[1]SIMULATION-BASED OPTIMISATION OF A REAL WORLD MANUFACTURING PROBLEM.

Ivana Budinská, Zoltán Balogh and Ján Zelenka

Institute of Informatics Slovak Academy of Sciences,
Dúbravská cesta 9, 845 07 Bratislava, Slovak Republic
Tel.: +421 2 59411242 Fax: +421 2 54771004
e-mail: budinska@savba.sk

Abstract: Many real world manufacturing problems are too complex to be modeled analytically. The complexity of enterprise systems, the increasing needs for advanced collaboration between various systems within one institution or among many collaborating parties, and the velocity of organizational, policy, structural and market changes present nowadays challenges for the research community to develop more flexible and reliable technologies. Simulation based optimization is a very actual topic in discrete-event simulation. The paper deals with the problem of knowledge modeling within industrial enterprises in order to simulate bussines processes and support inter enterprise collaboration.

Keywords: simulation-based optimization, discrete-event simulation, knowledge representation

1 INTRODUCTION

The challenges for business today are growing because of increased complexity of globalization focused on the end customer. Shorter product life cycles, the reduction in time-to-market with a growing network of partners and operating of new competitive enterprises are contributing to the rapid changes in business. The need for interoperability, cooperation (information sharing) and collaboration (tasks and information sharing) is prominent in the industrial enterprise environment (Nof, 2005). The main topics from the enterprise system development perspective are focused on relations of process modelling and execution languages with business ontologies and rules, and how they are applyed into technologies (e.g., semantic web) and architectures (e.g., service-oriented architectures) that enable collaboration between heterogeneous enterprise systems.

2 KNOWLEDGE REPRESENATION AND FORMALIZATION

Knowledge representation is the application of logic and ontology to the task of constructing computable models for some application domain. Each of the three basic fields: logic, ontology, and computation; presents a different class of problems for knowledge sharing (Budinska, 2007):

- *Logic.* Different implementations support different subsets and variations of logic. Sharing information between them can usually be done automatically if the information can be expressed in the common subset. Other kinds of transfers may be possible, but some of the information may be lost or modified.
- *Ontology.* Different systems may use different names for the same kinds of entities; even worse, they may use the same names for different kinds. Sometimes, two entities with

[1] This work is supported by Slovak Research and Development Agency under contract No. VSMP 0168-09.

different definitions are intended to be the same, but the task of proving that they are indeed the same may be difficult or impossible.

• *Computation.* Even when the names and definitions are identical, computational or implementation side effects may cause the same knowledge to behave differently in different systems. In some implementations, the order of entering rules and data may have an effect on the possible inferences and the results of computations. Sometimes, the side effects may cause a simple inference on one system to get hung up in an endless loop on another system.

There are three types of information to be handled within industrial enterprise networks: data – a pure record of basic data about production processes (e.g. number of workstations, lot sizes, etc.); information – data connected with context of production process (e.g., discrete production process with a number of workstation), and knowledge – interpretation of data (e.g., for discrete processes modeling use Petri Nets). The problem is how to organize data, information and knowledge that could be easily accessed, queried and shared. Creating knowledge infrastructure through a common language simplifies the conventional information management process in many ways (Alexakos, 2005):

- eliminates duplicated data entry
- increases the sophistication of data consistency checking
- accumulates and uses intellectual assets
- ensures hardware and software independence
- provides solution for the evolution and obsolescence of information technology (IT)
- provides knowledge management
- supports supply chain management
- supports concurrent engineering
- supports optimum application selection
- provides life-cycle support
- dramatically expands the usefulness of the Web and Web services.

Importance of knowledge representation formalization is obvious. Ontology has become a very promising approach to the knowledge modeling and formalization.

Informal ontology has to be represented by one of the formal ontology languages in order to build computer processed ontology that is only usable in knowledge management systems. Usually ontology development methodology has its own tool to support ontology and instances in formal ontology representation language. The brief description of some most used ontology languages and tools is given e.g. in (Noy, 2007), (Missikoff, 2003), (Ushold, 1995).

Ontology presents a shared understanding about a certain specific domain. In comparison to a database, ontology allows to handle not only numerical transactional data. It is suited to model unstructured informal knowledge. However the advent of object oriented databases, improved logics and faster inference is making the distinction between DBs and ontologies more fuzzy, i.e. there is (multiple) inheritance, strong encapsulation, fuzzy set algorithms, meta-data standards, neural networks to train, discover and disambiguate meaning, and increased computing. Ontologies are important because they not only enables processing knowledge and data, but the most important role of ontology is in defining sharing meaning, emergence and discovery of gaps and for improving tacit knowledge transfer. Besides ontology as a knowledge base may contain information specified in a declarative language such as logic or expert-system rules, but it may also include unstructured or un-formalized information expressed in natural language or procedural code.

Recently there do not exist many enterprise ontologies (although there are many resources from which these could be created), and those that exist have not yet practical utilization as ontologies. Among the most developed enterprise ontologies are TOVE's Organisation Ontology, AIAI's Enterprise Ontology, and Cycorp's Cyc® Knowledge Base.

Some projects aim to utilize the ontology for improving inter enterprise collaboration and interoperability, e.g., CEO project or VIPNET. A more detailed introduction to the VIPNET Project is available in (Yoshiaki, 2007)

3 ENTERPISE COLLABORATION AND INTEGRATION VIA WORKFLOW

Workflow represents a coordinated integration of industrial enterprise systems based on a semantic service - oriented architecture. Generally workflow is represented by sequentially or parallel executed activities that have to be accomplished to reach the final goal. In term of industrial enterprise network it means that the goal is a final product manufacturing and activities are defined by manufacturing of sub products that are necessary to accomplish the final goal (to produce the final product). In addition, each sub-product represents a sub-workflow and can be treated as a separate workflow (model and execution).

A multi-agent system approach can be used to model and control multi enterprise networks, where each enterprise can be modeled via an agent. Agents in the system can negotiate, interact and collaborate to perform common or concurrent tasks to achieve common or concurrent goals considering limited resources.

Coordination problems may arise from a rapidly changing situation that has being experienced by the agents, representing enterprises, under the manager's control. Besides the difficulty of coordinated behavior with incomplete information and dynamically changing situations, each enterprise must also confront uncertainty in the outcomes of its actions. Even if the global problem is known and the situation is stable, enterprises cannot just create a shared plan and execute it to completion.

Coordinated behavior can be modeled by many types of coordination mechanisms. One class of mechanism deals with the temporal location of actions, and tries to reduce the uncertainty brought by incomplete information. Examples of this class of coordination mechanisms are things like schedules, plans, timelines, appointments, and commitments. A second class of mechanism deals with reducing uncertainty in the dynamics of the environment (especially other enterprises that are represented by agents, actions, etc.). Examples of this class of coordination mechanisms are things like laws, rules, and norms. The third class of mechanisms lies somewhere between providing temporal location information and reducing dynamic environmental uncertainty. Examples of this class of mechanisms are formal organizational structures and organizational roles within entire enterprise.

A particular coordination problem instance can be described as follows: a definition of task – the final product, a definition of an abstract workflow to accomplishing the tasks – all sub-products and sub-processes have to be known, constraints of the sub-processes, relationships between enterprises that form supply chain, a set of enterprises and their capabilities to produce specific sub-products (which enterprise can produce what, how many and in what time), and a set of performance criteria.

In case of using federation architecture for collaboration, a federation contract defines the collaboration processes and roles between enterprises, and enterprise services. The B2B middleware can take responsibilities of the right enterprise and service discovery, contract management and monitoring the system behavior.

Using a collaborative business network models as a source of requirements for industrial enterprise applications requires identification of some commonly accepted properties, policy, and behavior alternatives (Kutvonen, 2004). This can be defined within ontology containing types and model repositories of collaboration infrastructure. It is not required to build a common ontology for all services. Ontologies are developed for particular domain or industry.

4 AN INDUSTRIAL ENTERPRISE WORKFLOW

Let us consider a network of enterprises that are capable to produce a final product P. The network consists of M enterprises. Consider that final product requires N sub-products Pi, and suppose that each sub-product can be produced by at least one enterprise, i=1,…M. In case of no sub-product can be produced by any of enterprise within the network, the goal cannot be reached. Creating a workflow for producing the final product represents the task of creating a supply chain for that product. To create the workflow, the following meta-information services are needed:

1. identification of the network structure;
2. constructing a plan for producing the final product including time constraints and all resources requirements (resources are manufacturing tools, workstations, human resources, and material and sub-products);
3. discovery of potential cooperating partners;
4. static verification of interoperability among partners;
5. contract management.

When an enterprise has to be involved in the network, the following information and knowledge has to be shared with other enterprises:

1. production capabilities and capacity – what products, in what series can produce;

2. time constraints and capabilities – when production of required products can start; and when the products can be delivered to the factory for the further processing (it is good when transport time is included in the delivering time, if not, the transport should be modeled as an independent service, as it is often done).

The other information that is important for workflow constructing process is a price of requested products or services. As enterprises within the network are concurrent, this information is not freely available and it is provided on demand. Note, that the price should be negotiable.

According to previous consideration, the decision process for constructing workflow is as follows:

1. According to the schedule, find all producers to supply the production of the final product (it is clear, that each sub-product by itself can create its own workflow for producing the sub-product that is for the enterprise the final product)

2. For each required sub-product evaluate the possible producers as to the capabilities and capacities. This can reduce the number of possible suppliers. The capacity and time constraints are hard conditions and must not be violated.

3. The negotiation process about prices can start with all possible suppliers. A known algorithm to find the best price can be integrated into the decision support system.

A final workflow creation can be characterized by Bellman’s optimality principle as follows:

“A final workflow is optimal, when all sub-workflow are optimal.”

In the content of this principle the following assumptions are considered:

- it is required that production of sub-products required for the final product were optimal;
- an optimal workflow ensures the optimal cost – the price can be reduced.

The system representing the industrial enterprise network can be open or closed. In the open system the number of possible producer can vary. It is better for finding new producers and to get better solution. On the other hand, the open system doesn't ensure the effectiveness, because a case when no suppliers for required sub products can occur.

Mathematical formulation of the problem:

A complex task represents a process of manufacturing of a final product. It consists of some hierarchically dependent sub-tasks that are represented by a number of sub-processes. Each sub-process can be modelled independently.

A process can be described as a set of sub-processes and activities as follows

$$P = \{P_i, A_j\}$$

$$P_i = (A_{ki})$$

There is a different process instance for each task, each final product, sub-product, service, etc.

Generally each sub-process and activity, associated with a workflow has attributes defined as follows:

$$P_i\{producer, DeliveryTime, Status\}$$

$$producer = \{enterprise_1, enteprise_2, ..., enterprise_n\},$$

defines a set of enterprises, which may produce the required sub-product

$$DeliveryTime = \{date, hour, ...\},$$

defines time, when the product is ready and delivered for further manufacturing (in order to simplify the mathematical description, *DeliveryTime* consists of a *CompletionTime*, when the product is completed and the time of transport).

DeliveryTime of the final product has to be minimal. *DeliveryTime* of the final product is computed as a sum of the maximal *DeliveryTime* in each subprocess and a sum of idle times in the final product manufacturing.

$$Status = \{working, iddle, down\},$$

reports about actual status of the particular enterprise. In order to make the final workflow, the information about when the enterprise is able to start working on the required sub-products is also needed. Each state has attributes defined (time information –estimated waiting time before execution, real waiting time, estimated completion time, real time of execution, estimated time of completion, real time of completion).

In order to control a flow of work within a process instance, conditions for conditional activities are defined. A process of producing a new product consists of several steps from which completion of several sub-products in one time is crucial for demonstrating conditional steps in the workflow.

Activity *Assembly* has a variable *sub-product-available.*

Trigger event for evaluation the variable is *availability of all required sub products..*

Conditions are:

If for $i = 1, \dots n$ (number of all required sub-products) *sub-product $_i$-available* = YES, then execute activity *Assembly.*

If *sub-product$_i$-available* = NO, then wait for the missing sub-product. New activity (*Assembly*) can start, when all sub-products for this activity are ready.

In order to make the final product produced in the minimal time, the workflow is constructed considering the time optimization criteria.

Designing a coordination mechanism means constructing the coordinated plan of enterprises within the network that all enterprise do the correct things in the right time.

This is a difficult proposition in many cases. The agents may have an incomplete view of the problem, and thus not realize which activities are redundant or that certain information would be helpful. The agents might be able to communicate in order to build a global view of the situation (if the data requirements are not too high), but the current situation in the episode can be changing dynamically. Thus even in traditional human hierarchical organizations where someone fulfills a management role, and has a `complete' view of the problem at hand, coordination problems may arise from a rapidly changing situation being experienced by the agents under the manager's control. Finally, besides the difficulty of coordinated behavior with incomplete views and dynamically changing situations, agents must also confront uncertainty in the outcomes of their actions. Even if the global problem is know by all and the situation is stable, agents cannot just create a shared plan and execute it to completion.

5 CONCLUSION

The paper tries to out some ideas about how to solve problems related to inter-enterprise collaboration and coordination. A problem of constructing workflow within industrial enterprises network is described and a solution is proposed.

REFERENCES

ALEXAKOS, C.; KALOGERAS, A.P.; LIKOTHANASSIS, S.; GIALELIS, J.; KOUBIAS, S. (2005) Emerging Technologies and Factory Automation, 2005. ETFA 2005. 10th IEEE Conference, Volume 2, Issue , 19-22 Sept. 2005 Page(s): 7 pp. - Digital Object Identifier 10.1109/ETFA.2005.1612690

BUDINSKÁ, I. - FRANKOVIČ, B. - ORAVEC, V. (2007): Collaboration in industrial network enterprises. In INES 2007 : 11th International Conference on Intelligent Engineering Systems. - IEEE ICS, 2007, p. 173-177. ISBN 1-4244-1148-3.

KUTVONEN L. (2004) Relating MDA and inter-enterprise collaboration management. www.cs.kent.ac.uk/projects/kmf/mdaworkshop/submissions/Kutvonen.pdf (Available in January 2010).

MISSIKOFF M., TAGLINO F. (2003) SymOntoX: A Web –Ontology Tools for eBusiness Domains. In the proc. of the Fourth International Conference on Web Information Systems Engineering (WISE'03), p. 343

NOF, SY; MOREL, G; MONOSTORI, L; MOLINA, A; FILIP, F (2005) From plant and logistics control to multi-enterprise collaboration, Preprints of the 16th IFAC world congress. Prague, 2005. pp 1-14

NOY, N.F.: (2007) Guidelines to ontology development www.ksl.stanford.edu/people/dlm/papers/ontology-tutorial-noy-mcguinness.pdf (Available in April 2007)

NOY, N.F., HAFNER C.D. (2006) The state of the art in ontology design, http://www.aaai.org/Library/Magazine/Vol18/18-03/Papers/AIMag18-03-006.pdf (Available in April 2007)

USHOLD M., KING. M., MORALEE S., ZORGIOS Y. (1995) The Enterprise Ontology, 1995, available at http://www.aiai.ed.ac.uk/project/enterprise/enterprise/ontology.html, (Available in April 2007).

YOSHIAKI Oomasa at al.: (2007) VIPNET – Virtual Production Enterprise Network – summary report. http://www.boroprogram.dsl.pipex.com/ladsebreports/ladseb_t_r_07-02.pdf (Available in April 2007).