NRC Publications Archive Archives des publications du CNRC

A framework for service enterprise workflow simulation based on multi-agent cooperation

Yang, F.; Shen, W.; Tan, W.; Ghenniwa, H.

This publication could be one of several versions: author's original, accepted manuscript or the publisher's version. / La version de cette publication peut être l'une des suivantes : la version prépublication de l'auteur, la version acceptée du manuscrit ou la version de l'éditeur.

For the publisher's version, please access the DOI link below. / Pour consulter la version de l'éditeur, utilisez le lien DOI ci-dessous.

<https://doi.org/10.1109/ICSMC.2010.5641910>

NRC Publications Record / Notice d'Archives des publications de CNRC:

<https://nrc-publications.canada.ca/eng/view/object/?id=f557cf80-70df-45a5-8666-b7fbe8f85c9c>

<https://publications-cnrc.canada.ca/fra/voir/objet/?id=f557cf80-70df-45a5-8666-b7fbe8f85c9c>

Access and use of this website and the material on it are subject to the Terms and Conditions set forth at

<https://nrc-publications.canada.ca/eng/copyright>

READ THESE TERMS AND CONDITIONS CAREFULLY BEFORE USING THIS WEBSITE.

L'accès à ce site Web et l'utilisation de son contenu sont assujettis aux conditions présentées dans le site

<https://publications-cnrc.canada.ca/fra/droits>

LISEZ CES CONDITIONS ATTENTIVEMENT AVANT D'UTILISER CE SITE WEB.

Questions? Contact the NRC Publications Archive team at

PublicationsArchive-ArchivesPublications@nrc-cnrc.gc.ca. If you wish to email the authors directly, please see the first page of the publication for their contact information.

Vous avez des questions? Nous pouvons vous aider. Pour communiquer directement avec un auteur, consultez la première page de la revue dans laquelle son article a été publié afin de trouver ses coordonnées. Si vous n'arrivez pas à les repérer, communiquez avec nous à PublicationsArchive-ArchivesPublications@nrc-cnrc.gc.ca.

<http://www.nrc-cnrc.gc.ca/irc>

A framework for service enterprise workflow simulation based on multi-agent cooperation

NRCC-53527

Yang, F.; Shen, W.; Tan, W.; Ghenniwa, H..

October 2010

A version of this document is published in / Une version de ce document se trouve dans:
2010 14th IEEE International Conference on Systems, Man, and Cybernetics
(IEEE SMC 2010), Istanbul, Turkey, October 10-13, 2010, pp. 1-8

The material in this document is covered by the provisions of the Copyright Act, by Canadian laws, policies, regulations and international agreements. Such provisions serve to identify the information source and, in specific instances, to prohibit reproduction of materials without written permission. For more information visit <http://laws.justice.gc.ca/en/showtdm/cs/C-42>

Les renseignements dans ce document sont protégés par la Loi sur le droit d'auteur, par les lois, les politiques et les règlements du Canada et des accords internationaux. Ces dispositions permettent d'identifier la source de l'information et, dans certains cas, d'interdire la copie de documents sans permission écrite. Pour obtenir de plus amples renseignements : <http://lois.justice.gc.ca/fr/showtdm/cs/C-42>

National Research
Council Canada

Conseil national
de recherches Canada

Canada

A Framework for Service Enterprise Workflow Simulation Based on Multi-Agent Cooperation

Fujun Yang^{1,2}, Weiming Shen^{1,2}, Wenan Tan³, Hamada H. Ghenniwa¹

¹Department of Electrical and Computer Engineering, University of Western Ontario, London, Ontario, Canada

²Centre for Computer-assisted Construction Technologies, National Research Council, London, Ontario, Canada

³School of Computer and Information, Shanghai Second Polytechnic University, Shanghai, P.R. China

fyang73@uwo.ca, weiming.shen@nrc.gc.ca, twajsj@sohu.com, hghenniwa@eng.uwo.ca

Abstract—*Service enterprise workflow simulation is an important approach to analyze service enterprise business processes dynamically. Traditionally, service workflow simulation is based on the discrete event queuing theory, which lacks flexibility and scalability. To address this problem, this paper proposes a service workflow simulation framework based on multi-agent cooperation. Social rationality of software agents is introduced into the proposed framework. Adopting rationality as a decision making strategy facilitates the implementation of flexible scheduling for activity instances. A simulation system prototype is developed and a business case study is conducted to validate the proposed framework.*

Keywords—*Workflow, simulation, multi-agent systems, social rationality*

I. INTRODUCTION

Most enterprise information systems can be viewed as discrete event systems (DES). The workflow represents the computerized business processes and workflow technology is one of the key techniques in current enterprise information systems. Therefore, service process modeling is the key for service systems and service management, and the service process workflow model is the core of service systems. As the analyzing tool for the workflow model, service process simulation can effectively support business process reengineering (BPR). Potential issues and bottlenecks of service processes can be detected through service process simulation before the deployment of a business management system. Simulation can reduce cost considerably as the result of the improved workflow model.

The workflow model is the abstract representation of business processes. Some workflow definition languages have been proposed and developed in the literature, such as WPD [1], IBM's FDL [2], and PSL (Process Specification Language) [3]. PSL, particularly, defines a neutral representation for manufacturing processes. Compared with other workflow research areas, service process simulation is still in its infancy. Discrete event simulation is the generic method used in service process simulation.

XPDL (XML Process Definition Language), which inherits WPD, is a standard for "interface I" defined by the Workflow Management Coalition (WfMC). The WfMC has identified five functional interfaces to a workflow service as part of its standardization program. "Interface I" is specified to support

Process Definition Import and Export, which includes a common meta-model for describing the process definition and an XML schema for the interchange of process definitions. The meta-model defines the objects and attributes contained within a process definition. One of the key elements of XPDL is its extensibility to handle information used by a variety of different tools.

In this paper, we propose a framework for service enterprise business process simulation based on multi-agent cooperation. The rest of the paper is organized as follows: Section 2 discusses discrete event simulation and agent technologies; Section 3 proposes a framework for business process simulation based on multi-agent cooperation, and discusses the process simulation architecture, decision-making strategies, and the multi-agent communication mechanism; Section 4 presents a prototype system, namely WfEmulator, that has been developed to validate the proposed architecture; Section 5 describes a case study to show the implementation of a simulation process; Section 6 gives some conclusions and discusses future work.

II. RELATED TECHNIQUES

A. Discrete Event Simulation for Workflow

The discrete event system is a discrete-state, event-driven system, that is, its state of evolution depends entirely on the occurrence of asynchronous discrete events over time. Most information systems in the service industry are discrete event systems, such as traffic systems and electric commerce systems.

Discrete event simulation (DES) is one way of building up models to observe the time-based behaviors of an enterprise system. The range of application areas is extremely large and there are numerous examples of the use of simulation in the service industry, manufacturing (batch production processes), and office environments.

The DES model involves a number of important concepts including:

- *Entities* are the tangible elements found in the real world.
- *Relations* link entities together, e.g., a part may be processed by a machine.
- *Simulation Executive* is responsible for controlling the

time advance and executing discrete events.

- *Random Number Generator* helps to simulate different data coming into the simulation model.
- *Result and Statistics* provide users a means of utilizing the simulation tool to gain a meaningful analysis of the model.

“Executive” is responsible for ordering the events. The executive removes the first event from the list and executes the relevant model logic. Any new events that occur as a result are inserted in the list at the appropriate point. The cycle is then repeated. A central clock is used to keep track of time.

When the DES is used in workflow simulation, activity instances are scheduled by the simulator according to the cooperative rules. The simulator will control the logical relationships between the activity instances and advance the clock to the new time. All the activity instances are stored in order in the execution queue of ready activity according to the scheduling rules, e.g., FIFO (First In First Out), HPFS (High Priority First Serve). This mechanism conflicts with reality when some activities have the same priorities or features. So, the SIRO (Service in Random Order) rule is usually used in the last step. In a large sized enterprise, different departments in the organization usually have different operation manners and scheduling strategies. For example, in a production department, most of the activities are scheduled basically on time, and sometimes based on work order priorities. While in another department, some activities may be scheduled by waiting time (retail department). A DES-based simulation system has to alter its scheduling strategies whenever the workflow model is changed. A new mechanism should be introduced to implement dynamic scheduling.

B. Multi-Agent Theory and Agent Social Rationality

“Agent” is a software entity which functions are proactive and autonomous in a particular environment. “Multi-agent system” (MAS) is a kind of intelligent system that interconnects separately developed agents, thus enabling the ensemble to function beyond the capabilities of any singular agent in the set-up [4].

There are two fundamental approaches used in modeling multi-agent systems: qualitative (some form of logic, e.g. the Belief, Desire, and Intention Model) and quantitative (e.g. Bayesian). “Utility theory” is a quantitative one to model MAS. “Utility function” is a mapping from states of the world to real numbers, indicating the agent’s level of happiness with that state of the world. Agents in the competitive MAS potentially have different utility functions.

In MAS, as to bounded resources and capabilities, the agent does not stand alone. In accordance with behaviors in reality, the agent must take actions based on certain strategies or rationalities. Traditionally, designers have sought to make their agents rational so that they can “do the right thing”. Rationality is how the rational decision is made among multiple strategies in the interaction of multi-agents [5].

The predominant theory of rational decision making in agents is that of the economic principle of maximizing the expected gain of actions [6]. Decision theoretic rationality dictates that the agent should choose an action which will

maximize the expected utility of performing that action given the probability of reaching a desired state in the world and the desirability of that state [7]. The action that maximizes individual utility may conflict with overall interest (social utility), or redundant actions could be taken due to a local utility preference. Hence, rationality needs to be considered, not only from the individual’s point of view, but also from the social perspective. Jennings and Campos proposed the principle of social rationality [8] as follows:

If a member of a respective society can perform an action whose joint benefit is greater than its joint loss, then it may select that action.

Here, joint benefit is defined as the benefit provided to the individual plus the benefit afforded to society as a result of an action. Similarly, joint loss is the individual plus societal loss of performing an action. Social rationality can be expressed as follows:

$$W_i(a_j) = \lambda_i u_i(a_j) + \lambda_{soc} \sum_{k \in \{1-i\}} u'_k(a_j) \quad (1)$$

Where $U_i(a_j)$ is the individual utility of agent i when it takes an action a_j , λ_i is the weight given to the individual utility of agent i ; $\sum U'_k(a_j)$ is the sum of utilities of other agents in the system when action a_j is taken by agent i , λ_{soc} is the weight given to the social utility part of the function. At a coarse level, equation (1) can be rewritten as:

$$U(i,j)=k_1*selfUtility(ps)+k_2*publicUtility(pp) \quad (2)$$

Where $U(i, j)$ is the utility of agent i when it takes action j ; k_1 , k_2 are the weight given to individual utility and public utility, respectively. ps and pp are the key influence parameters for individual utility function and public utility function, e.g. activity’s duration, waiting time, priority. The values of k_1 , k_2 can be altered to implement a wide range of decision-making strategies [9].

The concept of social rationality is to ensure that task planning proceeds when resource competition appears [10]. Social rationality can be used to guide an Agent’s decisions. In process simulation, when different activity instances could not share limited resources, competition appears. Thus, agent social rationality can be introduced into process simulation to represent the decision making strategies of organizations/departments. A related organization will prefer the activity instances which maximize their predefined rationality utility functions.

III. SIMULATION FRAMEWORK BASED ON MULTI-AGENT COOPERATION

A. Service Process Simulation Framework

The Multi-agent technique is used to address the issues of complex controls and solutions through intelligent behaviors such as cooperation, competition, and coordination in a set of autonomous agents under a dynamic distribution-oriented open environment [11]. These features of multi-agent systems (MAS) make them suitable for representing the entities in the enterprise environment. Compared to the discrete event

simulation, MAS is a more natural choice to build a simulation framework for an enterprise process model.

To achieve a flexible simulation structure, we model the workflow simulation structure as a multi-agent system, in which agents adopt rational utility functions as its scheduling strategies.

The proposed framework is illustrated in Figure 1. The proposed simulation framework consists of Process agent, Sub-Process agent, Activity agent, Resource agent and Organization agent. A relational database is necessary to support data import and export in the simulation process.

Figure 1. Service process workflow simulation framework based on multi-agent cooperation

Due to the complexity of reasoning, the intentional agent cannot react rapidly to the change, thus, it is not suitable in our simulation system. We use a self-controllable thread to represent the weak notation of Agent.

The core is the process agent, in which two lists are maintained: ready activity instance list and pre-running activity instance list. When all the former activities are completed, the instance is ready; when all the needed resources can be satisfied, the ready activity instance are pre-running, and awaiting the activity agent. The process agent records the information of all the activity agents, e.g., binding organization, utility value of running instance.

“Activity instance” is a plain object, which contains corresponding information of activity, such as activity’s state, priority, utility value. The ready activity instance list and pre-running activity instance list store its reference.

“Activity agent” is a continuously-running thread, which obtains activity instance from the pre-running activity instance list, simulates the activity, and sends the resulting data to the report module, and meanwhile sends a message to the process agent to inform when the simulation ends.

“Sub-Process agent”, who shares the same ready activity instance list and pre-running activity instance list with the

process agent, is the agent addressing the sub-process. If the type of activity instance is sub-Process (XPDL divides activity into three types: plain activity/Implementation, Route, and sub-Process/compound activity), instance will send to this agent. The inner structure of the sub-Process agent is similar to the process agent.

The resource agent and organization agent act as infrastructure models in the enterprise model. The two agents communicate with the process agent and sub-process agent, and monitor the ready activity instance list and the pre-running activity instance List. The organization agent understands all the organizational details and is responsible for the calculation of the activity instance’s utility.

The Event Generator is used to create the event message, which is sent to the process agent to activate simulation. More than one unrelated process could be defined in one process model, and different messages will activate different process.

The Random Number Generator is used in the calculation of the working time of the activity instance. Different time distribution (e.g. Beta, Standard) depends on a different random number generation algorithm.

The Report Module is a common object collecting statistical information shown in dialog boxes. The Simulation Context is the body of simulation, which is responsible for creating all the agents used in the simulation.

In the simulation framework, the states of the agent include: Initial, ready, running, suspended, complete.

The meanings of these states are explained as follows:

- *Initial* state is the state after the activity is instantiated. As we mentioned before, activity instance is a plain object containing simulation information of the activity.
- *Ready* state is the state that all the former activities of the current instance are completed.
- *Running* state is the state that an instance is in simulation by an activity agent. A running instance can be switched to ready when it is replaced by a higher-utility instance.
- *Suspended* state means the activity is in pause operation during the simulation procedure.
- *Complete* state means that the activity instance finishes its simulation. The instance object will be disposed of.

Figure 2. State transition in process simulation

Figure 2 illustrates the relationship between them. In the workflow management system, there is an activating state

(work-item is working). While in activating, the activity instance could transfer to “suspended” or “completed”. In simulation, what we are concerned with are the running features, and why the work-item does not work, and so on.

When an activity instance is running, it can be replaced by another instance whose utility is higher. Then the instance’s state will switch to “ready”, and the instance is put into the Ready Activity Instance List. Related information (e.g. running time, left working time) is updated in the instance. The process agent maintains the relationship between the activity agent and the organization agent, as well as the utility value of the running instance.

A running activity instance can be replaced, which is similar to preemptive priority scheduling in the computer’s operation system. When an activity instance (called A) is shifted to re-running Activity instance List, the process agent checks its utility value and performer, and then queries the registered activity agents. If the same performer is found (the two activity instances are performed by the same organization), the instance, which is running by the found activity agent (called B), may be replaced by instance A. The preemptive condition is that A’s utility is larger than B’s. For example, a production activity can be stopped for a newly arrived work order with higher priority (utility). A practical influence parameter may be the activity’s duration, cost, etc., but all these influence parameters are in utility function for different organizations.

In the framework, there is no global clock. Resources and roles are initiated, who maintain their own time. In Figure 3, we explain clock management in the framework.

When an activity instance is ready, that is, all the former activities are complete; *Activity Ready Time* is the latest complete time of these former activities. Before running the Ready activity, the required resources and role/role-set (in the organization) instances have to be free, so that it can be allocated to the activity. The corresponding start time of the resources and organization is the active time of the activity. When an activity is complete, the resources and role instances are released, and this time is the end time of service of the resource agent and organization agent.

Figure 3. Time management in the framework

B. Decision Making Strategies

It’s generally believed that different departments have their own interest in an organization. Sometimes local interest in the inner organization conflicts with overall interest in the organization, as a whole. Therefore the organization/

department can be described as a competitive agent, or a self-interested agent. The Utility function, defined as social rationality, is an appropriate form to represent the decision making strategies of organization.

A self-interested agent neither means that it wants to harm another agent nor means it only cares about things that benefit itself. In simulation, resource competition is inevitable because the bounded resource could not satisfy all the needs. Since a resource always belongs to an organization/department, the allocation of a resource is determined by the interest of the organization. When two or more activities compete for one resource, the organization agent calculates the utility value based on a certain index predefined in activity and k_1 , k_2 defined in the organization. The parameters k_1 , k_2 , which represent the orientation of the organization, are constants determined through the consultation between simulation engineers and customers, and modified in a few simulation procedures.

To support workflow simulation, we define some XML-based elements containing simulation information or facilitating simulation. These elements, such as <TimeDistribution> (containing information on the distribution of activity’s duration, e.g. beta distribution, standard distribution), <Utility> (containing information to facilitate utility calculation, used to support activity execution) can be seamlessly inserted into another XPDL-derived workflow model schema.

In our implemented system, *selfUtility* and *publicUtility* are defined in <ExtendedAttributes> in each <activity> element of the workflow model. <ExtendedAttribute>, which contains few <name, value> pairs, is widely used in XPDL. The utility can be a constant, or a key factor radically affecting the activity, such as the number of a specific resource, and the working time of the activity. For example, in the repertory department, the cubage is bounded, and deliveries should be done in time to contain newly produced products. When the delivery of the day is already scheduled, the repertory department prefers delivery according to the cubage of the delivery products. As a result, the delivery priority or delivery time is the key factor in *publicUtility*, and the cubage becomes the key factor in *selfUtility*. Combined with k_1 , k_2 defined in repertory, the utility of delivery activity can be calculated.

Once there is no conflict between local interest and overall interest, the social rationality of agent degenerates into a singular-parameter utility function. For example, in a sale department, the employee’s salary is directly affected by the sale amount. Receiving more customers is the same preference of local interest and the overall target. Thus, activities can be scheduled based on working time or customer priority.

In our simulation framework, the *organization agent* is responsible for calculating utility. When the two conflicting instances’ utility values are equal, the maximum waiting time first serve or random choice can be used to determine the choice.

The steps dealing with resource competition are as follows:

- (1) When activity instance is put into the Ready Activity Instance List, the resource agent and organization agent are informed.
- (2) The organization agent checks all the ready activity instances in the list and finds out all instances whose requirements in role/organization can be met.
- (3) Finding out conflicting instances. Only for activities that need to be supported by the same organization or resources, conflicts may appear.
- (4) Comparing the utility values of conflicting instances. We need to choose the instance whose value is maximal.
- (5) The process agent tries to allocate the resource agents to the activity instances in step 4.
- (6) Instances whose requirements in resource are put into the Pre-running Activity Instance List.

C. Multi-Agent Communication Mechanism

How to build more flexible and practical collaborative systems is clearly an important research topic. Collaboration technologies have the potential to enhance the effectiveness of teamwork within and between organizations. The roles played by the participants in a collaborative activity are important factors in achieving successful outcomes. The multi-agent coordination mechanism is achieved through communication and information sharing between the agents [11]. The main communication mechanism is the basis of collaboration, with the current means of communication – the blackboard system. The problem solving means of the coordination mechanism is the use of appropriate structural support distributed by the agenda. In the multi-agent workflow simulation system, we adopted the blackboard communication mechanism which is the provision of the public work area for all agents to communicate with each other in the system [12]. The agents can exchange information, data and knowledge with each other by using the public blackboard communication mechanism.

1) Request - Response Mechanism

The following are examples of the implementation of the Requests-Response mechanism on the assumption that an information system has inventory management subsystems, sales management subsystems and financial management subsystems. The financial management subsystems and the sales management subsystems need to access the product data from the inventory management system, and the messages between them are transmitted as shown in Figure 4.

- *Registered Service:* Every Agent sends the registration services message to EISP, registered as the users of the services. (The Registration process does not need to repeat).

Figure 4. Requests-Response and Release-Reservation mechanism

- *Provide Service:* Agent3 through EISP sent providing services registered message to SRC (The message must include the WSDL document description or URL address of the services), as a produce data services provider. (Has become providers do not have to repeat this process)
- *Request Service:* Agent1 and Agent2 send the request message to EISP to access the product data.
- *Forward Service:* After receiving the request message 1 of Agent1, EISP through the enquiry service registration centre designates Agent3 as the respondent, and transfers the request message 1 to it (the same deal with the request message 2).
- *Response Service:* Agent3 deals with the request message 1 (including the data conversion process of the inventory management system), returning a response message 1 to EISP (the same deal with the request at 2).
- *Forward Service:* EISP receives the response message 3 of Agent3, and transmits the message to Agent1 (the same treatment response at 5).
- *Conversion Service:* Agent1 will receive the response message 1 based on the SOAP, and then converts the message into the internal data format of the Sales Management System A (the same treat response at 2).

2) Releases-Reservation Mechanism

Updating events of the data object include data adding, modification and deleting. When the application system updates its data objects, the event messages should be transmitted to EISP. If the DEA wants timely access to data updating, the DEA should reserve the updating events through EISP. The reservation is achieved by sending the reservation message to EISP.

After the events releaser transmits the updating events of the data objects to EISP, EISP will be responsible for forwarding them to every DEA who reserves the updating events of the data objects. Since EISP forwards events messages, without informing the release of the original events, the original releaser doesn't need to be concerned about which applications receive the updating events in the case after the updating events are completed, as well as whether the events have been transmitted to the DEA who are reserving the events.

A message is written by an agent on the blackboard. The other agents can access the blackboard to read the message. When the message concerns the service that some agent wants, the agent should send a message to the writer. Meanwhile, a relationship message should be written to the blackboard to show the service used. An agent may access the blackboard at any time [13]. The communication mechanism is complemented by the service configuration message shown in Figure 5.

After an agent has read a message from the blackboard, the agent core will be activated by the message. The core is the main method or function of the call nested in coordinated inter-related tasks.

```

Struct Service {
    Agent ID Source; //The service name of the source agent;
    Agent IP SourceIP;
    //The service IP address of the source agent;
    Unsigned Servicetype; // Service type;
    CString Servicename; // Service name;
    CString Servicecontent; // Service content;
    CString Servicepriority; // Service priority;
    CString ServiceResource; // Service resource.
    .....
}

```

Figure 5. Communication configuration message

Overall, the strengths of this approach are that the finally constructed methodology is highly attuned to system conditions and the agents in the Workflow system. The challenges are to construct several blackboards, ensuring that linkages are in accordance with the local situation and that the interfaces of any pair of method fragments to be plugged together are compatible. Both of these can be facilitated by the use of software tools, the former with a process construction tool, both of which we have a prototype [14].

IV. PROTOTYPE SYSTEM-WFEMULATOR

To validate our framework, a prototype system, WfEmulator, has been designed and developed, which is written in Java. The workflow model generated by WfModel can be simulated in our prototype application. The architecture of the prototype system is shown in Figure 6.

Figure 6. Architecture of Wfemulator

In the architecture, WfEmulator is mainly composed of four parties: *User Interface*, *Process Center*, *Wfmodel Center*, and *Simulation Engine*. *Process Center* and *Simulation Engine* form the core of the simulation system.

- *User Interface*: the module shows the *Tools Bar* and *Menu Bar* to the users; The *Tools bar* includes many modeling figures, charts, etc. to model a work flow of a system. The *Menu Bar* includes load action, save action, operation action and system setting, etc.
- *Process Center*: the module is the operation module of the system. The *Process Agent* is defined by many current process rules. When a workflow is defined, the *Process Agent* should create the relevant relations for every operation of the workflow. The *Resource Agent* manages all of the resources of the system, such as resource assignment, resource statistics, and resource effective use analysis. *Activity Agent* is created by the super *process agent*. An activity takes charge of one activity from the beginning to the end of the activity.
- *Wfmodel Center*: simulation configuration is the initialization of the simulation job. There are many ex-defined workflow models on different operation processes in various enterprises. Initially, users are either allowed to open an existing workflow model or define a new one and save it in the model database.
- *Simulation Engine*: the module is the control center of the system. Many calculations, schedule rules, and analysis means are defined in the module which drives the whole of the operation of the simulation process. The Simulation Report shows the simulation result and provides advice to the users. The Simulation Monitor cooperates with the operations and communication of the different agents.

The workflow model used in our prototype system is generated by WfModel, which generates a XPDL-compatible model in XML format [15]. Simulation configuration is an XML file containing information on simulation, e.g., halt condition, resource instantiation, activity agent number [16].

V. A CASE STUDY

This section gives a real case regarding how we use the proposed approach to implement a process simulation in a virtual supply chain enterprise. For the integrity of the content, we concentrate on the modeling method and analysis, omitting some of the implementation details.

A. Workflow Modeling

X-C is a company offering car customization services. As a small and medium sized enterprise (SME), X-C outsourced some of its business functions (e.g., production and logistics) since the company wants to concentrate on its core competence, i.e., the ability to design and deliver customized cars. In this project, we have made some modifications to our formerly developed central workflow management system, as the cross-enterprise workflow system.

Using the above means, we analyzed the workflow of the X-C Company. When a customer wants to book an order, he

visits X-C's website and fills in the related information (type, brakes, pedals, etc.), some extra-specifications and the information about the customer. Then a car customization process starts. First, the sales department checks the order and completes possible missing fields. Afterwards, three tasks are performed in parallel: the financial department calculates the price, the stock department checks the stock, and the buyer informs the manufacturer. After all these three steps are completed, the system decides whether this order is feasible or not. If it is feasible, an order confirmation letter is sent to the customer and buyer, and delivery information is generated.

From a macro point of view, we built up the workflow model of X-C Company which is showed in Figure 7. The company's business processes P_1 is the first session of the workflow, which provides the company with required data and relevant information to support customers and market analysis. Through the process P_1 , the company will be able to get the market forecast (P_2), the process customer orders (P_3), and the shipping schedule planning process (P_4). Meanwhile, the P_1 is a guarantee for the implementation process. From activity A_7 , we can see the two situations, one is normal for a successful implementation of the supply. Another one shows a shortage and is unable to complete the requirements of P_1 .

Figure 7. A workflow model case

In Figure 7, the workflow is just the main workflow of X-C Company which is used to complete the main business in the company. Meanwhile, there are still some small workflows that were not drawn-out in the figure on the whole company operation. The workflow model is an abstract process of the actual work processes, which will help to describe and understand the real workflow in the company. Our workflow simulation system is easy for users to discover the problems during process simulation, by analyzing the status of various activities, using different colors to explicitly show the dynamic information.

B. Process Simulation Analysis

Process simulation analysis is very important for a simulation system. The core of the simulation analysis module is a set of calculations which can provide the intelligence for business process simulation [17]. Our simulation system provides the detailed reporting forms on the order, running

time, costs and other statements for each process execution. After a simulation is completed, those reporting forms will be automatically generated to support users to analyze the workflows. Meanwhile, process simulation also provides the simulation results in the mode of graphics to significantly support the manager to optimize the workflow. According to specific requirements, users can investigate the various indicators of the workflow execution results, such as activity running time, process average cycle, costs, and resource utilization. Figure 8 illustrates the various activities execution histogram of the above workflow simulation model.

Figure 8. Activity execution time histogram under dense business

In Figure 8, we can see the detailed information on the activity running time during the implementation process of the workflow. The time spent on the various events is clear and straightforward in the histogram. When we adjusted the probability distribution of the orders' arrival time, we found that the event running time also correspondingly changed.

Figure 9 exhibits the activity execution time histogram adjusting the order arrival probability as sparse distribution.

Figure 9. Activity execution time histogram under sparse business

According to our analysis of the simulation data, we found that enterprise resource utilization also changed with the change of the orders' arrival time at the same time. There is a relation between the changing trend of resource utilization and the change of activity execution time. In this way, a very useful management conclusion can be verified in the case, i.e., the enterprise resource utilization is affected by the denseness or sparseness of the business.

We even gave X-C some corresponding proposals to adjust the strategies of a different workflow enactment and resource allocation to deal with business process scheduling. Through this case, we see that the multi-agent workflow simulation system and the proposed method have helped X-C coordinate its business processes in different situations. Feedback from X-

C has shown that this approach has been very effective in increasing the flexibility of their processes, dynamically distributing their resources, and shortening their process execution cycle.

VI. CONCLUSION AND FUTURE WORK

This paper presents a framework for process simulation based on multi-agent cooperation to support flexible activity scheduling. Social rationality of agents is used to represent the utility function as the decision making strategies of the organization. Physical elements such as cubage and cost can be imported to schedule activity instances, which makes the simulation run in a more flexible and realistic manner. The proposed framework consists of socially rational agents [18]. If an agent has the ability to learn from previous decision making successes and failures, there is the potential that it could converge towards finding the right balance, depending on the situation, between individual and social needs. Hence, a knowledge mechanism can be imported into our simulation framework to accelerate decision making when a similar situation appears again [19].

Furthermore, due to the autonomy of the agents, the number of activity agents may vary and the activities can be distributed to a number of computers, according to the availability of the computational resources. Multiple sub-process simulations can run simultaneously to speed up the complex process simulation. Thus, a distributed simulation environment becomes a potential alternative. JNDI and RMI [20] can be used to help communication among agents in such a distributed simulation system.

REFERENCES

- [1] Workflow Process Definition Interface— XML Process Definition Language, Workflow Management Coalition Document Number WPMC-TC-1025.
- [2] Mohan, C. Workflow management in the internet age, Springer Berlin / Heidelberg, Volume 1475/1998, 2006. 8, pp. 31-35.
- [3] Gruninger M., Menzel C. The Process Specification Language, Principles and Applications, AI Magazine, 2003, pp.24: 63-74.
- [4] Kary F., Timo A.R., Mikko K., Jan H. Agent-based model for managing composite product information, Computers in Industry, 2006, pp. 57: 72–81.
- [5] Mingyu G and Vincent Conitzer, Optimal-in-expectation redistribution mechanisms, Artificial Intelligence, 2010,174(5-6), pp. 363-381.
- [6] Ali Ghorbani, and Evangelos Milios, Computational Intelligence, 2008, pp. 76-81.
- [7] Excelente-Toledo, C.B. and Bourne, R.A. and Jennings, N.R. (2001) Reasoning about commitments and penalties for coordination between autonomous agents. In: Proc. 5th Int Conf on Autonomous Agents (Agents-2001). , Montreal, Canada, pp. 131-138.
- [8] Jennings N.R., Campos J. Toward a social level characterization of socially responsible Agent, Proc. IEEE Software Eng., 1997, 144, pp. 11-25.
- [9] Hogg L., Jennings N.R., Variable Sociability in Agent-Based Decision Making, Proc. of 6th Int. Workshop on Agent Theories Architectures and Languages (ATAL-99), 1999, pp. 305-318.
- [10] Chen X.Y., Shi C.Y. Research on Socially Rationality of agent, Journal of Software, 2001, 12(12), pp. 1825-1829.
- [11] John R., Cathal H., Process modeling for simulation, Computers in Industry, 2006, 57, pp. 437–450.
- [12] Tan W., Fan Y. Dynamic workflow model fragmentation for distributed execution, Computers in Industry, 2007, 58, pp. 381-391.
- [13] Kendall E., et al. A Framework for Agent Systems, Implementing Application Frameworks, Object-Oriented Frameworks at Work, M. Fayad et al. (eds.). John Wiley & Sons, (1999)
- [14] Cossentino M., Potts M. A CASE tool supported methodology for the design of multi-agent systems, The 2002 International Conference on Software Engineering, Research and Practice (SERP'02), Las Vegas, USA, 2002, 7.
- [15] Iglesias C., et al. A Survey of Agent-Oriented Methodologies, Proceedings of the ATAL 98, Paris, France, July 1998, pp. 317-330.
- [16] W.M.P. Van der Aalst. Loosely coupled inter-organizational workflows: modeling and analyzing workflows crossing organizational boundaries, Information & Management 2000, 37 (2): 67-75.
- [17] Kang K., Cohen S., Hess J., Novak W., Peterson A. Feature-Oriented Domain Analysis (FODA) Feasibility Study, Technical Report CMU/SEI-90-TR-21, Software Engineering Institute, Carnegie Mellon University, (1990).
- [18] Tan W.A., Shen W., Zhao J. A methodology for dynamic enterprise process performance evaluation, Computers in Industry, 2007, 58(5), pp. 474-485.
- [19] Bellifemine F., Poggi A., Rimassi G. JADE: A FIPA-Compliant Agent Framework, Proc. Practical Applications of Intelligent Agents and Multi-Agents, 1999, 4 pp. 97-108.
- [20] Garcia A., Lucena C., Cowan D. Agents in Object-Oriented Software Engineering, Software: Practice and Experience, 2004, 5, pp. 1-33.