and duration for each task must be specified. We assume that the process agent has to accept either the whole bid or reject it.

Counter-Bid: Once the process agent receives the bids, it will respond the service agent with a counter-Bid where it bargains for the execution cost and the execution duration of a task. The different bargaining strategies the process agent adopts are as follows:

User-Friendly policy: Where based on user’s preference the process composers can define ECA rules to specify the counter-Bid. For example in a SCM process (when the number of bidders is more than 5.) the customer may need that cost and time for supplying Item1 should be within Rs.50000 and 2-days respectively. The ECA rule for this may be:

{Receive-Bid(b)}[Num(Bidder,Supply-Item1)>5]/Send-Counter-Bid (Rs.50000, 2-days)

Experience-Based-Policy: Here the process agent keeps the records of past negotiation sessions for different tasks. It can generate the counter-Bid based upon the past negotiation. For an instance, the process agent can search the bids for the same task in past negotiation sessions, and then select a bid that raises for both minimal cost and duration. If such a bid does not exist then the process agent picks the bid which asks for minimal value of cost and duration. Then a counter-Bid will be created by requesting for discount.

In order to create the counter-Bid the process agent will use the Experience-Based-Policy in the absence of User-friendly policy. When the service agent receives the counter-Bid, it will decide whether to generate a new bid to the counter-Bid. The process of Bidding and counter-Bidding will continue till the time deadline provided by the process agent. The process agent finally confirms the bid by assigning the tasks to the service agents.

C Execution Plan generation

Once the negotiation session is over the process agent can have the bids for different service agents, assigned for different tasks during workflow execution. Now the process agent integrates these service agents to execute the workflow. Here we define the following concepts:

1) ***Execution Schema***: For our workflow W= [t1, t2, t3, t4, t5, t6], the execution schema(s) can be:

s= [T1', T2', T3']

where T1'= [t1,t2] , T2' = [t3,t5] and T3' = [t4,t6] ,

Here:

(i) $T'i \cap T'j = \phi$ where $i \neq j$

(ii) $T'i \subseteq W$

(iii) $\sum_{i=1}^{W} T_i' = W$

In this case all the task-sets in s are mutually disjoint subsets and the combination of all the tasks is the workflow W and each Ti' can be executed by different service agents.

2) ***Execution Plan***: For our execution schema s, the execution plan is:

p= [< T1', b1, SPA1>, < T2', b2, SPA2>, < T3', b3, SPA3>], here the service agents SPAi sends bids bi for task Ti'.

The process agent uses the Graph algorithm [31] to generate a set of execution schema(S) for our workflow W.

S=[s1,s2,s3,…..sn]

The process agent for each si in S, generate an execution plan based on the available bids. Thus:

P= [p1, p2, p3…pn]

Here P is the set of all execution plans those can be used to integrate the service agents.

3***) Execution plan selection***

To select an execution plan the following criteria will be used:

(i)Cost and Time: The most basic criteria that is used to select an execution plan’s(pi) execution cost C(pi) and total execution time t(pi) which are computed as follows:

(a)Total execution cost is the sum of the execution cost for n bids(cb1+cb2+…..cbn) those are responsible to execute the tasks in the workflow W.

i.e $C(pi)=\sum_{i=1}^{n} c\,bi$…………………….(6)

If the desired cost is Cdesired, then C(p)/Cdesired is expected to be 1 for a good execution plan.

(b) In order to compute the total execution time we use critical path algorithm [32] where the CPT analysis uses the total effective execution duration in a particular path that constitutes a set of service agents to complete the workflow execution. Thus t(pi) = CPT(t1b,t2b,t3b…..tnb) ………………(7)

Again if the desired duration is Tdesired , then t(p)/Tdesired is expected to be 1 for good execution plan.

(ii)Agent Community Size: The size of agent community (Sp) for an execution plan depends upon the number of agents required to execute the tasks in the workflow. Larger size agent community results in more communication overhead.

If the number of tasks in the workflow is Sw then Sp/Sw is expected to be 1 for a good execution plan.

(iii)Service Agent's Reliability: While selecting an execution plan we must consider the reliability of the service agents. The reliability of the service agent constitutes two issues:

(i)Whether it can complete assigned tasks

(ii)If so, then whether it can complete the tasks in time as specified in the bid.

If the reliability factor for a service agent is Rs, Then

$$\mathrm{Rs} = \frac{\sum_{i=1}^{n} \frac{t_i\ b}{t_i\ \mathrm{actual}}}{n} \quad \ldots\ldots\ldots\ldots\ldots.\ (8)$$

Here:

tib is the service agents task execution duration as specified by the service agent in the bid for the task ti.

tiactual is the actual execution duration that service agent takes on the execution of the given task ti .

The ratio of tib/tiactual is expected to be 1 or close to 1 for a good execution plan.For an instance, if a service agent S has Rs=0.97, then if the service agent S indicates that it can compete a task in 30 hours in the bid, we can estimate that the actual duration of the service agent is 30/0.97 approximately equals to

31-hours.

i.e testimate for S=31 hours.

Thus based on the reliability factor of the service agents, we can estimate the total execution time of a plan p as :

te(p)=CPT(testimate 1, testimate 2, testimate 3,……. testimate n)……(9)

4***) Evaluation of Execution plans***: The execution plans are selected by the process agent in two phases:

Clipping phase: This phase uses event-condition-action rules to eliminate some of the execution plans from the set of all execution plans (P). For example the event-condition-action rule to eliminate execution plans for which the total execution cost is more than Rs.50000 is represented in (10) and the event-condition-action to eliminate all the execution plans where the total execution duration is more then 2-days is represented in (11)

{Evaluating –Execution Plan(p)} [C(p)>Rs.50000]/Exclude(p)…(10)

{Evaluating –Execution Plan(p)} [te(p)>2-Days]/Exclude(p)……(11)

Loading Phase: The result of clipping phase is a set of execution plans P'(say). The clipped (filtered) execution plans (P') which are available after the clipping phase shall move through this phase. Here we derive a formula based on the criteria defined above. The formula is

$$B_{cts}\,(P) = \left({}^{C(p)}\!/_{C_{desired}}\right) \times L_c + \left({}^{t_e}\!/_{t_{desired}}\right) \times L_t + \left({}^{S_p}\!/_{S_w}\right) \times L_s$$

This formula helps the process composer to balance the cost, time and size of agent community. Here Lc,Lt,Ls are load balancing factors of cost, time and size of agent community and their values are between 0 to 100%.The process composer provides the values of Cdesired, tdesired, Lc, Lt, and Ls. For example if the work duration is the priority factor while choosing an execution plan, then we make Lt =0,Ls=0 and Lt=1.

5***) Execution of workflow***

Once the execution plan is selected, the process agent integrates the service agents to execute the workflow by building an agent community.

During the workflow execution the process agent is used to assist the service agents to exchange their data. The monitor agent continuously pings the service agents in order to check their failure. If a service agent has consumed the allotted execution time and still could not finish the task, the monitor agent pings it more often, as such service agents are more likely to fail. Monitor agent receives the message from the service agents when they have completed their tasks. Such messages are forwarded by the monitor agent to the interface agent (through the process agent) so as to make the user updated regarding the progress of the workflow execution. Once the workflow execution gets completed the agent community gets released and a new agent community is formed for a new workflow instance.

V. CONCLUSION

In our approach, we have implemented agent based workflow system for dynamic B2B integration. In our approach the agent-community for specific workflow is optimally and automatically composed based on the context of workflow execution and can self-adapt and react to changes during the execution. We show how the workflow agent-community gets constructed by taking the example of supply chain management. We illustrate how the agent community executes the workflow specification. We also used monitor agents for monitoring cross-enterprise workflows. This facilitates the end user to know the status of the workflow instance

.

REFERENCES

[1] Papazoglou, Michael.” Web Services: Principles and Technology.” s.l. PrenticeHall, 2008.

[2]M.B. Blake, "B2B Electronic Commerce: Where Do Agents Fit In? ", proceedings of the AAAI-2002 Workshop on Agent Technologies for B2B E-Commerce, Edmonton, Alberta, Canada, July 28, 2002

[3] C. Petrie, and C. Bussler, ” Service Agents and Virtual Enterprises: A Survey” IEEE Internet Computing,August 2003, pp 1-12

[4] Web Services (2002) http://www.w3.org/2002/ws/desc/

[5] http://www.w3.org/TR/SOAP

[6] W3C: Web Services Description Language (WSDL). http://www.w3.org/TR/wsdl(2003)

[7] UDDI (2003) http://www.uddi.org/

[8] Buhler, P., Vidal, J. (2005). Towards adaptive workflow enactment using multi-agent systems. Information Technology and Management, 6(1), 61–87.

[9] Blake, B., Gomaa, H. (2005). Agent-oriented compositional approaches to services-based cross organizational workflow. Decision Support Systems, 40(1), 31–50.

[10] Ferber, J., Gutknecht, O., Fabien, M. (2003). From agents to organizations: An organizational view of multiagent systems. In: Int. Workshop on Agent-Oriented Software Engineering, Melbourne, Australia, pp. 214–230.

[11] Tarini Prasad Panigrahy and Manas Ranjan Patra; An Agent-Based Model for Cross-Enterprise Supply Chain Management, Transactions on Networks and Communications, Volume 2 No 4, Aug (2014); pp: 107-129

[12] M. Wooldridge and N.R. Jennings, “Intelligent Agents: Theory and Practice,” In Knowledge Engineering Review 10(2), 1995.

[13] Wooldridge, M. (2002). AnIntroduction toMulti-Agent Systems. Wiley.

[14] Klusch, M., Fries, B., Sycara, K. (2006). Automated semantic web service discovery with OWLS-MX. In: Int. Conference onAutonomous Agentsand Multi-Agents Systems, Hakodate, Japan, pp. 915–922.

[15] Sibertin-Blanc, C. (1985). High level Petri nets with data structure. In: Int. Workshop on Petri Nets and Applications,Espoo, Finland.

[16] W.M.P. Van der Aalst, “Don’t go with the flow: Web Services composition standards exposed”, IEEE Intelligent, February 2003

[17] Workflow Patterns (2003): http://tmitwww.tm.tue.nl/research/patterns/patterns.htm

[18] Heiberg, T., Matskin, M., Pederseb, J. (2002). An agent-based architecture for customer services management and product search.Informatica, 13(4), 441–454.