



14<sup>th</sup> European Conference on Artificial Intelligence, Berlin

## **ECAI Workshop Notes**

# **Agent Technologies and Their Application Scenarios in Logistics**

*Editors*

I. J. Timm, P. Knirsch, H.-J. Müller, M. Petsch, N. Abchiche,  
P. Davidsson, Y. Demazeau, F. J. Garijo, O. Herzog, St. Kirn, C. Petrie, C. Tessier



## Preface

In recent years the interest in agent technology and other agent-related topics has risen enormously. This is mainly a result of the spreading of both - global network agents that can act independently, and the expectation that very complex tasks can be solved by agents concurrently. Research groups around the world spend considerable amounts of time and effort to explore and develop new agent technologies which can be applied to numerous problems. Artificial intelligence approaches seem to be appropriate for the development of intelligent agent systems.

For long-term success of intelligent agents and multiagent systems it is absolutely crucial to transfer research results into commercial applications. Logistics is a promising field of application because of its inherent complexity. This cannot be overcome by conventional approaches, especially information and product logistics.

Two main topics can be identified. First, AI researchers create intelligent agents on a theoretical level in order to develop, refine, and evaluate various concepts, methods, and techniques. Second, multiagent systems are used for modeling, managing, controlling, and simulating information flows within single companies or networks of companies, e.g. logistic networks on a more application-oriented level.

The application potential of intelligent agents is almost unlimited and often leads to the use of "intelligent objects" in software engineering tasks. As yet, the engineering aspects of this technology have neither been realized nor satisfyingly considered. Therefore, only a small number of concepts, methodologies, and large-scale scenarios for adequate applications of multiagent systems in the industry are available to-date.

The workshop tries to bridge this gap in bringing together theory and practice through a hopefully productive dialogue of researchers from the different disciplines.

The following aspects will be discussed:

- Theoretical and methodological foundations of multiagent systems
- Identification, modeling, simulation, and formal representation of real-world business scenarios in logistics concerning either manufacturing or health care
- Further improvements of agent technology in the context of well-defined problems of the logistical domain
- Business- and market-related impacts of agent technologies
- Mutual influences of evolving business standards, e.g., Jini and XML, and their fruitful merging with existing agent technology standards.

Eleven papers have been accepted for presentation. The workshop has been subdivided into four sessions dealing with *Agent Technologies in Logistics*, *Transportation Logistics*, *Negotiation Models*, and *Application and Perspectives*.

The workshop has been organized by a joint committee consisting of Nadia Abchiche, Paul Davidsson, Yves Demazeau, Francisco J. Garijo, Otthein Herzog, Stefan Kirn, Peter Knirsch, Heinz-Jürgen Müller, Charles Petrie, Mathias Petsch, Catherine Tessier, and Ingo J. Timm. The members of the committee were referees for the papers as well. I would like to thank all authors and committee members for their helpful contributions.

Ingo J. Timm on behalf of the Workshop Organization Committee  
(Bremen, June 14<sup>th</sup>, 2000)

# Contents

<b>SCHEDULE</b> .....	<b>7</b>
<b>AGENT TECHNOLOGIES IN LOGISTICS</b> .....	<b>9</b>
<i>J. HENNOCH AND H. ULRICH: AGENT-BASED MANAGEMENT SYSTEMS IN LOGISTICS</i>	11
<i>H. BRECKLE: A MULTI-AGENT FRAMEWORK FOR PLANNING AND MANAGING MOBILITY SERVICES</i>	17
<i>O. HOLLMANN: MARKET-BASED CONFIGURATION OF COMPLEX PRODUCTS</i>	21
<b>AGENT TECHNOLOGIES IN TRANSPORTATION LOGISTICS</b> .....	<b>27</b>
<i>A. GERBER, G. VIERKE AND I. ZINNIKUS: GENERIC MODELING OF MULTI-AGENT SOLUTIONS FOR TRANSPORTATION DOMAINS</i>	29
<i>M. SCHILLO AND G. VIERKE: MULTIDIMENSIONAL UTILITY VECTORS IN THE TRANSPORTATION DOMAIN</i>	35
<i>T. THUM: OPTIMIZATION OF THE ON-SITE TRANSPORTATION LOGISTICS OF A CHEMICAL FACTORY USING MULTI-AGENT SYSTEMS</i>	45
<b>NEGOTIATION MODELS FOR AGENT TECHNOLOGIES IN LOGISTICS</b> .....	<b>53</b>
<i>A. BRUN AND A. PORTIOLI-STAUDACHER: NEGOTIATION-DRIVEN SUPPLY CHAIN CO-ORDINATION FOR SMALL AND MEDIUM ENTERPRISES</i>	55
<i>F. LOPES, N. MAMEDE, A. Q. NOVAIS, H. COELHO: NEGOTIATION IN A MULTI-AGENT SUPPLY CHAIN SYSTEM</i>	61
<b>APPLICATION AND PERSPECTIVES IN DIFFERENT DOMAINS</b> .....	<b>73</b>
<i>Q. MAIR AND Z. HAAG: AGENT-BASED PROCESS MODEL INTEGRATION IN VIRTUAL SOFTWARE COOPERATIONS</i>	75
<i>H. KNUBLAUCH AND T. ROSE: APPLICATION SCENARIOS OF AGENT-BASED INFORMATION LOGISTICS IN CLINICAL AND ENGINEERING DOMAINS</i>	85
<i>M. BEETZ, J., A. B. CREMERS, B. HELLINGRATH, C. MAZZOCCO: PERSPECTIVES ON PLAN-BASED MULTIAGENT SYSTEMS FOR DISTRIBUTED SUPPLY CHAIN MANAGEMENT IN THE STEEL INDUSTRY</i>	89
<b>WORKSHOP ORGANIZATION COMMITTEE</b> .....	<b>95</b>
<b>AUTHOR INDEX</b> .....	<b>97</b>
<b>KEYWORD INDEX</b> .....	<b>99</b>



## **Schedule**

### ***Agent Technologies in Logistics***

- 9:00 - I. J. Timm: *Introduction and Goals*  
- J. Hennoch, H. Ulrich: *Agent-Based Management Systems in Logistics*  
- H. Breckle: *A Multi-Agent Framework for Planning and Managing Mobility Services*  
- O. Hollmann: *Market-Based Configuration of Complex Products*  
-10:40 *Resume and Discussion*

### ***Agent Technologies in Transportation Logistics***

- 11:00 - A. Gerber et al.: *Generic Modeling of Multi-Agent Solutions for Transportation Domains*  
- M. Schillo, G. Vierke: *Multidimensional Utility Vectors in the Transportation Domain*  
- T. Thum: *Optimization of the On-Site Transportation Logistics of a Chemical Factory using Multi-Agent Systems*  
-12:50 *Resume and Discussion*

### ***Negotiation Models for Agent Technologies in Logistics***

- 13:50 - A. Brun, A. Portioli-Staudacher: *Negotiation-Driven Supply Chain Co-Ordination for Small and Medium Enterprises*  
- F. Lopes et al: *Negotiation in a Multi-Agent Supply Chain System*  
-15:10 *Resume and Discussion*

### ***Application and Perspectives in Different Domains***

- 15:30 - Q. Mair, Z. Haag: *Agent-based Process Model Integration in Virtual Software Cooperations*  
- H. Knublauch, T. Rose: *Application Scenarios of Agent-Based Information Logistics in Clinical and Engineering Domains*  
- M. Beetz et al.: *Perspectives on Plan-based Multiagent Systems for Distributed Supply Chain Management in the Steel Industry*  
-17:10 *Resume and Discussion*



## Agent Technologies in Logistics

<i>J. HENNOCH AND H. ULRICH:</i> AGENT-BASED MANAGEMENT SYSTEMS IN LOGISTICS.....	11
<i>H. BRECKLE:</i> A MULTI-AGENT FRAMEWORK FOR PLANNING AND MANAGING MOBILITY SERVICES.....	17
<i>O. HOLLMANN:</i> MARKET-BASED CONFIGURATION OF COMPLEX PRODUCTS.....	21



# Agent-Based Management Systems in Logistics

Jens Heno<sup>1</sup> and Heinz Ulrich<sup>1</sup>

**Abstract.** In logistics of today's economy we have to deal with distributed systems. To support the demanding management task the multi-agent approach offers promising perspectives. For human organizations a socio-technical system is to be handled. The theory of organizational cybernetics provides basic knowledge about structure-theoretic foundations to reach organizational fitness. For the design of management systems based on a multi-agent concept, we try to adopt as much as possible of the insight valid for human based socio-technical systems to the agent world. We particularly focus on identifying the main characteristics for design of management systems in order to achieve the ability to master a system in a dynamic environment evolving in a short and long-term horizon.

## 1 Introduction

Tasks in production and distribution logistics have to be accomplished within networks on the physical as well as on the informational level in today's global economy. On the physical level systems are locally distributed over large areas, sometimes worldwide, the information system is based on an intra or internet data network anyway. This system layout is a particular challenge for the control task, a distributed system has to be controlled in a dynamic fast changing environment. This requires the ability to react to events and evolutions in the system's environment as well as within the system itself in the short and long time range in a way to guarantee the envisaged short- and long-term goals of the system.

We have to deal with the control of distributed systems in which a larger number of actors (individuals) and agents (software objects) have to take decisions. These decisions have to be co-ordinated and directed towards the global goals of the system. If we consider actors, a socio-technical system has to be handled. For the planning and control task the theory of organizational cybernetics provides basic insight, whereas for automated systems or computer-based decision support tools, the multi-agent approach offers promising perspectives.

System control as we have in mind for logistics systems embedded in a dynamic environment is a management task. Satisfactory solutions can't be obtained by viewing the control task simply as a sequence of single decisions. Therefore we propose to design multi-agent systems in logistics as similar as possible to socio-technical systems with human actors and rely on the available theoretical background of organizational cybernetics.

## 2 Logistics

Our envisaged application field is logistics. We define logistics as the control of material, capital and information flow within an enterprise

<sup>1</sup> Institute for Operations Research, Swiss Federal Institute of Technology, Zürich, Clausiusstrasse 47, 8092 Zürich, Switzerland, email: {henoch, ulrich}@ifor.math.ethz.ch

in view directed towards their global goals.

As to their formal structure logistics systems have a lot of similarities. The following abstraction covers a wide range of practical examples in economy in the field of production, distribution and services.

The purpose of the system is well defined, the goals are specified on a strategic and operational level. The operational task consists in fulfilling a list of orders according to their requests in time and product specification. To process these orders operations according to a specified process flow have to be accomplished consuming determined scarcely available resources. These operations have to be executed in a locally distributed system. As to material, information and capital flow within the system their routing has to follow given directives. The logistics system is embedded in a dynamic environment, where a permanent interaction between the logistics system and the environment takes place.

## 3 Cybernetical Management Theory

Our approach for an appropriate handling of the management task is based on organizational cybernetics in management theory. Organizational cybernetics is a science, which applies principles and models of cybernetics for designing, controlling and developing organizations of all kinds. We refer as theoretical foundation for a cybernetic structure to the combination of the Model of Systemic Control (MSC) developed at the University of St. Gallen [12] and the Viable System Model (VSM) by Stafford Beer [3].

### 3.1 Model of Systemic Control

The developed model suggests that the management task has to be accomplished on different logical management levels, called operational, strategic and normative level. A system has to evolve continuously to meet short- and long term goals. Different control variables have to be considered in order to pursue these different goals. While steering a system by controlling the lead time on the operational level might be sufficient, it is not an adequate orientator for the long time horizon of the system. This leads to the introduction of the three management levels. The levels are tied together through an feed-back control cycle, where control variables and parameters of the higher logical level exert a pre-control function in relation to the lower ones.

### 3.2 Viable System Model

Beer pointed out, that a system is only viable if it has a specific management structure (Invariance Theorem). According to the proposed Viable System Model a set of management tasks is distributed to 5 systems ensure the viability of any social system. The 5 systems can be summarized as follows:

**System V** is the top normative decision level (policy-making) of the whole (logistics) system. The main roles of Policy are to provide clarity about the overall direction, values and purpose of the organizational unit.

**System IV** is the link between the primary activities and its environment. On this level the future developments according the systems capabilities and changing of the environment (customer demands) are planned.

**System III** is the controlling unit of the operational level (System I–III). It has to assure the internal stability and to optimize the allocation of resources.

**System II** co-ordinates and regulates the partly autonomous sub-systems.

**System I** controls and optimize the performance of the partly autonomous sub-systems on the short-term basis.

System I–III forms the operational management level, whereas system IV is the strategic and system V the normative one. Finally the management system comprises three logical management levels (cf. section 3.1).

### 3.3 Impact on Agent Technology

The inherent complexity of current systems leads to systems of embedding autonomous tasks within autonomous tasks, rather than to a system of delegating responsibilities from one level to the next (see[5]). For this reason recursively structured multi-agent systems as part of a socio-technical management system should be build, where autonomous units within autonomous units pursue a specific task.

A framework for establishing such viable control systems is provided by the cybernetical management theory, which defines a generic structure for embedding all activities in the management context. Viable in the context of logistics means the capability of a multi-agent system to adapt to changes of its environment (e.g. new order arrival) in order to meet the requirements of customers in the best possible way.

The prerequisite of a system to maintain viability is self-control. Agents can control processes on the basis of actual values compared to target values. In other words, agents are regulators, taking actions, if the output values differ substantially from the target value. This feedback control cycle can only be applied for variables which are controllable (e.g. lead times). In the case of non-controllable variables (i.e. frequency and mix of new order arrivals) a feedforward mechanisms such as demand forecasting models for anticipating this kind of disturbances have to be developed.

The feedback control cycle should also be applied for controlling the performance of agents by agents on a higher logical level (e.g. co-ordinating agents of System II of the VSM model). Furthermore, through this established control flow an agent can inform an agent on a higher logical level about the intractability of a task with respect to the system goals. In this case the informed agent has to handle the problem (e.g. by adjusting the target values).

## 4 Multi-Agent Systems in Logistics

[6] reviewed various definitions of the term agent. The most general way to describe the notion is derived as stated below:

*Agents are software entities, which are dedicated to a specific purpose and carry out some set of operations in order to accomplish tasks.*

From a logistics management perspective this definition is too simple. Agents taking responsibility in fulfilling the operational goals

as well as agents supporting these tasks in a computational way are needed. These thoughts motivate the introduction of two generic meta types of agents in the logistic domain. *Management agents* pursue goals with respect to their environment and their defined action space, whereas their contractors, the *service agents* solve well specified tasks autonomously. Management agents need various algorithmical problem solving methods, such as scheduling of tasks with respect to certain conditions, so it makes sense to delegate these kind of tasks to computational agents particularly designed for such purposes. With this task distribution, management agents are able to focus on decision related problems.

## 4.1 Management Agents

### 4.1.1 Basic Description

Management agents are the central part of a management system and defined<sup>2</sup> as follows:

*Management agents in logistics are software entities for meeting operational goals on behalf of a human actor or another managerial agent with some degree of independence or autonomy, and in so doing employ some knowledge or representation of the user's goals or desires.*

More explicitly: management agents are goal directed, pro-active, take goal responsibility, make decision, and have a model of their environment. They act autonomously, but the actions are constrained by the provided information and models<sup>3</sup> (e.g. from a supervising agent). A management agent needs, therefore, the following information:

- a goal to pursue
- its skills and behavior
- model of its environment
- its role in the agent community (e.g. supervising agent, its sub-agents, collaborating agents)
- communication and co-operation protocols

Since we are in a logistic domain, where the co-ordination of joint actions play an important role, a management agent must exhibit the following properties:

- act in a collaborating manner
- apply various problem solving strategies
- communicational abilities
- methods for solving conflicts among its sub-agents
- capabilities for goal and model building, which are used by its sub-agents

### 4.1.2 Instances of Management Agents in Logistics

We are introducing 3 types of agents (similar to [13]), which have all in section 4.1.1 mentioned properties.

**Resource Agents** A resource is an abstraction of the production facilities and requisites, such as machines and raw materials. A resource can be either allocated by applying simple rules or managed in a more sophisticated manner by resource agents. The agent has knowledge about methods to allocate and control the production resources.

<sup>2</sup> This definition was adapted from the IBM definition for intelligent agents (see [6]).

<sup>3</sup> [7] outlined the benefits of using models in DAI Systems.

**Product Agent** A product agent holds all process information. It knows the components needed to be able to complete a task. Furthermore it has to assure that a task is correctly and in a sufficient way done.

**Order Agent** A flowing entity in a logistics network can be represented by an order agent. It holds all relevant customer specific data, such as due dates. It keeps track that logistical task is performed within the time-frame.

## 4.2 Service Agents

As mentioned earlier in this paper instances of this agent type mainly work on behalf of managerial agents:

*Service agents are contractors of other agents with special computational capabilities without having any representation of their client's goals.*

Typical tasks for service agents in the domain of logistics are:

- computing schedules
- monitoring the logistic system for special events, e.g. machine breakdown
- computing performance data related to logistics such as machine utilization

Moreover, service agents are free to split tasks among sub-contractors.

## 5 Management Systems with Agents

### 5.1 Architecture

Agents are responsible for the management tasks on the operational level. As software artifacts, they are restricted in their activity range by the capacity they get by the implemented software procedures they are equipped with. Only formally comprehensible activities can be executed within our system. We concentrate therefore on technical aspects of management, nevertheless we try to profit as much as possible from the basic experience of management in human organizations. We will base our approach on the management concepts of organizational cybernetics described in section 3.

For a management system comprising an agent community we have conceived two management levels, an upper and a lower level.

The 5 systems of a Viable System Model (VSM) comprise the necessary functionality of a management system. These 5 systems we allocate to our two management levels in the following way:

System V and IV, the normative and strategic tasks, we assume to be given by an external system user, a formal handling of these task will be hardly ever possible.

To the upper management level, we allocate system V, IV and the inner supervising functions of system III. The functionality consists on the one hand in co-ordinating externally the system with its environment on a long-term basis and on the other in co-ordinating internally the actors of the lower level in their effort to reach their operational goals. In doing this, the effort to fulfil the global system goals should always be the guideline.

To the lower management level, we allocate the systems I–III. It comprises the operational management skill to solve specific tasks in the running process in view of the given operational goals and the reaction to events in the environment on a short-term basis. The link to the upper level is established by system III as already mentioned.

Finally, we propose the following generic structure for the agent community on the lower management level:

**Planning** corresponds to System III of the VSM and represents an interface of the autonomous agent system to the higher recursion level. A planning agent has to assure the internal stability (e.g. by adjusting the work load) and optimize the performance of the system with respect to the given operational goal by actors of the same recursion.

**Scheduling** corresponds to System II of the VSM and co-ordinates the actions of the executional part by applying a co-ordination mechanism<sup>4</sup> like Kanban or heuristics like shifting bottleneck procedure (see [1]).

**Execution** and controls either a machine or represent an autonomous sub-system.

### 5.2 Design parameters

The internal design of a management system in a human environment is disputed fundamentally in the economy. Different organization concepts for a management system are investigated. This discussion is also relevant for mechanical systems up to a certain point. The realization of those concepts implies a specific internal design of the management system. In the following we try to identify the most important design characteristics. An important issue for a performance analysis is the variation of these characteristics in order to find out which characteristics is the most effective for a given practical case. The following design parameters characterize a management system in its most important aspects:

#### 5.2.1 Organizational Structure

In a management system the actors with their task to accomplish and their competence within the decision process are members of an organization, which is structured as to ranking levels and functional connections. The lay-out of this structure determines essentially management activities, a strictly *hierarchical* order tends towards a centralized control concepts whereas a *heterarchical* order favors de-central concepts.

#### 5.2.2 Management Technique

For the way management tasks are handled in human organizations different techniques exist. They differ mainly in the part the top management level takes. If it acts only in a passive supervising function, concepts as *Management by Exception* or *Management by Delegation* are to be considered, whereas if the top management level takes an active directing part *Management by Objectives* is a more adequate concept. For a proceeding according to these concepts, in management theory for human organization exists a detailed literature (e.g. [4, 14]), which offer a lot of insight valid for an agent community as well.

#### 5.2.3 Co-ordination

Like a human organization an agent community needs co-ordination of actions for performing their task effectively. Co-ordination can be realized in different ways. According to their individual operational goals, autonomous units can either co-operate or compete with each other. A co-operative society tries to solve their problems by mutually supporting each other with an open information exchange. A society with a competitive culture is based on competition for scarcely

<sup>4</sup> [2] gives a survey of control algorithms which can be implemented in a multi-agent heterarchy.

resource or orders and only the fittest survive. Problem solving is an individual task, information exchange takes place only on a low level basis.

In logistics, in the case of production or procurement of standard products with frequent repetition and fixed batch sizes, the Kanban technique can be applied (see [11]). The key to Kanban is that once it is installed, a very limited exchange of information is necessary for co-ordination. The mutual provision of information is, as it were, 'materialized' in the Kanbans [5].

### 5.2.4 Control Concept

The traditional controversy in planning and control is the question how to execute these tasks, in a centralized or de-centralized way. In a centralized concept the upper management level is directly involved in the operational process directing actively the activities towards the global goals. In a de-centralized concept, decision competence is associated mainly to the lower level, the upper level is not directly involved in the operational process and has in maximum a function as supervisor.

## 5.3 Application Concepts

We are focussing on Management systems run by a community of agents in the field of logistics. These management systems can be applied in different ways. As agents are part of a software program we have to deal with representations of logistics systems, in which the management system functions in a mechanical way.

For analysis purposes we can distinguish two types of applications:

Either we have a model representation of a socio-technical system on a relatively high level of abstraction. We are interested only in technical aspects of management and neglect to capture the purposeful, intentional and also largely voluntaristic character of individuals and groups, which constitute modern human organizations. In this context agents represent directly human decision-makers and determine the performance of the system. The logistics system to manage has to be linked as simulation model to the developed management system.

Or we have an agent-based management system as decision support tool, which is running besides a human organization. Agents in this tool propose solutions to well-defined decisions or take even decisions autonomously as far as decision competence is delegated to agents. The link of the management system with its logistics system to control can be realized in two ways, either to a representation in a simulation model or directly to the real world system. With a simulation model, off-line analyses "what happens if?" are possible, otherwise linked directly to the real world system, we get real-time support.

## 6 Application

In this section we give a brief description of the utilized test bed and we outline the proposed management concept.

### 6.1 Test bed

We consider a shop-floor as it can be found in the semi-conductor industry. The manufacturing process takes place in several automated work-cells embedded in the shop-floor. A work-cell supports four

steps in the assembly process. On the die-bonder the electrical devices are picked from a wafer and glued on a metallic strip (the leadframe). A typical strip carries up to 15 devices. That semi-finished product goes to an oven where the glue is dried. Afterwards the electrical connections (gold wires) between the device and the leadframe are bonded on the wire-bonder. On the die-bonder the strips are filled into magazines and the finally the leadframes are pressed into a plastic form on the mold. The transport of these magazines within the work-cell is automated and is performed by a robot.

The main characteristic for the four equipment types are:

*Die-Bonder:* needs setup when the product changes, can produce approximately 1500 devices per hour, has a limited output buffer where magazines can wait to be moved to the oven if there is no capacity available.

*Oven:* has a limited number of oven chambers that can hold one magazine simultaneously, has no input or output buffer, the duration each magazine has to stay in the oven is product independent.

*Wire-Bonder:* needs setup when the product changes, can produce up to 300 devices per hour, has a limited input and output buffer. Modern wire-bonders can make up to 10 connections per second.

*Molding:* needs setup when the product changes, can produce up to 500 devices per hour, has a limited input and output buffer.

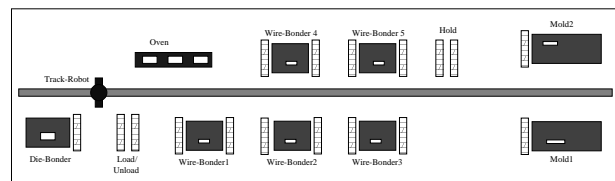


Figure 1. The Automated Work-Cell

Due to the different performances of the equipment a work-cell normally contains one die-bonder, several oven chambers, several wire-bonders and two molds.

## 6.2 Applied Management Concept

The above described logistics system contains two recursion, the shop-floor and the work-cells. A further recursion could be obtained, if machines of the same type but with different specification are regarded as unit. Since the utilized machines of the same type don't differ in their configurations, they are regarded as a simple group represented by an agent.

### 6.2.1 Hierarchical Approach

A shop-floor agent receives a shop order from an actor (e.g. by terminal input) and configures the shop order (lot sizing), determines the due date by lead time scheduling and releases the order to a specific work-cell. Then, a work-cell agent is responsible for the scheduling (sequencing of the lots) with respect to the given due dates. Finally, a machine agent execute the task given from its work-cell agent.

### 6.2.2 Market Mechanism

The order of task to be executed (from order configuration to job processing) are similar to the centralized concepts, but the decision

are taken in a de-centralized way. On the recursion shop-floor the agent has to configure the order and announce the lots with due dates to the work-cells. The planning part of a work-cell formulate a tender depending on:

- the announced lots (quantity, due dates, etc.)
- the arising costs: executional agents can influence the tender by varying their cost models
- demand forecasting model (feedforward): a more lucrative lot could be announced soon
- given operational goals (e.g. quantity of work in process)

The approval of a tender depends, in turn, on its quality (e.g. meeting the due dates), price, offers from other competitors and its own demand forecasting model.

After a bid is accepted, the work-cell and to be more precise the scheduler, continues with the internal planning with respect to the operational goals (e.g. target lead time) and the current situation on the executional level (e.g. amount of work in process). The scheduling agent decides about releasing a lot for its execution and the kind of applied coordination and control concept for the agents on the executional level. As mentioned above, this executional level can be another recursion (e.g. wire-bonder level) and in principal the same procedure can be applied on this recursion.

### 6.2.3 Heterarchical Approach

A pure de-centralized concept means eliminating the work-cells and the emergence of a 'sea of machines'. For each machine an agent is introduced, who competes with other machine agents of the same type for new orders represented by order agents. In principal distributed markets emerge. The difference to the approach outlined in the previous section is, that each machine agent includes the function of planning, scheduling and execution. Moreover, the actions of the agents aren't co-ordinated by a common body.

## 6.3 Evaluation Tool

The simulation platform HIDES (cf. [8]) is particularly designed for the simulation of logistics systems. The most important characteristics of this discrete-event simulator are the reference model for logistics systems providing a small number of simple generic elements as modeling basis and the strict separation of physical and logical level in the modeling process. In order to be able to test various management concepts the simulation kernel was extended by introducing a management level [9]. This management level comprises a management system including an agent community, where agents are in charge of the operational tasks in the logistics system.

## 7 Summary & Outlook

The approach to adapt cybernetical management concepts to multi-agent systems in the case where a close intercation between actors and agents are necessary is quite promising, because

- the agent organization is structured compliant to a socio-technical system, which hopefully improves the understanding of decisions taken by agents for actors<sup>5</sup>

<sup>5</sup> Several authors in [10] give examples in which traditional PPS-Systems fail due to the fact that computer-based planning decisions aren't well understood by the operators.

- the suggested organizing principles are generic for any logistics system and allows an fast adaptation to changing conditions

Further information about the ongoing research project are provided on the Web: <http://www.ifor.math.ethz.ch/research/mgmt>

## REFERENCES

- [1] Joseph Adams, Egon Balas, and Daniel Zwack, 'The shifting bottleneck procedure for job shop scheduling', *Management Science*, **43**, 391–401, (1988).
- [2] Albert D. Baker, 'A survey of factory control algorithms which can be implemented in a multi-agent heterarchy: Dispatching, scheduling and pull', *Journal of Manufacturing Systems*, (1998).
- [3] Stafford Beer, *Brain of the firm*, John Wiley & Sons, Cichester, 1981.
- [4] K. Bleiker, *Das Konzept integriertes Management*, Campus, Frankfurt a.M., 4 edn., 1996.
- [5] Raul Espejo, Werner Schuhmann, Markus Schwaninger, and Ubaldo Bilello, *Organizational Transformation and Learning*, John Wiley & Sons, Cichester, 1996.
- [6] Stan Franklin and Art Graesser, 'Is it an agent, or just a program?: A taxonomy for autonomous agents', in *Proceedings of Third International Workshop on Agent Theories, Architectures, and Languages*. Springer, (1996).
- [7] Les Gasser, 'An Overview of DAI', in *Distributed Artificial Intelligence: Theory and Praxis*, eds., N.M. Avouris and L. Gasser, chapter 1, 9–30, Kluwer Academic Publishers, (1992).
- [8] A. Graber, Heinz Ulrich, D. Schweizer, and A. Zimmermann, 'A Highly Interactive Discrete Event Simulator designed for Systems in Logistics', in *European Simulation Symposium Proceedings*, ed., E. Kerckhoffs A. Verbraeck, (1993).
- [9] Jens Henoch and Heinz Ulrich, 'HIDES: Towards an Agent-Based Simulator', in *Proceedings of Workshop 2000 on Agent-Based Simulation*, ed., Christoph Urban. SCS European Publishing House, (May 2000).
- [10] *Shop Floor Control - A Systems Perspective: From Deterministic Models towards Agile Operations Management*, ed., Eric Scherrer, Springer, 1998.
- [11] Paul Schönsleben, *Integral Logistics Management - Planning and Control of Comprehensive Business Processes*, CRC/ St. Lucie Press, Boca Raton, FL, USA, 2000.
- [12] Markus Schwaninger, *Management-Systeme*, Campus, 1994.
- [13] Hendrik Van Brussel, Jo Wyns, Paul Valckenaers, Luc Bongarts, and Patrick Peeters, 'Reference architecture for holonic manufacturing systems: PROSA', *Computers In Industry, Special Issue on Intelligent Manufacturing Systems*, **37**(3), 225–276, (1998).
- [14] H.P. Wiendahl, *Betriebsorganisation für Ingenieure*, Hanser, München, Wien, 4 edn., 1997.



# A Multi-Agent Framework for Planning and Managing Mobility Services

Hans Breckle<sup>1</sup>

Position Paper

**Keywords:** mobility, services, multi-agents, simulation.

**Abstract.** Mobility services cover a wide field of applications, ranging from pizza delivery to complex inter-modal routing as found in logistics.

In order to understand and optimize the complete operational sequences of these mobility services, it is necessary to build a model which suits all relevant applications, known or foreseeable. This model must encompass the static information, which is typically used in workflow-analysis, as well as the dynamic behavior, which is even more important. In order to achieve these aims, we propose the following steps:

In the first step, we identify individual items which are necessary to model mobility services and which can be combined in arbitrary ways to build new applications. These items take the role of agents, and typically correspond to items in the real world. In order to achieve maximum flexibility, various software based management systems are modelled as agents, too. In the second step a toolbox framework is built to supply the basic environment for the agents to „live“ in. We achieve this with a simulation environment. The third step consists of modelling the identified items with their respective static and dynamic models, including the interaction between agents.

After modelling and simulation of existing mobility services, an evaluation of the framework is done by comparing the simulation results with the real-world situation. Most notably we were able to easily detect application-critical components and bottle-necks of the analyzed mobility services, such as scarcity of resources or inefficiencies in the execution process. Further work can then be focussed on optimization and on modelling and evaluation of new mobility services.

## 1 Introduction

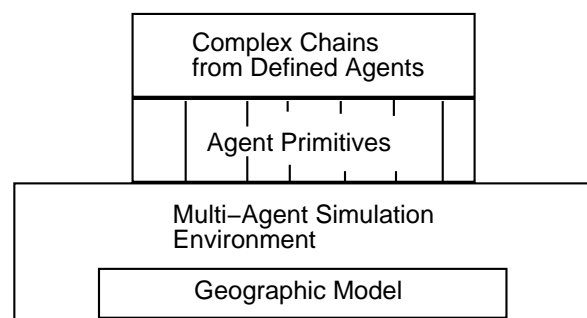
Mobility is a key factor of today's productivity and innovation. In order to maintain or even increase growth rates, it is necessary to perform optimization of all aspects of the production chain. Logistics as a complex application field within mobility offers great potential for optimization, ranging from perfecting transport chains to intelligent traffic management and even autonomous vehicles. A far reach looks beyond mobility itself and is focussed on an optimization which avoids mobility of vehicles, persons or goods where possible and looks for replacements through advanced information technologies. Mobility, whether it is real or virtual, is

vital for today's economics, and it is necessary to get a good understanding of the principles and rules in order to obtain a coherent model.

The set goals of our research work include the development of such a coherent model and the validation of the model. After reaching these goals, the next step is to detect application-critical components and bottle-necks of the applications, such as scarcity of resources or inefficient information flows. The ultimate target is to automatically make improving adjustments to the processes. Along this process, these two aspects are constantly being monitored:

- time constraints: Acceptance of logistic chains usually is directly related to cost vs. time. The exact timing of running processes must be exactly monitored in order to detect delays.
- cost factors: Each component modelled is assigned a cost model to describe all costs, including the third party costs which most often go beyond the scope of our model. This allows an economical evaluation of the processes, which in addition to the time constraints is critical for companies.

Agent systems are the most promising software technology for these applications. The correspondence between objects, vehicles or even people and their interactions in the real world and appropriate agents is obvious. Even management systems which already contain software systems can be modelled as an agent which represents the user employing the software. Due to the diversity and high number of components along with the necessity to have means of communication between the components it is appropriate to use a multi-agent-system.



In order to reach the specified goals, a three-layer model has been developed which is easily extendible and universally

<sup>1</sup> Forschungszentrum Informatik, Dept. Mobility Management and Robotics, Haid-und-Neu-Str. 10-14, D-76131 Karlsruhe, Germany, email: breckle@fzi.de

applicable, even beyond the mobility and logistics theme. The first layer is a universal simulation environment for agents to live in. The second layer consists of basic models of the specific application, such as transport primitives or communication links. The third and highest layer is the user-interaction layer which allows instantiation of primitives and assembling them into complex chains. The next chapters follow the layer structure and detail each layer.

## 2 Multi-Agent Framework

In order to set up a multi-agent framework, it is necessary to first have a definition to work from:

„An agent is a system situated within and a part of an environment that senses that environment and acts on it, over time, ... and so as to effect what it senses in the future“ [3]. For multiple agents, it is necessary to expand this definition to include more than one agent. We can do this by assuming all other agents to be part of the environment when focussing on a particular agent.

A multi-agent system therefore must contain these main characteristics (cmp. [8]):

- an environment, which, from the programming point of view, we will call the framework,
- agents,
- a bi-directional interaction between an agent and the environment (message passing),
- and implicitly, interaction between agents.

We shall detail each aspect in the following.

The framework for the agents to live in is a simulation environment employing an event-driven simulation. Events are time-based, which means that time is the motor of the framework.

The framework is inhabited by agents which communicate solely by passing messages between each other. Reception of a message by an agent will cause an event to occur which in return may send out one or more messages either to the agent itself (a „reminder“), to other agents which must be known to the sender, or cause an action which will show an effect in the framework (such as movement). Message passing is handled by the framework, which is responsible to deliver the messages „in time“ to the recipients. The agent is not only re-active to messages received from others, but can also become active on its own which is modelled by sending messages to itself.

There is also an interface between each agent and the environment, through which the agent is able to know its current position where applicable, as well as to determine the situation (e. g. traffic state, nearby agents) around it. This interface is modelled by meta-messages which are passed between agents and the simulation framework.

In order to achieve a real-world model, the framework utilizes an underlying, complex geographic model, which allows agents to be positioned correctly and move within the world. Movement is achieved by a traffic simulation, which is part of the geographic model. For logistics and mobility applications a well-defined geographic model is vital, because loading/unloading, transport operations, the position of vehicles, goods and personnel as well as the time constraints of travels and movements are major factors of mobility. The two keywords here are **time constraints** and **cost factors** which have been described previously.

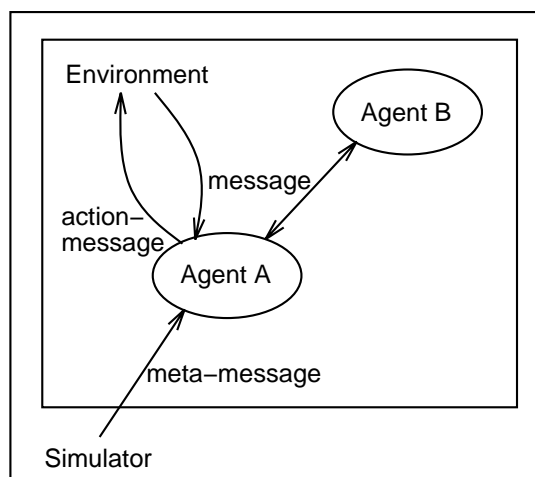
These keywords are the focus of an evaluation system which is part of the framework.

A simulation run will take all agents in the system and will continue to perform different actions by the agents over time until all agents have finished their tasks or until the simulation times out. After a simulation run, results of each agent are recorded. The evaluation system is able to compare different instances of the same agent (see below) or compare agents across simulation runs.

## 3 Messages

Everything that is used to pass information between agents within our framework is called a „message“. Messages are classed into one of four types, these are information transaction, material transaction, meta-messages and agent actions. Monetary transactions are a fifth type with a special behavior which can be derived from both the material transaction and the information transaction.

Each message uses a medium to relay the message over. This medium is responsible for communication restrictions. The different media are described for each type of message below.

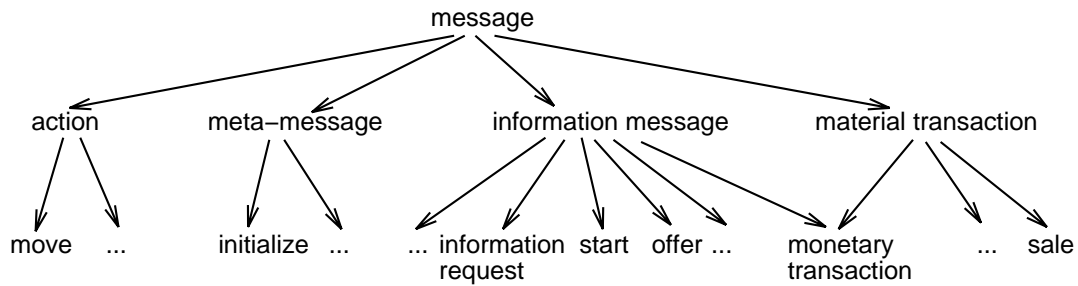


Meta-messages are necessary to ensure that the simulation will run. Two examples of meta-messages are the initialization of each agent at the start of the simulation and the result collection at each agent at the end of the simulation. Due to the meta-character of the messages, there is no medium involved.

Material transactions are used when an object is transferred from one agent to another, or from the simulation environment to an agent or vice versa. It is necessary for the one giving and the one receiving to be at the same geographic position. This is modelled by a „direct vicinity“-medium.

Monetary transactions occur in one of two ways. When cash is involved, the transaction is handled exactly like a material transaction. Credit transfers are possible through most media types: Phone, Fax, Internet, direct vicinity (to the bank), etc.

The media types of this list are also used for all information transactions. This is the most diverse transaction type, used between agents as well as between framework and agent. Speech is one medium used for human agents, which must be considered under the „direct vicinity“ restriction, electronic agents typically prefer



electronic information transmission. It is of course simple to model an electronic agent which is able to speak and hear, but the multifaceted problems which are yet unsolved for a real-world application must remain unconsidered.

Actions by an agent within the environment, manipulation of the environment, or, in the mobility application, movement are the last type of message passing.

Of special interest to us is the employed medium for transactions. While „direct vicinity“ communication does not require additional effort, long distance communication must rely on technical equipment of some sort. This is always connected with costs which are the most important factor in economics. [4] mentions the problem that communication which will cause additional costs requires specific consideration. The main concern is how the cost for communication relates to the average gain from the communication. Our system contains a decision module which will try to determine how high the communication costs are and what the possible gains are. This is a complex task to achieve which requires the agent to either have a detailed knowledge built on experience about the effects of sending a message, or it requires a model which allows the agent to estimate the effects. This problem is heavily influenced by the game theoretical approach as described in [1]. As a first solution, we will use a straightforward low-depth recursive approach for a prognosis of possible outcomes. Obviously, this will allow us to obtain results about the possible cost structures of communication means which we await with anticipation.

#### 4 Agents

Agents have been frequently defined e.g. in [3], [5], [6], [7] with different assignments and goals in mind. Our agents consist of the following parts:

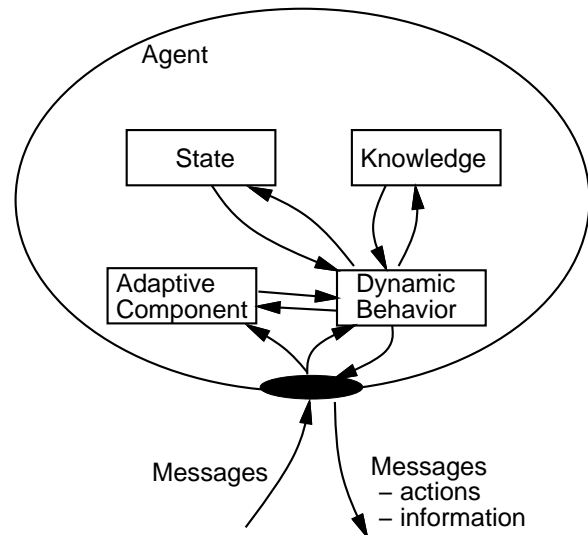
- a static component which describes the state of the agent,
- a knowledge base where the agent can extract information which was previously gathered,
- a dynamic component which is responsible for re-active behavior towards incoming messages,
- an adaptive component which is able to modify the dynamic behavior based on previous success or failure.

The static data is two-fold. It is possible to modify aspects of the agent through user interaction. It is possible to use the same basic agent with different parameter settings (agent instantiation) which will behave differently in situations but will still show the

same basic character traits. User configuration is only possible in advance of a simulation run, which is scenario based.

The knowledge base is still in the developing stage. It is possible to store information and extract this information again later by using search keys (cmp. [2]). Future plans include associative information retrieval and information abstraction.

The dynamic behavior of the agent contains re-active actions to environment input. The agent is able to influence the environment through a set of actions which is predefined by the environment. Currently, more than 20 different actions are defined. Since we consider other agents to be part of the environment, each agent has exactly one interface to it's surroundings. It is then easily possible to define this interface between the agent and its environment by employing a generic model. This model will also take care of dealing with unknown input into an agent. Typically, either a generic re-active behavior is present, or the agent will ignore the message. It is also possible that the agent will ask the sender for an interpretation of the message, and, depending on the sophistication of both the sender and the recipient, new behavior can be learned.



Agents have been modelled to fulfill specific tasks. Even though it is possible for our agents to adopt new behavior, this is currently not our main research interest and shall not be discussed further at this point.

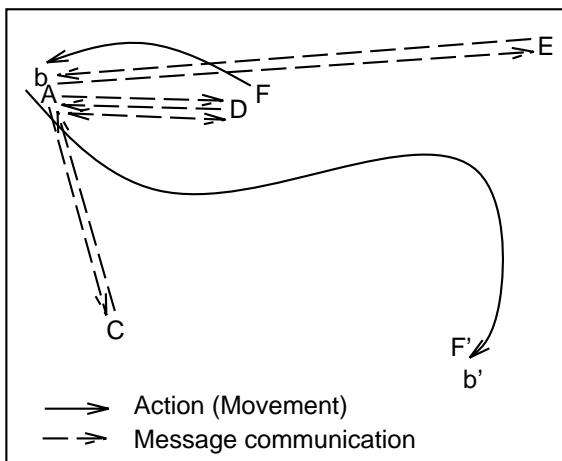
What is more important to us is to gather knowledge of the dynamic behavior of an agent in relation to its environment. The

system behavior follows the laws of the chaos theory, but it is far too complex to be able to apply known mathematical models. This also affirms our basic approach of employing a simulation framework. Some of the concepts of such complex dynamic behavior is described in the next section.

## 5 Complex Logistics Chains

The agents within the framework are able to form complex chains which usually follow the „just in time“ principle. The first approach to complex chains is to manually add connections in order to observe and analyze such complex systems. An automated method for dynamically establishing chains has been designed and will be discussed in detail in future publications.

In order to document an application of the agents' setup, consider the following example:



A customer (agent A) requests some of his goods (material b) to be transported from his company to a customer company. There are several logistics companies (agent C,D,E) competing for the shipment order. Each of these companies utilize different resources (trucks, drivers, etc.) (e.g. truck-driver-entity-agent F of company D). The customer (A) requests an offer from offerers C, D, and E and will typically accept the offer which suits best according to several factors (cost, speed, timing, etc.). Then the exact conditions between the customer and the logistics company are negotiated. Assuming company D got the order, it is then responsible to manage its resources so that the goods are collected at the right time at the right place and are delivered again (F', b'). It may be necessary to reload and use more than one resource for the entire chain, but this is not the main focus. After the successful shipment, payment will occur.

We would like to stress that in addition to the pure transport, we also consider offers, bids, ordering, negotiations before and the payment afterwards to be important parts of the logistics chain, which cannot be ignored. Each of these parts contains potential for economization which needs to be explored. The economization is especially important with regards to the communication costs as described in chapter 3.

Additional focus is put on resource management and cost calculation of every entity in the system. A detailed economic

model which allows the user to see the efficiency of each resource and each component has been integrated.

## 6 Conclusion

Our simulation framework for analysing logistics and mobility service chains represents a complex model of service primitives, a communication network, resources, and an appropriate environment. We focus on completeness of services, with applications in logistics and mobility. Completeness means that the service chain will start with an offer and will end with the payment and will include every step, such as bidding, service negotiation, communication and the transport itself. Our goals are multifaceted. A primary target is to receive a complete model which will be valuable in finding hidden costs and analysing service chain efficiencies. Resource management can be observed and optimised under the current market conditions. Customer demand and provider supply have been modelled as well in order to bring the simulation to life.

By supplying data and cost quotes from real world applications, we are able to receive a very exact model of the real world. This allows us to employ the simulation to prognose the economic effects of technology and resource decisions for a company. It is possible to assess critical components and bottlenecks in the service and logistics chains early and we are able to compare different scenarios and perform optimisation before any misinvestment has been done.

## 7 Acknowledgement

The described research and prototype work is being conducted for an international automobile manufacturer who wishes to remain anonymous. We would like to thank them for the opportunity to proceed with this work, and for the input and data necessary to bring the system to life. Our thanks especially extends to Prof. Dr. P. Levi for his scientific support and assistance.

## 8 References

- [1] A. L. C. Bazzan: An Evolutionary Game-Theoretical Approach for Coordination of Traffic Signal Agents. Dissertation, 1997.
- [2] M. Benerecetti, E. Guinchiglia, L. Serafini, A. Villafiorita: Formal Specification of Beliefs in Multi-Agent Systems, International Journal of Intelligent Systems, Vol. 14, J. Wiley & Sons, 1999
- [3] S. Franklin and A. Graesser: It is an Agent, or just a Program? A Taxonomy for Autonomous Agents. In N. Jennings and M. Wooldridge (eds.): Intelligent Agents, volume III, Springer, 1996.
- [4] A. Huhn: DIPLOMA - Ein System für verteilte Multiagentenplanung in einer Echtzeitumgebung. Dissertation, Karlsruhe, 1991
- [5] F. Klügl: Multiagentensysteme und Simulation, Einführung, Internal Report University Würzburg, 1996.
- [6] Sh. Li: Verteilte Steuerung von kooperativen autonomen mobilen Robotern für Transport-aufgaben. Fortschrittberichte VDI Nr. 599. VDI, 1996.
- [7] T. Lüth: Technische Multi-Agenten-Systeme: verteilte autonome Roboter- und Fertigungssysteme. Hanser, 1998
- [8] A. Uhrmacher: Concepts of Object- and Agent-Oriented Simulation. In ASIM (ed.): ASSIM-96: Multiagentsystems and Simulation, vol. 53 of Mitteilungen aus den Arbeitskreisen, ASIM 1996.

# Market-Based Configuration of Complex Products

Oliver Hollmann<sup>1</sup>

**Abstract.** The agent-based approach *eConfig* is discussed which extends the idea of electronic markets with aspects of knowledge-based configuration. Complex products are composed from various components which are available in an electronic marketplace. The knowledge representation of demands and offers and a flexible communication framework for trading activities is shown. The paper sketches the main idea of *eConfig*, where innovative products can be configured recursive in new logistics networks.

## 1 INTRODUCTION

The configuration of products with a wealth of variants has enormously gained in relevance over the last years. Customers increasingly expect consideration of their individual requirements on a product.

There was a paradigm switch from *mass production* to *mass customization*.

In the automobile industry, for example, a customer can configure a car directly on the internet page of a company [15] [16] and can order it without any personal contact to a storekeeper.

There are some electronic marketplaces and auctions for trading informations, services, and products in the internet [17] [18]. The offered goods are described textually and can be found with manual, keyword- or fulltext-search. Complex products can only be configured manually on an electronic marketplace. There still is a great need for intelligent retrieval, matchmaking and configuration support in electronic marketplaces.

Intelligent software agent technology can help to support time-intensive and complex searching and buying processes in electronic marketplaces.

In this article we discuss the agent-based approach *eConfig* which supports the product configuration in electronic markets. The next section introduces knowledge-based configuration. Section 3 provides a short discussion on electronic marketplaces, requirements, and a short overview of existing approaches. Section 4 shows the framework *eConfig*. The knowledge representation, the communication model and some eCommerce scenarios are discussed. Finally, a short summary and a discussion are given in section 5.

## 2 CONFIGURATION

Solving a configuration task means selecting, parametering, and composing a system from single components to a valid solution according to all requirements and constraints [14] [7]. A configuration task consists of a problem representation and algorithms generating a solution. There are several problem-solving methods for configuration problems (e.g. logic-based, structure-oriented, resource-

<sup>1</sup> TZI, Center for Computing Technologies, University of Bremen, D-28359 Bremen, Germany, email: oho@tzi.de

oriented, associative and function-oriented) which were discussed in [2] and [7].

In this approach we use the domain-independent configuration tool *EngCon* [1] which works with the structure-oriented configuration method. The tool is successfully used for the complex product configuration of electronic drives. *EngCon* is implemented in Java and has an ontology for representing domain knowledge. Domain concepts can be defined with parameters and relations in a concept hierarchy. Modeling descriptions for taxonomic (is-a), partonomic (has-part) and user-defined relations can be used. The action of the configuration engine can be controlled with declarative control knowledge. Requirements and restrictions between configuration objects are represented with constraints. A concept hierarchy of *EngCon* corresponds to a company's product model. The product configuration increasingly influences internal and external business processes and value chains (see figure 1).

The sales distribution of a company uses a configurator for creating customized offers of products with a wealth of variants. The selected components of a solution should be available in stock. Not available components can be manufactured in the production or procured by the buying department of the firm.

Several *logistic constraints* should be considered directly in the configuration process. Such a constraint can postulate for example, that only in stock available components can be integrated into a valid configuration. There will be a configuration conflict if a component is out of stock.

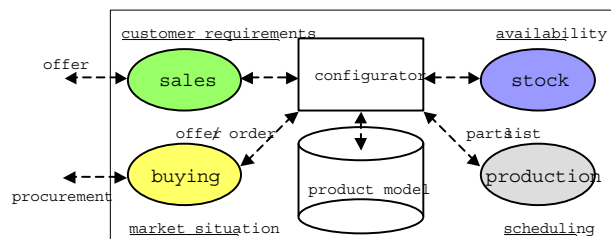


Figure 1. Configurator in a company

The conflict resolution process initiates an automatic procurement task. If the components are not available in the market, the conflict would be resolved by manufacturing the components on the spot.

## 3 ELECTRONIC MARKETPLACES

Electronic commerce is often subdivided into the areas *business-to-consumer (b2c)* and *business-to-business (b2b)* and means the electronic transaction of business processes. In electronic commerce applications information and communication technologies are used for

electronic integration and tooting of value chains [12]. Today the product configuration is very important for the business-to-consumer field. The consumers want to buy individual products which meet their requirements in the best possible way. With an online configuration tool they can configure their solution themselves and order directly via internet.

The *business-to-business* field has got more importance the last time. There are a few industrial auctions and first electronic marketplaces in the internet, where firms can trade industrial products and services.

The acquisition processes of a firm as an element of the supply chain should be supported by electronic procurement (see [12]).

An electronic marketplace with integrated product configuration should possess the following attributes:

- shared ontology for the participants
- scalability: unlimited products and participants
- intelligent (semantic) matchmaking
- distributed architecture
- support of b2c and b2b electronic commerce
- expandability with clear APIs

### Existing Approaches

First we discuss some commercial approaches. There are some online-configuration-systems and a few simple approaches to integrate configuration systems into shopping-systems. The company GEDYS has integrated their rule-based configurator *Net.Select* in INTERSHOP's *enfinity* and IBM's *net.commerce*. The company camos has developed an own online-interface *SECON WEB* for their configuration engine. In this context it will be interesting to observe how the market leader in business software SAP will integrate their configurator *SCE* into their marketplace *mySAP.com*.

The internet auction houses eBay and ricardo.de have developed electronic auction forums for business-to-business eCommerce. Firms can trade different services and goods in *eBayPro* and *ricardoBIZ*.

Siemens try to establish an electronic marketplace *industrialweb* for several old and new industrial goods. All commercial marketplaces mentioned do not support the configuration of complex products.

In the academic field there are some interesting approaches to agent-based eCommerce systems. Maes et al. discuss some agent-based frameworks for online-shopping (PersonaLogic [19], Firefly [20], Jango [21], AuctionBot [22], Kasbah/Market Maker [23], Tete@Tete [24]).

The software agent technology can help to optimize time intensive buying processes like product search and price negotiation.

Maes et al. did an assessment of the frameworks concerning the support of the phases from the "*Consumer Buying Behavior Model*". The *CBB model* was introduced in [10] as a simplification of different models and theories about consumers and the buying process. The frameworks should aid the complex phases *product brokering*, *merchant brokering* and *negotiation*. The system *Tete@Tete* was the only one fitting all phases (see table 1).

	Persona Logic	Firefly	Bargain Finder	Jango	Kasbah	Auction Bot	Tete-a-Tete
product brokering	X	X		X			X
merchant brokering			X	X	X		X
negotiation					X	X	X

Table 1: Agent-based eCommerce systems

The mentioned frameworks are relatively simple *shopping-bot-systems*. Some of them implement flexible negotiation strategies (*Kasbah*, *AuctionBot*, *Tete@Tete*). *PersonaLogic* allows the requirement specifications through constraints and *BargainFinder* makes it

possible to compare different solutions.

All the known commercial and scientific market portals and solutions have no configuration support. They do not offer an ontology or concept hierarchy which is important for intelligent matchmaking and complex configuration. The frameworks and solutions are text-based and the retrieval process is based on full-text search.

## 4 MARKET-BASED CONFIGURATION WITH eConfig

The framework *eConfig* combines the technologies *EngCon* [1] and *eMarket* [9] to an electronic marketplace with integrated product configuration.

In this framework the participants can integrate buying parts directly in their local configurations. Solutions or partial solutions can be offered and demanded on an electronic marketplace. Complex configuration tasks can be subdivided into partial problems which are solved distributed through the market mechanism. It is possible to optimize a configuration manufacturer spanning.

In this section we discuss the knowledge representation of demands and offers, the communication framework, and some electronic commerce scenarios.

### 4.1 Why Agents?

There are different definitions for software agents and their properties. Agents are autonomous, integrated in a world, in which they can act goal-driven, pro-active or reactive, and communicate with other agents and the environment.

*"Software agents are programs to which one can delegate (aspects of) a task. They differ from traditional software in that they are personalized, continuously running and semi-autonomous."* [10]

Agents are especially suitable to support information and process-intensive tasks like those belonging to eCommerce applications. In an eCommerce area are many distributed operating, autonomous actors, which communicate direct or indirect for trading. There are different distributed and heterogeneous acting companies involved with different information technologies, so a monolithic approach drops out.

In *eConfig* we use a flexible agent-based approach for market-based configuration. The criteria for the application of multi-agent-system-technology discussed in [11] are in the context of *eConfig* fulfilled.

- **Natural distribution:** Electronic markets are characterized by the geographical and organizational distribution of the participants in the world wide web. The actors have different roles (e.g. *buyer* and *seller*), various interests and goals.
- **Complex and flexible interaction:** The product and merchant retrieval and the price negotiation require flexible communication mechanisms for the actors.
- **Dynamic world:** In an electronic market there are permanently changing amounts of buyers and sellers trading different products.

### 4.2 Market Actors

In *eConfig* we differentiate three types of agents with diverse properties and skills.

- **Broker-agent:** A central agent in *eConfig*, which receives bids and asks, coordinates the matchmaking, informs potential contract parties and mediates the negotiation process between buying and selling agents. The broker-agent can act as an auctioneer or a mediator. A broker-agent implements several auctions and negotiation protocols. The kind of auction and the negotiation strategy of the broker-agent are determined by the seller of a product.
- **Consumer-agent:** The consumer-agent is a representative for any consumer and asks as a buyer for a product. The consumer-agent implements various tactics and negotiation strategies and is able to negotiate autonomous or semi-autonomous with the broker-agent or any business-agent.
- **Business-agent:** This kind of agent is a representative for a company in *eConfig*. A business-agent can both offer and ask for some products. Business-agents are buyers and sellers.

### 4.3 Knowledge Representation

The products of the *eMarket* are represented in a dynamic expandable concept hierarchy which defines a shared ontology for the actors in *eConfig*. Offers and demands are represented as frames with information about seller/buyer, price and product specifications.

```
(def-content-offer
  :name      "Drive system"
  :seller_id "Trading Inc."
  :price     [10000 15000]
  :product   ((?a :concept Drive
                :parameters ()
                :relations (
                  (has-part ?r ?m)
                  (?r :concept Regulator
                    :parameters (
                      (Frequency 200))
                    :relations (
                      (part-of ?a)))
                  (?m :concept Motor
                    :parameters (
                      (Speed 150)
                      (Momentum 10)
                      (Frequency 200)
                      (Power 1500)
                    :relations (
                      (part-of ?a)))))))
```

The demand for a product is defined by a query consisting of variable-pattern-pairs and some constraints restricting the match-making process.

```
(def-content-demand
  :name      "Drive with motor
             and regulator"
  :buyer_id  "Engineering Ltd."
  :price     10000
  :deadline  "01.08.2000"
  :query     ((?a :concept Drive)
              (?r :concept Regulator
                :relations (
                  (part-of ?a)))
              (?m :concept Motor
                :parameters (
                  (Speed [1000 5000])
                  (Power [100 2000])
                :relations (
                  (part-of ?a))))))
  :constraints ((EQUAL(?r.Frequency,
                      ?m.Frequency))
                (MULTIPLIER(?m.Power,
                          ?m.Speed,
                          ?m.Momentum))))
```

The specifications of offers and demands can be transformed to XML in a future extension of *eConfig*. XML seems to be the standard representation form in electronic commerce.

The XML/EDI group has developed a framework for web-based EDI [25].

### 4.4 Communication

The agents of *eConfig* communicate with flexible speech-acts. The communication can be realized with well-known approaches like KQML [6] or FIPA ACL [5]. There are the following predefined communication primitives in *eConfig* (see figure 2):

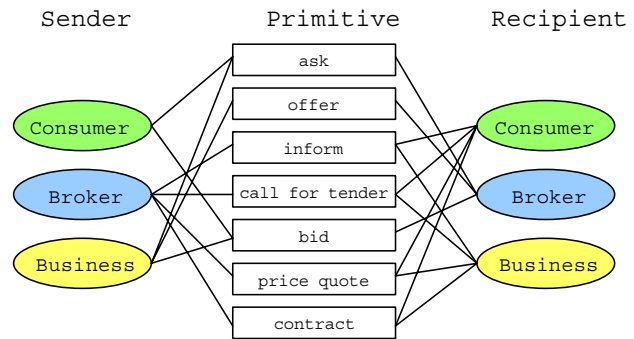


Figure 2. Communication primitives in *eConfig*

- **<ask>** demand of a consumer- or business-agent
- **<offer>** supply of a business-agent
- **<inform>** broker-agent informs potential contractors
- **<call for tender>** broker-agent invites to tender
- **<bid>** bidding of a buyer
- **<price quote>** the current price
- **<contract>** acceptance of a deal

The consumer-agents and the business-agents can use the primitive <inform> to ask which products are actually traded in the electronic market.

The flexible and expandable communication primitives of *eConfig* allow the implementation of diverse communication protocols. The agents can communicate either directly or mediated through the broker-agent.

### 4.5 Electronic Commerce Scenarios

The framework *eConfig* combines product configuration with electronic markets and supports *business-to-consumer* and *business-to-business* electronic commerce (see figure 4 and 3).

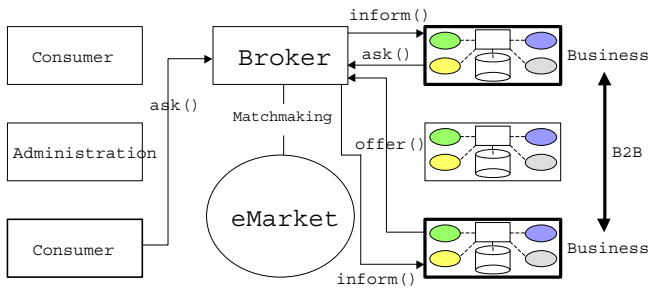


Figure 3. Business-to-business eCommerce

Consumers and firms can participate in trading with representative agents. *Consumer-agents* place demands for products in the market. They send an <ask>-message to the *broker-agent*. The *broker-agent* registers the demands in the *eMarket* and informs the potential contractors if there is a successful *matchmaking*.

*Business-agents* recognize the demands in the market as potential orders. They try to configure a suitable offer. The product configuration is directly integrated into *business-to-consumer* electronic commerce (see figure 4).

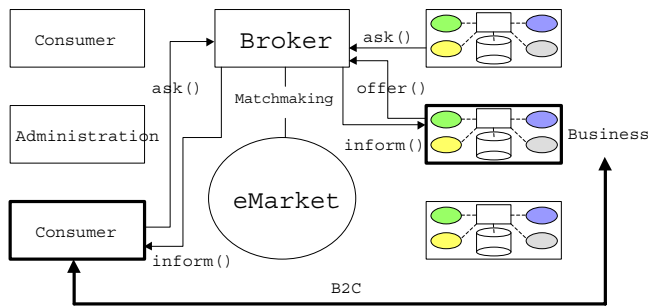


Figure 4. Business-to-consumer eCommerce

During the configuration of a complex product it might be necessary to initiate an automatic acquisition process of any part. The *business-agent* of the company is instructed by the configurator to place a demand for a specific part in the electronic market. With this mechanism products can be configured recursively in *eConfig* and several firms can participate (see figure 3).

## 5 DISCUSSION AND SUMMARY

The framework *eConfig* integrates the product configuration directly in an electronic marketplace. The approach supports both consumer-to-business and business-to-business electronic commerce.

Consumers can define representative agents to configure products in an electronic market. Firms offer configured products via business-agents and ask for parts of configurations in an electronic procurement scenario.

A platform like *eConfig* defines the infrastructure for logistics networks and virtual enterprises. It is possible to configure innovative products on a world-wide electronic marketplace. New logistics networks can be defined through the market-based configuration process.

The flexible communication framework of the actors in *eConfig* allows the implementation of various negotiation protocols and *silent commerce* of software agents with no user interaction.

The approach is scalable, because the number of participating agents and products is not restricted. The complex and time-intensive phases *product brokering*, *merchant brokering* and *negotiation* of the *Consumer Buying Behavior* model (see [10]) are supported by *eConfig*.

An interesting point for future research will be to standardize the representation of the market ontology. The approach should be open for other configuration engines and various enterprise resource planning systems. There are several XML-approaches to represent product data in electronic catalogs [26] or to define interchange formats [25] which can be integrated in *eConfig*. A more interesting point will be to integrate semantic translation and ontological mediation technologies into the matchmaking process. A buyer asks for the product *X*, but there is only a product *X'* which seems to fit the requirements, too.

## REFERENCES

- [1] V. Arlt, A. Guenter, O. Hollmann, L. Hotz, T. Wagner. *Engineering & Configuration - a knowledge-based software tool for complex configuration tasks*, in AAAI-99 Proceedings, Orlando, AAAI-Press 1999.
- [2] A. Brinkop. *Variantenkonstruktion durch Auswertung der Abhängigkeiten zwischen den Konstruktionsbauteilen*, Infix, 1999.
- [3] A. Boehm, H.J. Mueller, J. Rahmer, S. Uellner. *A Discussion of Internet Configuration Systems*, in AAAI-99 Proceedings, Orlando, AAAI-Press 1999.
- [4] A.E. Campbell, S.C. Shapiro. *Algorithms for Ontological Mediation*, Technical Report, Dept. of Computer Science, SUNY Buffalo, Number 98-02, January 23 1998.
- [5] Foundation for Intelligent Physical Agents. *FIPA 97 Specification, Version 2.0, Part 2, Agent Communication Language*, www.fipa.org, 1998.
- [6] T. Finnin, Y. Labrou. *A Proposal for a new KQML Specification*, TR CS-97-03, Computer Science and Electrical Engineering Department, University of Maryland Baltimore County, Baltimore, MD 21250, 1997.
- [7] A. Guenter. *Wissensbasiertes Konfigurieren: Ergebnisse aus dem Projekt PROKON*, Infix, 1995.
- [8] A. Hermanns, M. Sauter. *Electronic Commerce - Grundlagen, Potentiale, Marktteilnehmer und Transaktionen*, in A. Hermanns and M. Sauter (eds.), *Management-Handbuch Electronic Commerce*, Verlag Vahlen, 1999, page 13-30.
- [9] O. Hollmann. *eMarket - Matchmaking komplexer Produkte in elektronischen Märkten*, internal technical report, TZI Bremen, 2000.
- [10] P. Maes, R.H. Guttman, A.G. Moukas. *Agents That Buy and Sell*, Communications of the ACM, Vol. 42 No.3, 1999, page 81-91.
- [11] H.J. Mueller. *Towards Agent Systems Engineering*, Universitaet Bremen, 1996.
- [12] M. Nenninger, M. Gerst. *Wettbewerbsvorteile durch Electronic Procurement*, in A. Hermanns and M. Sauter (eds.), *Management-Handbuch Electronic Commerce*, Verlag Vahlen, 1999, page 283-295.
- [13] P. Rohrbach. *Electronic Commerce im Business-to-Business-Bereich - Herausforderungen, Konzeption und Fallbeispiele*, in A. Hermanns and M. Sauter (Hrsg.), *Management-Handbuch Electronic Commerce*, Verlag Vahlen, 1999, page 271-282.
- [14] D. Sabin, R. Weigel. *Product Configuration Frameworks - A Survey*, in IEEE Intelligent Systems July/August 1998, Seite 42-49.
- [15] <http://www.opel.de/webkauf>
- [16] <http://www.bmw.de/carconfigurator>
- [17] <http://www.industrialweb.com>
- [18] <http://www.ebayPRO.de>
- [19] <http://www.personalogic.com>
- [20] <http://www.firefly.com>
- [21] <http://jango.excite.com>
- [22] <http://auction.eecs.umich.edu>
- [23] <http://maker.media.mit.edu>
- [24] <http://ecommerce.media.mit.edu/tete-a-tete>
- [25] <http://www.xmlmedi.org>
- [26] <http://www.bme.de/bmecat>



## Agent Technologies in Transportation Logistics

<i>A. GERBER, G. VIERKE AND I. ZINNIKUS:</i> <i>GENERIC MODELING OF MULTI-AGENT SOLUTIONS FOR TRANSPORTATION DOMAINS</i> .....	29
<i>M. SCHILLO AND G. VIERKE:</i> <i>MULTIDIMENSIONAL UTILITY VECTORS IN THE TRANSPORTATION DOMAIN</i> .....	35
<i>T. THUM:</i> <i>OPTIMIZATION OF THE ON-SITE TRANSPORTATION LOGISTICS OF A CHEMICAL FACTORY</i> <i>USING MULTI-AGENT SYSTEMS</i> .....	45



# Generic Modeling of Multi-Agent Solutions for Transportation Domains

Andreas Gerber, Gero Vierke, Ingo Zinnikus<sup>1</sup>

**Abstract.** In this paper we describe a proposal for re-using agent specifications in logistics. Based on the application of the multi agent approach, we use object oriented programming concepts in order to build a three-layered object hierarchy. At the top of the hierarchy general structures are specified that are needed in distributed multi agent domains. The intermediate level contains features that are needed for agents used in the transportation domain. On the lowest level domain-specific features are defined explicitly. We describe this hierarchy with respect to the three layered INTERRAP agent architecture.

## 1 INTRODUCTION

Agent technologies provide useful concepts for the design and implementation of large scale distributed software systems (for an overview see Wei99). The agent oriented design enables to model complex distributed domains in a robust, flexible, and natural way. These advantages are bought by an additional effort regarding computational resources like time, space, network load and human resources which is in fact a significant expense factor.

Due to this high effort, re-using code and concepts is an extremely important issue in developing multi agent systems. Many general features like reactivity, planning, communication, and mobility are demanded in most agent systems. Hence, the according code can be re-used easily for different applications. Domain specific features of an agent cannot be generalized; nevertheless, some of these properties can be generalized within a restricted class of application domains. In fact, relatively few characteristics are so specific that the code can never be re-used again.

During the past few years, the authors have been involved in the development of prototypical multi-agent systems for transport domains. The TELETRUCK fleet management system supports the dispatchers of transportation firms with the planning and scheduling of transportation tasks (BFV99). The means of transport like driver, trucks, trailer, etc. are modeled by autonomous agents, which are composed to *holons*, i.e. agents consisting of sub-agents, that represent the vehicles of the company. These vehicle holons are headed by special planning agents that perform the local tour planning of the vehicle. The vehicle holons in turn form the company holon which is headed by a *company agent* that coordinates the overall planning process. Within the PLATFORM system, where an intermodal transportation chain is modeled, a transport is decomposed into road based pre- and end run and rail based main run (FVB99). Like in the TELETRUCK system, the physical means of transport are agentified. However, the additional functionalities for the

intermodal planning had to be integrated into agents that represent the intermodal transportation tasks. TELETRUCK-CC is an extension of the TELETRUCK system that allows several independent shipping companies to cooperatively optimize their fleet schedules (BV99). The additional functionalities were integrated into the company agent and into a new agent that coordinates the cooperation process. Furthermore, the system was distributed over several computers, so that the agents representing a company are hosted by a machine owned by that company.

Agent design methodologies and architectures provide conceptual tools for the design of agent oriented applications but do not support the implementation. Our agent societies were structured according to the holonic multi agent systems methodology (GSV99): agents are composed to holonic agents in order to model the application domain on different levels of abstraction. The single agents are based on the three layered INTERRAP architecture (Mül96): the requirements of reactive behavior, deliberative planning, and social interaction are distributed to the layers.

We consider agent and holon oriented programming as an extension of object oriented programming and we use object oriented concepts in order to implement agent architectures and societies. For our applications which are based on similar concepts but differing demands, we have developed a hierarchical object model in order to rationalize the implementation effort. At the top of the hierarchy very general objects model the skeleton of the agent architecture which allows to design generic INTERRAP agents. The lower levels become more and more specific for transportation agents.

## 2 REALIZATION

Before describing the realization in detail, some remarks about our method might be useful. Because we had worked on a given domain and had already implemented main parts of the system, we had developed some techniques for designing agents in our transport scenario. The claim for a generic model arose when we thought about transferring multi agent techniques to other logistic problems (e.g. modeling intermodal transport domains as mentioned above).

A classical approach would try to develop some kind of interpreter for, say, communication or planning tasks. Indeed, we considered to build a protocol interpreter based on the idea that a protocol can be seen as a finite automaton. But quickly it became clear that this approach would go far beyond our ends, since we restricted ourselves to transport related scenarios.

Instead of inventing interpreters for communication or planning, we decided to start from our implementation, to compare it with other possible applications for our multi agent approach and to abstract common features from these concrete applications. By

<sup>1</sup> German Research Center for Artificial Intelligence (DFKI GmbH), Stuhlsatzenhausweg 3, D-66123 Saarbrücken {agerber,vierke,zinnikus}@dfki.de

using this method, we came to some generic functions, and avoided the effort of developing interpreters.

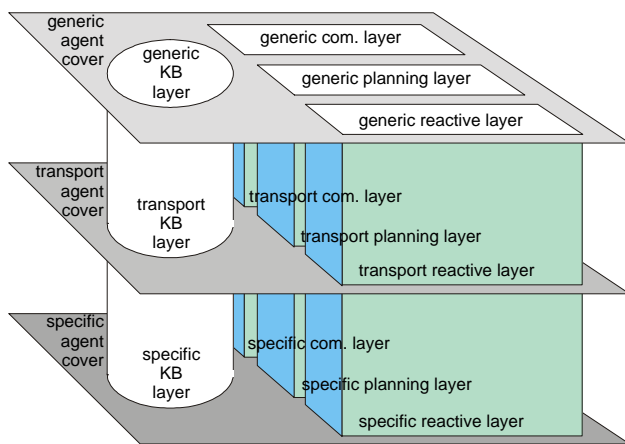


Figure 1. The layers of the architecture on several levels of abstraction

## 2.1 The Agent Cover

As already mentioned, an agent consists of three layers according to the INTERRAP architecture. These layers are enveloped by an *agent cover*. This cover has three main functions, first to protect the agent against unauthorized access, second to represent the agent to the agent society and third to provide methods that initialize the layers, prepare the communication with other agents and administrate the interaction between the layers. In our implementation, the agent cover is represented by an object. These features are common to all agents. As a result, comparison and abstraction lead to some functions which can be used for every type of agent. Differences among the agents concern only the type

of layers they consist of. Hence it is possible to freely choose among some predefined agents. It is even possible to construct hybrid agents with different specific layers, e.g. a holon agent with a motor agent planning layer. When choosing to create a new agent, these functions are spawned subject to the agent in question. If for example a holon agent has to be created, the agent cover contains, beside the basic communication devices, functions that create the layers that in general are used by a holon agent. In fact, we have predefined agents with three levels of abstraction: a generic agent which uses generic layers, a transport agent which uses domain specific but still general layers for transport agents, and some agents with predefined transport specific layers such as 'holon agent', 'motor agent', or 'load-space agent'.

Basic communication devices are connection methods for message exchange in network(s), e.g. via TCP/IP. In earlier versions, we developed agents represented by objects within one system, i.e. within one process; now we have distributed the system by providing several agents with own processes. We aim at keeping these devices independent of the programming language used. Therefore, we had to adopt a standard of message forms used in communication among agents. We chose the KQML standard, and constructed the basic communication devices so that messages of that form can be sent from one agent to another without actually paying attention to details of transmitting data.

## 2.2 The Communication and Protocol Layer

We use KQML (Knowledge query and manipulation language) to model the communication among agents (FF94). KQML is an adoption of speech act theory (Sea69). It divides an utterance into three aspects (locution, illocution and perlocution), whereby *illocution* (also called *performative*) is of special interest. In ordinary speech, this aspect is not explicitly stated. KQML allows to represent several performatives such as assertion, query, reply etc. It is now possible to separate between domain-specific and domain-independent semantics of messages.

Like the agent cover, the layer for social interaction (which we call

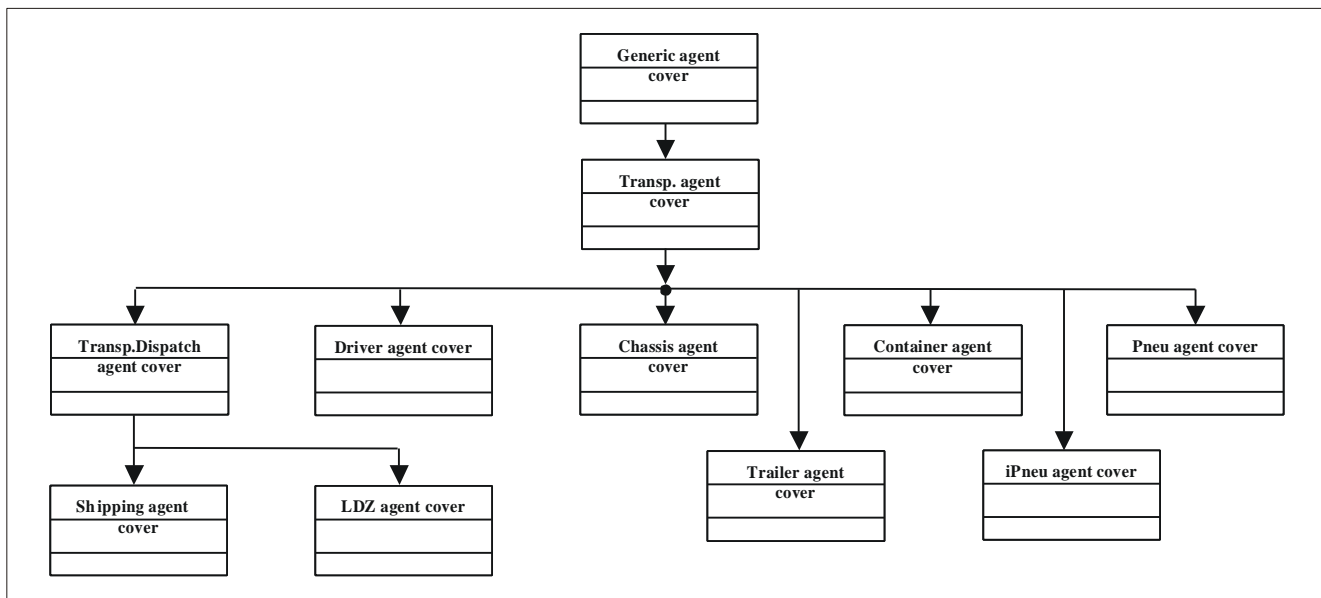


Figure 2. The hierarchy among the agent cover objects

*communication layer*) is modeled hierarchically. The generic communication layer consists of two methods for receiving and sending messages. An agent can act as a sender or a receiver. Furthermore, performatives are represented by methods that compute a message in respect to its content. By now, these methods are mere compositions of *if – then* statements which trigger further computations. Roughly spoken, two methods for performatives *ask* and *tell* are needed. They correspond to the known basic message types or performatives *assertions* and *queries*. Both methods are part of the transport communication layer.

One major problem is dealing with asynchronous communication, i.e. sending and receiving messages in an order which is a priori unknown. To avoid this problem, KQML provides a field *:in-reply-to* which signifies uniquely the message to which a reply is sent. A proper generic implementation has to provide methods that can cope with such messages. We solved the problem by binding all outgoing messages to a dictionary and waiting until each receiver has replied. Additionally, since agents might be unable to reply, we integrated a timeout mechanism.

To perform concrete interaction between agents, special protocols are needed. A protocol is a set of formal rules describing how to perform interaction. In our transport scenario, we use a *contract net protocol* to specify announcing and bidding for transport tasks. Furthermore, since agents can join to build holons, we need a specified protocol for announcing a holon building request. Other kinds of domain-specific interaction require further protocols.

These domain-specific protocols have several features in common. Hence, we can take advantage of these common features by abstracting. A transport specific method for contract net protocols takes a task and announces it to list of agents, then it waits until all agents have replied, chooses the best bid (or chooses an agent which fits best into a holon), and grants or rejects the bids by sending an answer to the contractors.

In fact, the situation is more complicated because we use a nested contract net. A task is announced to all known transport holons. These are represented by PnEUs (planning and execution units), which themselves use contract net to acquire agents as new components and to announce a task to their subagents. Hence making one's bid, when acting as a contractor, requires first starting a new contract net with one or more subagents. An agent might act as a manager and as a contractor at the same time. We solved this problem by creating a special *protocol object* which is associated to an agent and his role in a contract net. This object manages the communication in respect to the role.

This protocol object is part of the transport communication layer. The methods which contain agent specific methods, e.g. methods for a holon agent, are located in the lowest communication layer, which distinguishes domain specific agents from each other. They complete the hierarchical structure of the communication layer.

## 2.3 The Local Planning Layer

Planning is an essential task which is highly domain specific. So it is difficult to get an useful organisation of the abstraction layers. Therefore, it is important to make clear detachments between the abstraction levels. There are four levels of abstraction we distinguished within the local planning layer in our transportation domain.

On the top of the abstraction hierarchy is a generic planning layer which contains only basic structures and the interfaces to the other layers. It is able to include some kind of more domain specific

planning modules and it should coordinate the planning tasks between the inserted modules.

The next level in the hierarchy includes a more domain specific planning layer. In our transportation domain it is a level which does not contain any planning algorithms. It includes all significant operations to deal with plans – regardless of the structure of the plansteps. There are methods for deleting, inserting and searching of certain plansteps. Furthermore, it is able to perform some easy computation over the plan, e.g. counting the plansteps.

The following level in the abstraction hierarchy is the domain specific planning layer. It contains information about the object the agent represents in the real world. The planning layer of a shipping company agent has to be able to divide an order into several suborders, if it is necessary to get a solution of the problem of allocating the orders to the road trains of a shipping company. The most important function of this level is to do all computations which need some extra information about the real world, e.g. the resources of represented objects. Therefore it could be necessary to build different planning components which consider the functionality of the real world objects like the driving time component of a driver. A container of a road train does not have to represent the used driving time in the way it is necessary for the driver-agent. It does not make sense to take care about the weight and capacity of the loaded units by the driver-agent. This is the job of the container- or road train agent.

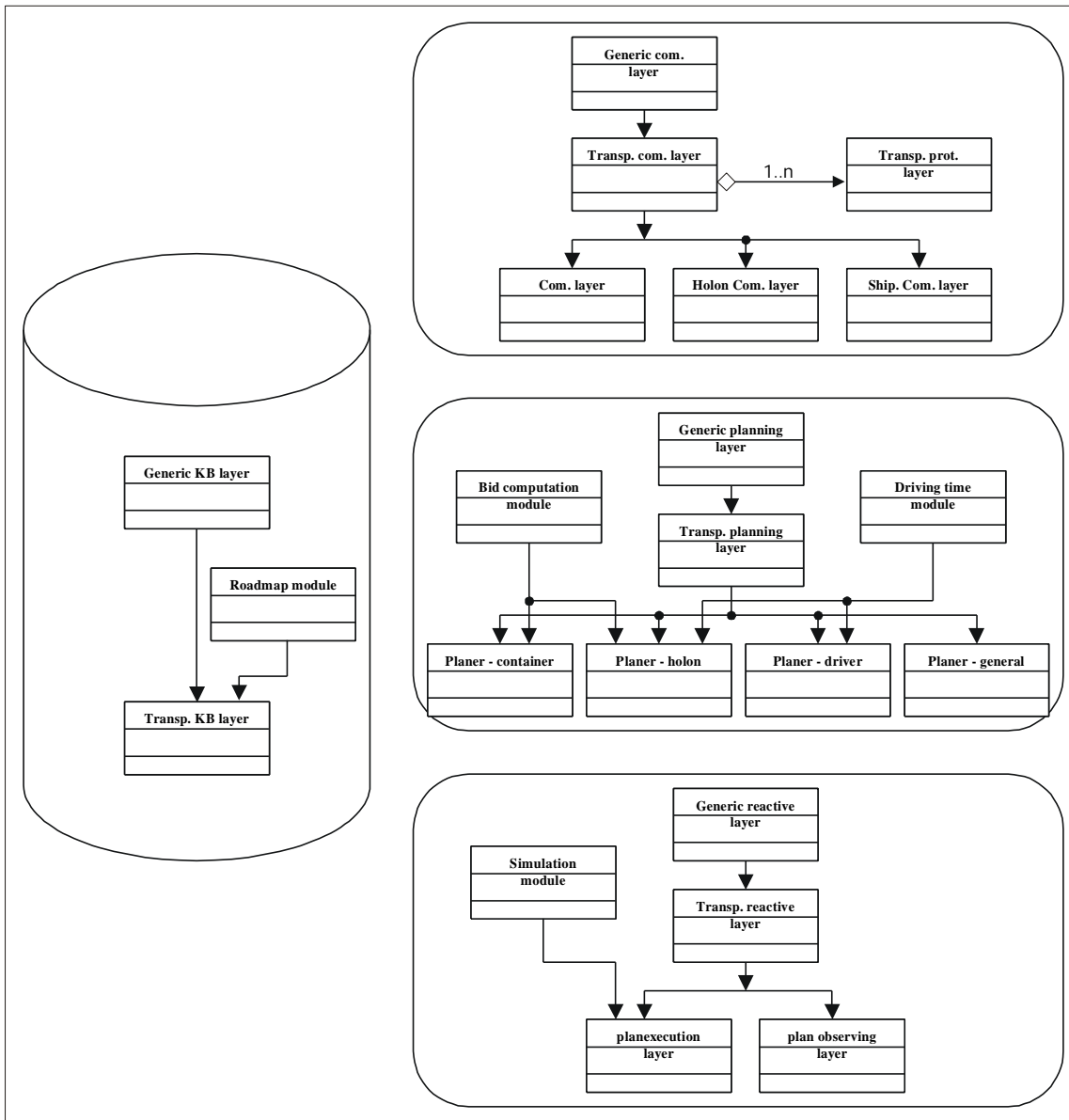
At last there is one additional level which contains only modules for some difficult computations which need specific algorithms to get quick solutions for a given problem. These modules contain efficient algorithms for solving particular problems. It is desirable to change these modules easily without reimplementing the whole level for that special kind of agent which need these changes.

The jobs of the planning layer are divided into a hierarchy of domain independent to domain specific levels. This technique of structuring the planning layer into different levels is close to the technique of transferring the ability of objects to other objects like it is done in object oriented programming. In our domain there are only three to four levels useful. In domains with a lot of different types and groups of agents which share different partitions of functionalities it might be necessary to build more levels.

## 2.4 The Reactive Layer

The realization of the reactive layer differs from one domain to another. It depends on the actions the agent should execute and on the kind of sensors providing information about the environment. In our transport domain there are only few sensors. The road trains might be equipped with an GPS-System which returns the position of the truck. Upon the actual time and the position of the road train the agent is able to check its plan and if there is time enough to serve all orders correctly. In other domains there are much more sensors thinkable, e.g. if the agent represents a robot, it might have several infrared sensors etc. So this shows that the structure of the abstraction levels really depends on the domain of the agents.

In our domain the abstraction hierarchy is divided into three levels. First, there is a generic reactive layer which observes the status of the agent. It has to control whether the communication works and if all sensors and actors are in order. This level is responsible for technical correctness of the hardware and the technical surroundings of the agent.



**Figure 3.** The object hierarchies within the agent architecture

The next level creates a correlation between reactive and planning layer. This level of the reactive layer has to check if all entries in the knowledge base are still okay, and if not, it has to initiate some action which will rectify the knowledge base. However the processing of the plans of the agent differs from the knowledge handling. If a fault in the future of the plan is noticed, say that the execution of the plan would be impossible if the agent does not modify its plan, this level has to generate a call to the planning layer of the agent to build a new plan on the base of the current sensor data and knowledge base.

The third level of the reactive layer is responsible for the correct execution of the actions which are described in the plans of the agent. Therefore this level has to be able to communicate with the relevant hardware components to execute the tasks at hand. This level is responsible for short time observation whereas the level above is considered to do a long time observation.

Analogous to the planning layer there is another level below which can be seen as a module level. It contains some algorithms for quick replanning to get out of critical situations and some modules for a kind of simulation. Simulation is necessary to make a visualisation of the activities on the one hand and on the other hand it is essential to project the situation to the future to see what can happen. From a software developer's point of view the life time of these modules is short. So it is necessary to change this modules without a lot of work and without changing the whole layer.

## 2.5 The Knowledge Base

To build a generic knowledge base for all kind of scenarios is not easy because knowledge base systems are very complex and cannot be divided in independent parts. Hence, we decided to restrict to two abstraction levels.

Due to the fact that most knowledge bases are build up on databases, the highest knowledge base level mainly prepares interfaces between database and agent layers. If only a standard SQL-database has to be included, this level should provide some methods which enables the agent to request information from the database using the SQL syntax.

In our domain there is no ,active' knowledge base needed which has to draw inferences about the knowledge. The knowledge base layer has only to be able to store data about the resources of the agents and it has to save the plan information if the system is going down. From an abstract point of view the knowledge base is just a data pool for an agent. However, this level includes all interfaces to maybe different database systems and to systems which can hold persistent data.

In the next lower abstraction level there are algorithms to draw inferences. This level includes the mechanism which constitutes a real knowledge base system.

## 3 CONCLUSION

We have described an approach that allows to design and implement multi agent systems in a simple and convenient manner. By abstracting from several existing agent systems for transportation purposes, we have constructed a generic agent model that in turn serves to derive new specific agents in a straightforward manner and that allows to save redundant implementation effort. The model combines a sophisticated agent architecture with the holonic agent design paradigm.

Our future work aims at implementing a computer aided software engineering tool that supports the automated generation of agents independently from programming languages and hardware platforms. We consider our generic agent model being an appropriate basis for this purpose.

## REFERENCES

BFV99: H.-J. Bürckert, K. Fischer, and G. Vierke. *Holonic Fleet Scheduling with TeleTruck*. In Proceedings of the Second International Conference on Computing Anticipatory Systems (CASYS'98), 1999

BV99: H.-J. Bürckert and G. Vierke. *Simulated Trading Mechanismen für Speditionsübergreifende Transportplanung*. In H. Kopfer and C. Bierwirth, editors, *Logistik Management - Intelligente I+K Technologien*. Springer-Verlag, 1999.

FF94: T. Finin, R. Fritzon. *KQML – A Language and Protocol for Knowledge and Information Exchange*. Proceedings of the 13<sup>th</sup> International Distributed Artificial Intelligence Workshop, 1994.

FVB99: P. Funk, G. Vierke, and H.-J. Bürckert. *A Multi-Agent Systems Perspective on Intermodal Transport Chains*. In H. Kopfer and C. Bierwirth, editors, *Logistik Management - Intelligente I+K Technologien*. Springer-Verlag, 1999.

GSV99: C. Gerber, J. Siekmann, and G. Vierke. *Holonic Multi-Agent Systems*. Research Report RR-99-03, DFKI, 1999.

Mül96: J. P. Müller. *The Design of Intelligent Agents – a Layered Approach*. Volume 1177: Lectures notes in artificial intelligence, Springer, 1996.

Sea69: J. R. Searle: *Speech Acts*. Cambridge University Press, 1969.

Wei99: G. Weiss: *Multiagent Systems – A Modern Approach to Distributed Artificial Intelligence*. MIT Press, 1999.



# MULTIDIMENSIONAL UTILITY VECTORS IN THE TRANSPORTATION DOMAIN

Michael Schillo, Gero Vierke

Multi-Agent Systems Group  
Saarland University  
Im Stadtwald  
66123 Saarbrücken, Germany  
schillo@ags.uni-sb.de

Multi-Agent Systems Group  
DFKI GmbH  
Stuhlsatzenweg 3  
66123 Saarbrücken, Germany  
vierke@dfki.de

fon: +49 681 302 4578

fax: +49 681 302 2235

## Abstract

Fleet scheduling is known as a complex problem which is only in a simplified form to some degree tractable. Practitioners often claim that soft constraints like driver or customer preferences should be regarded within the planning processes. The Habitus-Field-Theory provides analytical concepts to describe different kinds of worth, also called capital, e.g. money, machines, manpower, or reputation. We classify the forms of capital that are relevant to a transportation company and investigate the operations that transform the capital from one form into another.

This allows to measure the utility of planning operations in several dimensions that correspond to the different forms of capital. Some of these dimensions - e.g. reputation - are too fuzzy to be compared computationally. Nevertheless, the violation of soft or hard constraints corresponds to a loss of some sort of capital and hence to the decrease of a utility parameter. We propose to use the utility vector as a heuristic guidance for a multi-agent fleet scheduling process. Those utility parameters that correspond to violated soft constraints and that cannot be valued automatically are presented to the user who is requested to decide whether the constraint may be violated or not.

**Keywords:** multi-criterion decision making, DAI, rationality, sociology

## 1. Introduction

The TeleTruck CC project (Bürckert et al., 1998) at the German Research Centre for Artificial Intelligence (DFKI) addresses the problem of designing intelligent dispatch agents for shipping (road haulage) companies. Human dispatch agents not only map incoming orders to available trucks and drivers (the typical domain of centralised planning systems), but must also collaborate with the dispatch agents of other companies to pass on orders that may be unprofitable for one reason or another, or simply not convenient. Collaboration between different companies creates an open, dynamic, and potentially hostile environment in which social competence plays a very important role. Our intelligent dispatch agents must not only deal with the social field of inter-agent collaboration, but must also address the social fields of agent-driver and

agent-customer interaction. Each social field has its own logic, its own set of resources (capitals) that may or may not be convertible - for example, a driver may be happy to give up a weekend for extra pay or a couple days holiday, but may resent doing so if it means missing his/her child's Birthday.

Our collaborative shipping scenario requires that a dispatch agent not only knows about its own abilities, but also the abilities, motivations, attitudes, goals, plans and the behaviours of the other competing/co-operating dispatch agents and truck or driver agents. For example, a dispatch agent needs to reason about how reliable the available drivers are, how beneficial business contacts to other dispatch agents are, whether it can trust other agents to fulfil the contracts they commit to, etc. Empirical research (interviews with human dispatchers) further shows that customer/driver models must also take into account the fact that certain customers may not want certain drivers to deliver their goods – adding yet another level of complexity.

Another model hardly considered in current transport scheduling systems is the model for drivers. Again, interviews with human dispatchers tell us that it is very important for the co-operation of the dispatcher with the truck drivers to take into account their personal preferences. Such preferences can be preferred routes, overnight stays, holidays, trucks, cargo etc. They again can put constraints on the total planning and the decisions as to whether to pass on orders to competitors or not. It is therefore important to know (a) when to consider driver preferences; and (b) when to override them. Also, many systems do not take into account the added interaction of driver, customer and vehicle. Certain orders can only be processed if the right driver and the right vehicle are available at the right place – this is especially true when shipping companies deal with transporting food, highly explosive liquids or containers and only have a limited number of vehicles which are available to transport these special kinds of cargo.

Not only does a dispatch agent need to meet the constraints in its planning activities, it also needs to know how important a particular constraint is for medium- or long-term goals. For example, if it discovers that there is no solution for the current set of constraints, it needs to know which constraints can be relaxed. Decisions therefore require an understanding of the relative importance of qualitatively different constraints, which in turn requires an understanding of the relative convertibility of resources between the different social fields.

### **1.1.A sketch of Bourdieu's habitus-field theory**

Sociologist Pierre Bourdieu's concept of habitus consists of a set of dispositions to actions and ways of perception. These dispositions depend on the history of the individual and what it experienced in the past, they may be incorporated or imitated, i.e. learned by observation and acquired by advice. We feel that the concept of these dispositions is a perfect starting point to connect bounded rationality research with the DAI research of social contexts. Furthermore, the fact that Bourdieu emphasises the practical application of his theory and has reported extensively on his practical work, gives us the hope that his methodology can be used for application in DAI. For Bourdieu, the habitus is the result of processes that adapt to the surrounding social structure according to the logic of this social context. This marks the importance and

the influence of the structure of the agent society on the behaviour of the individual, while still explaining how the structure is shaped by the individual.

According to Bourdieu's investigations<sup>1</sup>, the forces of social life cannot be reduced to single dimension. The models of utilitarian approaches are in his view much too simplistic to explain social phenomena. With the logic of the practice, Bourdieu broadens the narrow model of the utilitaristic perspective. He adds other forms of social resources: cultural, social, and symbolic capital. The actors are then classified over a space according to their possession of these capitals. Competition in a field leads to a permanent and latent conflict of actors for transferring one form of capital into another. The substrate of the economical capital is money – it is objectified as “possession”, and institutionalised in the form of ownership rights. The probability of converting money into other forms is “high”, with risks of deterioration lying in social crisis (wars, revolutions, economical crisis). A similar analysis can be made for the other capitals, as for example with the capital of culture (or more precisely, the capital of information). The substrate of cultural capital is “knowledge”. Cultural goods and knowledge are its objectified forms, and it is institutionalised by titles of education. The substrate of social capital can be identified as relationships, which in its objective form, Bourdieu terms this capital “networks”. Its institutionalised forms are titles of aristocracy as individual predicates and the status of profession as an collective schemas. To convert social capital has little reliability and is risky, but often necessary.

## 1.2. Why use habitus-field theory in the shipping domain?

There are several reasons to seek for more universal theories for approaching the fleet scheduling problem. First of all, as Castelfranchi and Conte (1998) state, there is a fundamental problem underlying the notorious notion of rationality as simply economic rationality, namely the merely implicit notion of goals<sup>2</sup>. Not only is the assumption that rationality can be reduced to simply maximising a single utility (be it money, level of satisfaction etc.) is unrealistic. Looking at the practice shows that a whole range of criteria influence the decisions made e.g. in the transportation domain, not all of which can be reduced to a definite monetary worth.

Castelfranchi and Conte also argue that reducing rationality to optimising action selection according to a single measure (cf. the *instrumentalist* rationality, cf. Moser, 1990) makes modelling the domain with the concepts of AI more complicated. Concepts like goals and beliefs are not concepts of such theories. Instead, they call for the explicit modelling of goals (cf. the *substantialist* rationality, cf. Moser, 1990), from which a instrumentalist i.e. economic rationality will emerge, if no other goals are present.

Thus, we are confronted on the hand side with a call for broader concepts of rationality for action and on the other hand side with a sociological theory that tries to cover all the different criteria that are taken into account by human decision makers. This results in what Dederichs et al. (2000) call *socially bounded rationality*. This term covers the concept of decision making based on multiple criteria (beyond only the

---

<sup>1</sup>For an extensive overview of Bourdieu's works see Bourdieu and Wacquant 1992.

<sup>2</sup>There are of course other points of criticism: the omniscience assumption, unbounded computation resource assumption etc. which, however, are commonly agreed unrealistic assumptions in game and decision theory.

economic capital) and takes into account forms of rationality that are shaped by the individuals history. It also encompasses that agents do actively determine their position in their social field and can formulate and reason about goals concerning their positions to pursue their goals.

The application of such a concept is further motivated by the fact that the complexity of the shipping domain is such that the *real* plan space of a dispatch agent is far greater than that covered by existing route planning and cost minimisation dispatch systems. This is clearly reflected by the fact that all the existing systems on the market require human operators to provide the missing levels of (social) competence. Many of the requirements of such systems are strikingly similar to the requirements of autonomous agency: Agents hold inconsistent beliefs, have multiple competing concerns which are qualitatively diverse, and must be robust in the face of hostile and unknowable environments. In building socially competent intelligent agents, we will inevitably address many of the same problems faced by researchers in the field of intelligent autonomous agents - we hope that our approach will provide insights that are beneficial to both communities. In this sense, we also believe that there will be significant synergy between research on socially situated agents and research on *bounded rationality*, or *bounded optimality* (Russell 1997) in the more general AI and computer science communities.

## 2. Application of Bourdieu's concepts to the domain

Let us now investigate how Bourdieu's concepts of capital and action can be applied to the shipping domain.

### 2.1. Capital in the shipping domain

We can identify four types of capital that are relevant to a forwarding firm:

- **Material resources** are physical objects that can be bought or sold and that are required for at least one of the business processes of the company.
  - **Means of transport** are the mobile machines that are needed for the central process of transport: trucks, truck tractors, trailers, semi-trailers, chassis, and containers. The primary capital associated with the means of transport is their availability. The vehicles can drive permanently but there are restrictions regarding the maximal speed, capacity, etc. The secondary capital is the technical condition of the material, wear results in an decreasing state which can result in limited availability (i.e. breakdowns).
  - **Fuel** is a consumable capital that is needed for transport.
  - **Other material resources** are necessary for the remaining business processes not all of which are typical for the transportation domain: offices, storehouses, loading facilities, forklifts, furniture, computers, software, etc.
- **Human resources** are the employees of the company. The capital they supply is work. Unfortunately, their are not permanently available like the material resources, but their availability is limited. The quality of the work strongly depends on the

worker's qualification and experience which by most companies is regarded as a valuable capital. The secondary capital associated with the employees is their *degree of satisfaction*. The quality of the work will decrease if the satisfaction of the employee decreases. This might decrease also the satisfaction of the customer which could result in decrease of income. These notions are quite fuzzy by nature, and it is not possible to map the degree of satisfaction directly to utility. Moreover, the situation that influence the employees' satisfaction individually differ.

- **The dispatchers** are doing the planning for the transportation process.
- **The drivers** are executing the transportation plans.
- **Other employees** are needed for the remaining processes.
- **Social Resources** represent the relationship to companies and persons outside the firm, which are in one way or another important for the firm.
  - The **customers** are the most important social resource since they supply tasks and pay for their execution. The *satisfaction* of the customers is maybe the most vital social capital a company has. Contractual agreements with the customers are also a very useful capital since they guarantee regular tasks.
  - The **partner companies** are those firms that not belong to the customers but are helpful in some other way. This can be other forwarders that serve as sub-contractors, or some kind of service provider, e.g. garages which sell and service the vehicles or banks which provides the vital resource money. Contracts with partner companies are a quite concrete form of capital, the trust in partners and the trust of the partners are - alike satisfaction - rather fuzzy forms of capital.
- **Abstract Resources** are those forms of capital that are not directly associated with objects or persons.
  - An **offer** of an customer to execute a task is certainly a form of capital since it can be transferred into an **order** which - by executing it - can be transferred into an **outstandings** which is almost equivalent to money.
  - **Information** and **plans** should be regarded as a form of capital as well since the success of almost every action depends on it: the driver needs a plan in order to execute his tasks efficiently, the dispatcher needs to know about the current traffic situation, and the factors that influence driver's and customer's satisfaction, etc.
  - **Money** is a special kind of abstract resource, it is not only the symbol for capital. It is the most flexible form of capital since it can be transferred to almost all other forms of capital quite easily. The ultimate goal of an commercial company is to maximise money.

## 2.2.Actions in the shipping domain

The motivation to start a commercial enterprise is to transform the different forms of capital into each other in such a clever way that finally the overall capital - measured in money - increases. An *action* is the transformation of capital. A business process can be

regarded as a chain of capital transforming actions. In an efficient company the processes are optimised in such that the grow of capital is maximised.

Let us partition the actions in the transportation domain into two groups, firstly those that are needed for the central, productive business process of transportation, which we investigate in detail, and secondly those that are not specific for transportation, which we do not explore completely.

Action	<b>transfers</b>	into	<b>influencing</b>
Accept task	Offer	Order	Customer's satisfaction
Reject task	Offer	Nothing	Customer's satisfaction
Planning task	Dispatcher's availability	Plan	
Execute task	Driver's availability, Order, Plan, Availability of means of transport, Fuel	Outstanding money	Driver's satisfaction, Condition of means of transport, Customer's satisfaction
Pass task	Order	Money / Nothing	Partner's satisfaction, Customer's satisfaction
Employ subcontractor	Order, Money	Outstanding money	Partner's satisfaction, Customer's satisfaction

Figure 1: Capital Transfer within the Transportation Process

Figure 1 illustrates the actions that form the transportation process. The process is initiated by the offer to execute a task of a customer. The first action is usually the acceptance of the offer which transfers into an order. This might increase the customer's satisfaction. If the offer is rejected, it is transferred to nothing, and, maybe the customer's satisfaction decreases.

The next step is the generation of a plan for the order. The working time of the dispatcher is transformed into a plan. The execution of this plan transforms the order, the plan, the availability of driver and means of transportation, and fuel to outstandings at the customers. This action can influence the satisfaction of the driver or the customer and the technical state of the components.

The order can also be passed to a partner forwarder. There are two ways to do this. Firstly, to pass on the task, in this case the order is transferred into money, if a charge is claimed, or nothing if not. Secondly, to pass the order to a subcontractor, in this case, the order and money - which is paid to the partner - is transformed into outstandings at the customers - which hopefully exceeds the price paid to the partner. In both cases customer's and partner's satisfaction can be influenced.

In Figure 2 there are some examples for actions listed that are not directly associated with the transportation process. Invoicing usually transforms the outstanding money into liquid money; to buy or to sell material or to employ or to dismiss people transfers money to material or people or vice versa, respectively; to train an employee transfers the employees availability and money to qualification; to serve a means of

transportation transfers its availability and money to a better condition of the material; etc. Completing this list would go beyond the scope of this paper.

Action	transfers	into	influencing
Invoicing	Outstanding money	Money	Customer's satisfaction
Buy material	Money	Availability of material	Partner's satisfaction
Sell material	Availability of material	Money	Partner's satisfaction
Recruit employee	Money	Employee's availability	Employee's satisfaction
Raise salary	Money	Employee's satisfaction	
Dismiss employee	Employee's availability	Money	Employee's satisfaction
Train employee	Money, Employee availability	Employee's qualification	Employee's satisfaction
Serve material	Money, Material availability	Condition of material	Partner's satisfaction

Figure 2: Capital Transfer within the Non-Transportation Processes

### 3. Conclusion and Future work

#### 3.1. Practical implications for planning

What has been said so far is the call for reaching more adequate and realistic systems in the shipping domain by integrating representation and reasoning over many capitals (or kinds of resources), preferences (crisp and soft constraints) that can range from economic goals to the aim of achieving a certain reputation. We view the term multi-criterion decision-making as central for this effort and list the following methods for dealing with finding a total order for multi-dimensional vectors:

1. Reducing the multi-dimensional utility vector to a single number by linearisation. This requires the determination of factors expressing the dependency between the dimensions of the utility vector. We believe that this is not a promising approach as it implies that such factors exist, which we deny. Investing in partner satisfaction is not measurable (without extensive empirical studies) and therefore not available to the calculations of the system. Looking at this problem as the vector maximisation problem (Zimmermann, 1987) is a similar technique, with similar shortcomings.

2. Setting thresholds to eliminate option by option until only one option remains. This can only be realised in interaction with the human dispatcher on a case by case basis and we doubt that this is a considerable relieve to him/her.
3. The rough set theory approach (cf. Pawlak, 1991) consists of the assignment of objects (in our case actions) characterised by a number of feature attributes (in our case the effects on the different capitals of the shipping company) to a category (e.g. accept, reject, offer to partner company, request for additional information<sup>3</sup>). This approach is highly appropriate in so far as it (as a special case of rule learning) models how to learn the classification the human dispatcher would use. Despite its intriguing interactive features, the theory relies on a static, constant set of rules that does not change over time. For an adequate model of the real shipping domain this is unacceptable.
4. Heuristics for choosing operators that cannot be distinguished.
5. Application of multi-objective decision making from fuzzy set theory (cf. Zimmermann, 1987).

We propose the latter two options as they seem most feasible. At the moment we plan interviews with human dispatchers to elicit such heuristics and determine how these heuristics are formed. We expect to find that they are partially explicit (by education) and partially implicit (learning by doing, experiences made by collaboration in the same company). Fuzzy set theory appears promising as it explicitly deals with the notions of not convertible features of objects.

### **3.2. Implications for the graphical user interface to the human dispatcher**

If a dispatch decision support system is equipped with representations of the kinds of capitals as mentioned above, we envisage the following improvements for the interface of the system to the human dispatcher. First of all, a dramatic speed up can be reached by neglecting to replan current schedules for every incoming order, as the system will be equipped with heuristics to pre-evaluate these orders and filter out any orders the company is very unlikely to be able to perform. Secondly, and more important, we hope to reduce the number of actions the system suggests to a minimum. Thirdly, we believe that the visualisation of the different capital improvement estimations for each of the suggested actions will make it possible to further reduce the number of alternatives at a glance by the human dispatcher. Finally, receiving this kind of feedback from the human dispatcher, the system has the necessary interaction that user modelling requires to further annotate the utility vector of a possible action.

### **Acknowledgements**

This work is supported by DFG under contract Fi 420/1-1. We would like to thank Sociologists Michael Florian, Andrea Dederichs and Frank Hillebrandt from the Department of Technology Assessment at the Technical University of Hamburg-

---

<sup>3</sup>Cf. Slowinski, 1993 for an application of this theory to preference modelling

Harburg for most fruitful discussions. Further thanks go to Christof Klein for his valued feedback on the sociological parts of our work.

## References

- Bourdieu, P. (1987). *La Distinction: critique sociale du jugement*. Translated by Nice, R.: *Distinction : A Social Critique of the Judgement of Taste*. Harvard Univ Press.
- Bourdieu, P. and Wacquant, L. (1992) *Invitation to Reflexive Sociology*, University of Chicago Press.
- Bürckert, H.-J., Fischer, K. and Vierke, G. (1998). Transportation Scheduling with Holonic MAS -- The TeleTruck Approach. *Proceedings of the Third International Conference on Practical Applications of Intelligent Agents and Multiagent systems (PAAM'98)*.
- Dederichs, A., Fischer, K., Florian, M., Hillebrandt, F. and Schillo, M. (2000). Spoken Communication, March 2000.
- Castelfranchi, C. and Conte, R. (1998). Limits of economic and strategic rationality for agents and ma systems. In: *Robotics and Autonomous Systems, Special Issue on Multi-Agent Rationality*, vol. 24 (3-4):127-139.
- Moser, P.K. (1990). *Rationality in action:General introduction*. Cambridge University Press.
- Pawlak, Z. (1992). *Rough Sets: Therretical Aspects of Reasoning about Data*. Kluwer, Dordrecht.
- Russell, S. (1997) *Rationality and Intelligence*. *Artificial Intelligence*, 94, p. 57-77.
- Slowinski, R. (1993). Rough set learning of preferential attitude in multi-criteria decision making. Jan Komorowski and Zbigniew W. Ras, editors. *Proceedings of the 7th International Symposium on Methodologies for Intelligent Systems (ISMIS'93)*, volume 689 of LNAI, Trondheim, Norway, Springer Verlag.
- Zimmermann, H. J. (1987). *Fuzzy Sets, Decision Making, and Expert Systems*. International Series in Management Science/Operations Research. Kluwer Academic Press, Boston - Dordrecht - Lancaster.



# Optimization of the On-site Transportation Logistics of a Chemical Factory using Multi-Agent Systems

Thomas Thum<sup>1</sup>

**Abstract.** This paper presents a possible way to optimize the on-site transportation logistics of engineering works and chemical factories, which have evolved without steering or deliberate planning over time, to save costs and energy. The general methodology employed is the problem solving cycle of systems engineering. Hereby object-oriented process-based simulation, agent technology and multi-agent systems were used as tools for system analysis. In an underlying application example of optimizing the on-site transportation logistics a chemical factory a potential to save costs of transportation of more than 20 % could be realized, without having to spend much money for investment in capital equipment. With this project a contribution has been made to demonstrate the suitability and characteristics of agent technology and multi-agent systems to optimize industrial logistic processes and transportation networks. The greatest advantages of multi-agent systems may be seen in their inherent adaptivity, which gives them a great advantage compared to the conventional monolithic systems when there is a fast evolving business environment.

**Keywords:** Agent Technology, Chemical Factory, Engineering Works, Management of Complexity, Multi-Agent Systems, Optimization, Planning and Scheduling, Systems Engineering, Transportation Logistics

## 1 INTRODUCTION

Still today, there are engineering works and chemical factories, which show structures regarding their on-site transportation logistics, that have 'evolved' without steering or deliberate planning over time. But structures of that kind may not be optimal in view of costs of transport and energy consumption.

The business environment and market forces compel industrial manufacturing companies to act as efficient as possible. Therefore immediately the question is raised, how an 'evolved' transportation logistics of an industrial manufacturing site can be restructured and specifically improved, to save money and energy. By example of a chemical factory this paper will outline, how this question can be treated successfully using multi-agent systems.

Hereby this paper will show a possible way, how new opportunities opened up by current research results of computer science, i.e. the specialized knowledge of the subjects of object-oriented process-based simulation (cf. Fischer and Ahrens [12]), intelligent agents (cf. Nilsson [18]) and multi-agent systems (cf. Ferber [11], Weiss [23]), can be used to generate value by managing the com-

plexity of industrial logistic processes and especially transportation networks better. Taking into account relevant engineering aspects, it is therefore intended to make a contribution to narrow the gap between the aforementioned fundamental research subjects and the application of 'intelligent' objects and multi-agent systems in industry by building a 'engineering'-bridge in between.

In particular this paper will present, in which way the application of the methodology of systems engineering and especially the construction and deployment of multi-agent systems were used as tools to identify and realize the optimization potential regarding the transportation logistics of the chemical factory mentioned, and some experiences gained doing this.

## 2 METHODOLOGY

### 2.1 General Framework: Problem-Solving Cycle of Systems Engineering

Systems engineering is a well-established methodology for management of complexity [6]. Because the design or optimization of a transportation logistics is a complex problem, the methodology of systems engineering was used as the general framework to solve the optimization problem at hand.

Especially there is a problem-solving cycle provided by systems engineering, which can serve as a template for proceeding and contains the following individual steps:

- *Situation-Analysis:* The purpose, why the existing transportation logistics is analyzed first, is to gain a better understanding of the user requirements, the relevant phenomena taking place in the factory, their relationships and so on. Especially the following information is obtained [14]:
  - The type and temporal sequence of the individual transport jobs
  - The existing transport vehicles and further equipment
  - The existing roads on the factory grounds and their conditions including potential points of danger
  - Specific costs and energy consumption of the individual units of the transportation process.
- *Setting of specific goals:* The results of the previous analysis provide the information basis, on which the general objective of the project is reviewed and put in specific goals, which are operable and measurable.
- *Design and analysis of distinctive variants:* The first step to reach these goals is to design distinctive and (hopefully) improved variants of the transportation process. The setting-up of these individual variants represents the key for the generation

---

<sup>1</sup> Dr. Thum · Process-Engineering and Project-Management Consultancy, Neue Strasse 49, D-21073 Hamburg, Germany, email: IngBueroDrThum@t-online.de

of value, because only through this new possibilities come into existence, which make an improvement of the existing transportation logistics possible.

At the same time, by designing distinctive variants there will be created a good instrument for the evaluation of the individual variants and especially the 'best' variant, since a broad basis for comparison is provided this way.

When designing new variants of the on-site transportation logistics in the present example, the following criteria have been taken into account:

- Deployment of alternative transport vehicles or other aiding equipment
- Use of new roads on the factory grounds
- Modification of the generation time and sequence of the individual transport jobs (as far as possible for limiting operational reasons)
- Modification of the existing scheduling policies
- Improvement of the information flow between the plants, warehouses and transport vehicle drivers.

The previously described design of various variants is accompanied by a simultaneous analysis of the individual set-up variants. There are especially two reasons for this. On the one hand, it is possible to improve the individual designs in an iterative, gradual manner by better understanding and modifying them derived from this, instantly. At the same time, the analysis is the basis for the determination of the operating behavior and measurement of the performance of each individual variant, leading to the next step.

- *Evaluation of these variants and decision-making:* The individual variants of the on-site transportation logistics, which were developed in the previous step, are now evaluated thoroughly especially according the following criteria:
  - Operating behavior of each variant, e.g. reliable supply of the individual plants with raw materials
  - Necessary investment in new equipment
  - Life-cycle costs of optimization measures
  - Reduction of energy consumption
  - Reduction of the risks of accidents and legal liabilities.

Taking into account 'political' considerations [2] the 'best' variant is eventually selected. In this context 'best' means comparative instead of absolute 'best', whereat imperative constraints have to be satisfied and the weighted sum of the aforementioned criteria serves as a measure for comparison. Doing this, the individual weight factors depend on the specific situation and the subjective evaluation of the relative importance of the various criteria (cf. [2]). Through this, conflicting criteria are reconciled, and a satisfying compromise is reached. It shouldn't be forgotten here, to take the cost and time required to draw up the underlying study into account, and a sensible compromise between the corresponding expenditures of time and money and the quality of the decision obtained should be sought for, too.

- *Implementation:* Finally the 'best' variant is implemented, which means not only the procurement and installation of the possibly necessary equipment, but also the information and training of the operating personnel (if this hasn't been done already) and the documentation of the measures taken.

With this, the problem-solving cycle comes to a close, and the value of the improvement of the on-site transportation logistics is generated in fact. The reduction of costs of transportation and consumption of energy takes place as striven for from the beginning.

## 2.2 Modeling and Simulation of the On-site Transportation Logistics: Multi-Agent Systems

The formal modeling and computer simulation of the designed variants are indispensable tools in the analysis, understanding, individual optimization and evaluation of these variants. Especially these tools help to identify and evaluate the operating characteristics and economics of the individual variants in virtual reality before the 'best' variant is implemented in actual reality. It is well known [16], that this scientific way of action is cheaper, faster and less dangerous compared to the empirical trial and error procedure used in earlier times (and sometimes today, too).

By the way it should be mentioned for orientation, that within the edifice of systems engineering simulation is categorized as a method affiliated to operations research, whereas operations research in turn is classified as a technique or aid of systems engineering [3]. On the other hand, modeling is seen as a basic element of the "modeling and simulation enterprise", constituting a connecting link between reality and simulation, and therefore having a character of its own [24].

Even when the foundations of discrete-event system specification and process-based discrete-event simulation have been laid already some years ago [24], [4] and [20], computer science recently has opened up new possibilities of modeling and simulation, e.g. object-oriented process-based simulation, agent technology and multi-agent systems (cf. [12], [18], [11], [13] and [23]). And these new possibilities fit naturally to the problem at hand. Therefore the various variants of the on-site transportation logistics, which had to be analyzed and evaluated, were modeled in the form of multi-agent systems.

The following lines give an overview and some details, how this modeling and simulation was done.

There are in principle two 'contrary' ways to model a transportation process: top-down and bottom-up from local view. But these two ways work together during the construction of the system model. First the whole system is mentally partitioned into separate components (possibly only preliminary and / or in several successive steps), until the size of the resulting individual components (i.e. objects and agents) is 'small' in terms of DeMarco [10]. In this context 'small' means, that the size of the components is of such a kind, that it doesn't exceed the powers of the human imagination, even when every relevant detail of a particular component is taken into account.

Then, each individual component, which was determined by the prior mental partition, is modeled formally, and furthermore, the modeled components of the system are related to each other (maybe hierarchically structured). This way, i.e. by suitable partitioning of the system and reunification of the modeled components, a foundation is laid, on which tools can be built, which enable to manage the complexity of a transportation system successfully, which is 'larger' than 'small' – and this is, what the present topic really is about.

In practice, it should be recommended, to start with 'simple' models, which may not represent reality truly, but can be under-

stood and handled easily. Only when the corresponding size of complexity has been coped with successfully, size and complexity of the system model should be increased. Therefore, in practice, the main emphasis when constructing models of transportation processes lies – in the opinion of the author – on the bottom-up approach.

A note has to be made regarding the value and validity of the models, which are constructed by this way: Properly speaking, according to Popper, a model can never be validated but only refuted [21], cf. also [16] and [20].

But for purposes of practical application, thoughts of Kossen and Oosterhuis [16] combined with theorems of Zeigler [24] may help, which can be formulated loosely and informally the following way, taking into account the question that should be answered or problem that should be solved with the aid of the constructed model: If all the relevant objects of the real world are present in the system model, if the structure is preserved adequately when portraying the real world (or rather the relevant part of it) into a system model, and if, during the simulation experiments serving for the purpose of prediction, all objects stay within their range of validity, then prognosis is allowed, otherwise not. (This discourse is by no means philosophical. It is, according to Kossen and Oosterhuis, the essence of engineering activities, like design. If, for instance, one of the relevant elements is not present in the system model, or fails during simulation, the model as a whole can give erroneous results [16].)

And here is where – from point of view of the author – agent technology can play its trump cards: Agents (which are extensions of objects and, according to Taylor “are nothing more or less than objects with rules and legs” [22]) enable in modeling and simulation a much better representation of the active and situation based behavior of transport vehicles compared to the restriction on passive objects imposed by classical object-oriented modeling.

When constructing a system model according to the “bottom-up” principle from local view – which may be sometimes the only way to build such a model – the whole system may be more than the sum of its parts, and the system behavior may show emergent phenomena, which cannot be derived from the structure and behavior of the individual components. But simulation experiments are an excellent tool to trace the dynamic behavior of the whole system, and therefore to determine the emergent phenomena, which depend on the arrangement of and relationships between the various components as well as on their individual structure and behavior, and cannot be foreseen and predicted otherwise.

In other words: Under the assumption, that the system models represent the real transportation logistics adequately (and they should), the modeling and simulation of the individual transportation system variants are excellent tools to determine, understand and design (!) the emergent phenomena, which show not only in simulations, but (would) arise in reality as well. Therefore the modeling and simulation of on-site transportation processes of industrial manufacturing companies are excellent tools to manage the complexity of these processes successfully.

Let us take a closer look at the individual components of the transportation system models now:

The transportation vehicles (e.g. trucks, tractors and fork-lift trucks) were modeled as reactive, reflexive autonomous agents (classification in accordance with Ferber [11]). Their operating behavior was determined by establishing suitable rules, which were embedded in rule bases local to the respective agents. The principle of an activity cycle of a fork-lift truck agent is depicted

in Figure 1 for illustration. One can see the sequence of cognition (i.e. perceiving, thinking and deciding / selecting a new transport job) and acting (i.e. going with or without load and loading respectively unloading) revolving the whole time again and again until all its transport jobs are done or a simulation run stops. Trailers, which are also elements of the transportation system, were modeled as passive objects, i.e. entities independent of and separate from the tractors, which were modeled as agents, too.

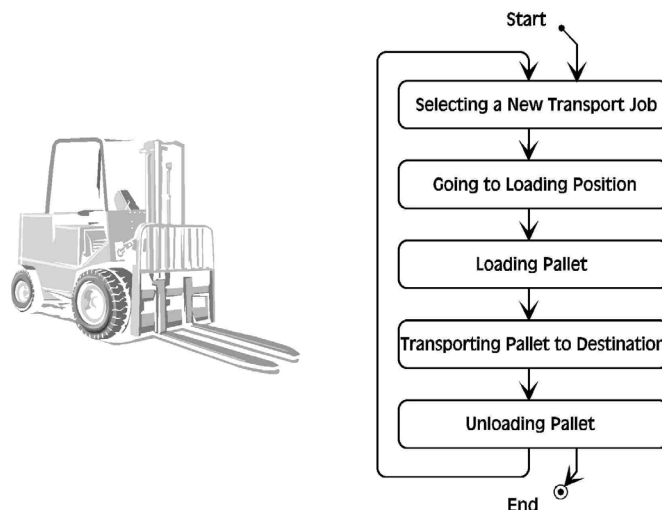


Figure 1. Principle of the Activity Cycle of a Fork-lift Truck Agent

The premises of the factory were modeled as a graph [1] or rather as a network. The individual plants and warehouses, which all are parts of the factory and generate as well as absorb transportation jobs, were modeled as active generators (i.e. sources) respectively passive sinks. The sequence of work periods and breaks of the operating personnel (i.e. succession of shifts) were taken into account.

Figure 2 depicts the principle of the structure of the on-site transportation system for illustration. The arrows symbolize the journeys of the transport vehicles or the flow of material. Please note that the manufacturing plants and warehouses may be not regarded as components of the system but as parts of the environment influencing the system.

Besides, in the present case a ‘tele’-communication medium, which also stores the individual transport jobs, had to be included in the system model. It enables the individual agents to inform themselves about the state of the outside ‘world’ and especially pending transport jobs whenever needed and also from whatever location they are at the moment. Therefore by the communication medium they are supplied with the necessary information, based on which they decide, which transport job to carry out next.

This means that an indirect, decentralized control of the system takes place. In other words: The coordination of the individual activities of the transportation agents is directed by the content of their individual rule bases as well as dependent on the system state.

Throughout the whole modeling phase, the assumption is workable, that the individual agents can be seen as ideal parallel processes (in terms of computer science), which are active simultaneously. But this illusion has to be given up when implementing the model on a ordinary computer, which operates in a sequential

manner. Therefore an simulation 'kernel' was used [12], which constitutes an excellent bridge between the ideal world of thoughts and models and the real world of sequential computing machinery. This simulation 'kernel' enables the individual agents and generators to run in a 'autonomous', quasi-parallel manner on a sequential machine, and provides synchronization and scheduling mechanisms and statistical evaluation tools, in addition.

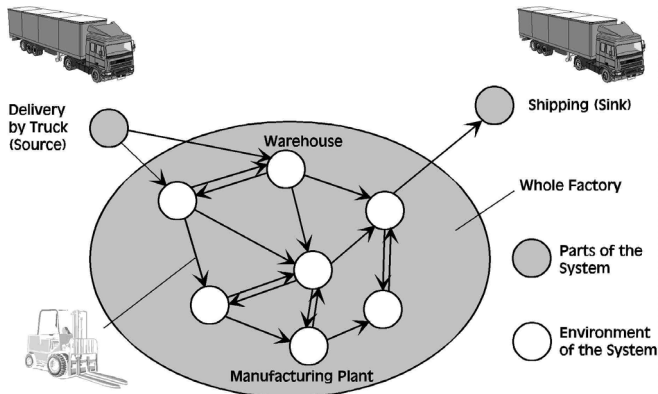


Figure 2. Principle of the Structure of the On-site Transportation System

The analysis and evaluation of the various variants of the transportation process took place by simulation experiments, which represented the operation of the individual variants of the transportation logistics over a period of one week each time. This time period showed to be sufficient to portray the operation characteristics of the variants with sufficient accuracy.

By running the simulation experiments the following information was generated for each individual variant of the transportation process:

- Operating behavior
- Operating costs
- Energy consumption.

The information obtained provided the basis for the economical evaluation of the individual variants and the following decision-making and implementation process.

### 2.3 Economics: Cost-benefit analysis and Discounted-cash-flow Calculation

To find out, which of the variants is economically best, a discounted-cash-flow calculation was made in each case considering possibly necessary capital expenditures [7]. Effects of taxation were not taken into account for sake of simplicity.

The economical calculations provided the answers to the following questions: Which of the available variants is best in economical terms? If there has a capital expenditure or rather investment to be made, does this make sense? What is the relating ROI?

## 3 RESULTS

### 3.1 Application example

The following results have been achieved with regard to the optimization of the on-site transportation logistics of the chemical factory, which served in the present context as an industrial application example:

- An optimization potential for cost reduction of considerable more than 20 % could be identified and realized, without having to spend much money for investment in capital equipment.
- At the same time a relevant reduction of energy consumption could be achieved, which constitutes a beneficial contribution to the maintenance and improvement of the reputation of the chemical company as a side effect, because the energy saving helps the company to comply with its 'duties' being imposed by the Eco Audit Regulation issued by the European Community.
- To realize the optimization potential, it is necessary, to install a communication network like the one depicted in Figure 3. In this example the plants, warehouses and drivers of the transportation vehicles or rather the corresponding personal computers or terminals are connected by the local area network of the manufacturing site. Part of this network are one or more relay station(s), which are located on the factory premises and link the transport vehicles to the local area network by wireless radio communication.

Besides there is a server program running on a dedicated machine, which represents a 'blackboard' on which the transport jobs are managed. (A certain similarity of this structure with the classical blackboard architecture (cf. [18]) is surely not coincidentally, however there is no explicit blackboard in the 'mind' of each agent (as described in [18]), but similar to Brooks [5] the 'world' serves as its own model or rather is represented by only *one* blackboard common to all agents.)

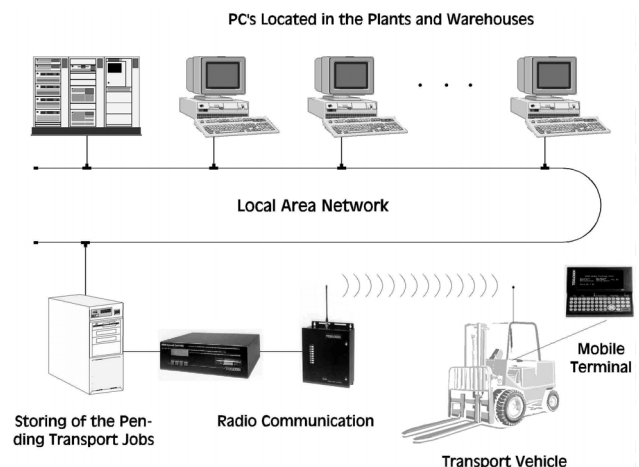


Figure 3. Principle of a suitable Communication Network

- There are psychological advantages too, which arise out of the decentralized, implicit control used in this example, and should not be overlooked: Because by optimizing the transportation

logistics this way, no substantial part of their autonomy, they were used to, had to be taken away from the operating personnel. Therefore the necessary changes of their working procedures and behavior were accepted by them without dissent. This is why there was no risk, that they would 'sabotage' the implementation of the measures suggested by the study previously drawn up and stop the optimization potential to be realized through this.

### 3.2 Results in general

One can say, that a contribution has been made, to establish a methodology to identify and realize a possibly existing optimization potential of on-site transportation logistics of industrial manufacturing sites, which show 'evolved', not steered or deliberate planned structures. The practical value of this contribution has been verified by an industrial application example.

Therefore a stepping-stone has been laid, which can serve as a starting point for carrying out similar or derived industrial and commercial applications, as well as for the extension and further refinement of the methodology.

## 4 DISCUSSION AND CONCLUSION

### 4.1 General Meaning of the Methodology and the Results

In the opinion of the author the following general evaluation of the methodology used and results obtained can be made and the conclusions listed below can be drawn:

Systems engineering is a well-established methodology to manage complexity, the integration and application of multi-agent systems as a tool for modeling and simulation of complex systems even enhances this already very good methodology.

The sole application of analysis on its own is not enough, when we want to build new or optimize existing transportation networks, manufacturing processes and plants and other man made constructs for industrial or commercial purposes. The analytic thinking has to be accompanied if not to say led by design thinking to successfully manage our factories and businesses. DeBono puts it this way: "You can analyze the past, but you have to design the future" [8]. Therefore the solitary application of multi-agent systems, which themselves mainly are tools for analysis, may be not enough in many application cases.

Mental models lead the way to formal and computer models. They are generated by vivid imagination and / or careful observation of reality – an only 'mechanical' application of computer models without deep understanding of the underlying reality and its possibilities may be not enough to solve industrial and commercial problems.

The 'intelligent' behavior of the whole (transportation) system may be seen as an emergent phenomenon arising from the cooperation of the various agents, even if the individual agents are not very intelligent (cf. Brooks [5] and Ferber [11]). Basic examples to illustrate this kind of phenomenon, which cannot be attributed to an individual component of the system alone, are e.g. the emergent course of action of an individual transportation vehicle (which depends not only on its internal rule base, but develops out of the interaction with its environment, i.e. especially on the course of action of the other transportation vehicles) and the ac-

cumulation of product or the running dry of raw materials in certain places of the transportation network or factory (which both may be unwelcome and should therefore be avoided or otherwise taken care of).

### 4.2 Engineering aspects

Even if modeling and simulation alone are not sufficient, they are – in the opinion of the author – nevertheless indispensable and very valuable tools to manage complexity, and especially the complexity of the on-site transportation logistics of industrial manufacturing sites. Especially multi-agent systems are an excellent way for this purpose, because they are (at least in principle) easy applicable and enable an approximation of actual by virtual reality with high quality.

Multi-Agent Systems are a very natural way of modeling transportation processes and transportation logistics especially for the following reason: Every component which takes up energy (especially fuel, electricity, food) is an active component of the system and may be fully or partly autonomous (depending on the system design). Therefore it is better to model such components as active objects or rather as agents instead of passive objects.

Systems, which are designed as multi-agent systems, are modular and therefore easily adaptable and extensible, especially when the system environment or the user requirements change.

Multi-agent systems are a fine tool to understand and manage emergent phenomena, which are not welcomed – if not to say feared of – in classical engineering, because they arise often unexpectedly and are always hard to grasp and control.

Multi-agent systems are structured modular on principle, and can be constructed of individual modules or components. Therefore it might be worthwhile, if there would be a possibility to exchange or trade prefabricated components to construct multi-agent systems more easily, faster and cheaper by integrating work already done by others. But the realization of this idea depends not least on common standards, which – at least as the author knows at the moment – have to be established and agreed on, yet.

With regard to more information on the construction and validation of multi-agent systems in the present context, the reader is referred to section 2.2 of this paper.

Not least, it should not be overlooked, that the proper communication and documentation of constructed models, the operating principles of simulators used, as well as the results of the study and implementation phases of projects are important aspects for the acceptance and successful application of multi-agent systems in industry and the further business world, too.

### 4.3 Adaptivity of Logistics

Without getting involved deeper in the ramifications of adaptivity—there are two basically different mind-sets of businesses and many flavors in between: the organization can be seen as an engineered machine or as a living organism (cf. [9], [17], [22])—one can say that the ability to adapt to a business environment evolving at a ever faster pace is and will be an essential success factor for survival and well-being of organizations.

This fact is valid not only for organizations as a whole, but is passed on to its parts. Therefore it is necessary, that even the transportation logistics of a manufacturing organization shows a high degree of adaptivity.

Object technology is an ideal foundation for building adaptive business systems [22]. Because agent technology is based on object technology, this feature of object technology is transferred to agent technology and multi-agent systems directly. And possibly herein may lie the greatest advantage of agent technology and multi-agent systems in logistics: Software systems based on these technologies may be considerably more adaptable compared to conventional monolithic software systems and therefore yield much more value for the user.

#### 4.4 Work to be done

As has been shown in section 4.2, the user friendliness is an important characteristic when constructing and using simulators for industrial and commercial applications besides functionality. Parunak puts it this way: The tools used to develop systems for practical applications (i.e. running simulations for industrial use) must be packaged [19].

The multi-agent simulation system constructed and used by the author, is at the moment based on textual operation. Even if it offers the complete functionality required at the moment, there are some unfulfilled wishes regarding its ease of use.

For example there should be added a graphical user interface and component library, to make it easier for the user, to transfer mental models into running computer models. It would be very comfortable to have a scanner combined with an interpreter to read in the layout of the factory documented usually by existing maps or blueprints. In addition, an animation of the resulting dynamics of the transportation process would be advantageous, to make it easier for the user to understand the dynamics of the system generated by the simulator and to present the system behavior to his clients.

The existence of the features given in section 4.2 are a prerequisite, if the simulator should serve as a stand-alone system applied by an user of a chemical company or engineering works himself. De Geus sees this requirement, i.e. that managers of industrial companies are able to operate and 'play' with a simulator by themselves, to be very important for the methodology of modeling and computer simulation to be accepted along a wide front [9].

Besides, if necessary, there should be an enhancement of the individual agents, for example an extension regarding their rule base, to be able to test other, modified or refined operating policies. Another direction, which could be followed, is to extend the present off-line use of the simulator into an on-line application for the purpose of on-line optimization and control of on-site transportation processes (by control is meant above all a decision support for the drivers of the transport vehicles).

Not least, there is the possibility of developing other industrial and commercial applications based on the experience gained in the construction and use of individual agents and multi-agent systems, leading especially to the fields of supply chain management and automation of trade, i.e. business-to-business e-commerce.

#### 4.5 Résumé

To summarize and give the prospects in the opinion of the author: This paper and the underlying project strive to make a contribution in forging links between current findings of computer science,

especially in the fields of parallel processing, intelligent agents and multi-agent systems, and the industrial application of these technologies in the field of transportation logistics.

As already said and shown by a specific industrial application example, intelligent objects and multi-agent systems are fine tools, which help in general to manage complexity and especially to design and optimize the on-site transportation logistics of industrial manufacturing companies, e.g. engineering works and chemical factories. But the employment of multi-agent systems alone, which are mainly tools of analysis, is not sufficient, it has to be completed by a methodology for creative design of systems, to find better ways of doing things.

Multi-agent systems will enable to construct and manage very large and involved system models, which can nevertheless be an approximation of actual by virtual reality of high quality. Through this, for example they open up the possibility to model the logistics of the factories combined with the manufacturing processes when required (Zeigler et al. have contributed to the underlying basic theory regarding this only recently [25]). This means there will be tools, which can help to search for (and attain) global optima of the whole factory, and to think another step further, even the whole supply chain.

But the greatest advantage of multi-agent systems may be seen in their inherent adaptivity, which gives them a great advantage compared to the conventional monolithic systems when there is a fast evolving business environment.

To close with high expectations: Multi-agent systems and its neighboring fields will probably help us to even better understand the logic and organization principles underlying complex living beings, like individual biological organisms, organizations or whole societies (cf. Miller [17], Kelly [15]). It is to be hoped, that the relating bio-logical knowledge will merge with our techno-logical know-how, so that we will be able to make our technical processes and socio-technical systems even more integrated with itself and into the greater whole for the benefit of everyone.

#### ACKNOWLEDGMENTS

Even if the present paper has been written by only one author, the underlying project was influenced and supported by other people, to whom I want to express my gratitude.

First of all, I want to thank Mr. Michael Klussmann, Director of Operations of the "Chemische Fabrik Dr. Weigert", Hamburg. His acute observations of his on-site transportation logistics and astute foreseeing of the regarding optimization potential gave rise to the project. He supported our work with all his strength and was always an excellent interlocutor. It was a pleasure to work for and with him.

Next, I want to thank Prof. Joachim Fischer and Dr. Klaus Ahrens of Humboldt University, Berlin. By their book they gave me an excellent introduction to object-oriented event-based simulation and opened up the world of discrete-event system simulation and parallel processing for me. Especially, without the use of their simulation 'kernel' [12], this project would not have been possible.

Furthermore, I'd like to thank Dr. Reinhard Bachmann, Department of Technology Assessment, Technical University of Hamburg-Harburg. He drew my attention to the ideas of multi-agent systems, mechanisms of interaction between individual

human (and artificial) agents as well as emergent phenomena arising from the dynamics of complex systems.

As well, I want to thank Prof. Bernhard Lang, Professor for Digital Multimedia Systems at the University of Applied Sciences of Osnabrueck, Germany. In many discussions he has been a critic of my work and source of good ideas especially concerning object-oriented modeling and programming and parallel processing on standalone computers and distributed networks.

Last, but not least, many thanks to my wife Hannelore for her patience and support when working the many hours acquiring and advancing the theoretical foundations of the project, which underlie this paper.

## REFERENCES

- [1] Antonakos, James L. and Kenneth C. Mansfiels Jr.: Practical Data Structures Using C/C++. Upper Saddle River, NJ: Prentice-Hall, 1999.
- [2] Athey, Thomas H.: Systematic Systems Approach. Englewood Cliffs, NJ: Prentice-Hall, 1982.
- [3] Becker, Mario: SE-Wissensbaum (Knowledge-Tree of Systems Engineering). Supplement to [6].
- [4] Birtwistle, G.M.: A System for Discrete Event Modelling on SIMULA. London: Macmillan Press, 1979.
- [5] Brooks, Rodney A.: Intelligence without Representation. *Artificial Intelligence*, 47: 139-159, 1991.
- [6] Daenzer, W.F. and F. Huber (eds.): *Systems Engineering: Methodik und Praxis*. 9<sup>th</sup> ed. Zurich: Verlag Industrielle Organisation, 1997.
- [7] Daeumler, Klaus-Dieter: *Grundlagen der Investitions- und Wirtschaftlichkeitsrechnung*. 7<sup>th</sup> ed. Berlin: Verlag Neue Wirtschaft-Briefe, 1992.
- [8] De Bono, Edward: *New Thinking for the New Millenium*. London: Penguin Books, 2000.
- [9] De Geus, Arie: *The Living Company: Habits for survival in a turbulent business environment*. Boston, MA: Harvard Business School Press, 1997.
- [10] DeMarco, Tom: *Controlling Software Projects: Management Measurement and Estimation*. German ed. Wolframs Fachverlag, 1989.
- [11] Ferber, Jacques: *Multi-Agent Systems: An Introduction to distributed artificial intelligence*. Harlow, England: Addison Wesley Longman, 1999.
- [12] Fischer, Joachim and Klaus Ahrens: *Objektorientierte Prozesssimulation in C++*. Bonn: Addison-Wesley, 1996.
- [13] Fischer, Klaus; Christian Russ and Gero Vierke: *Decision Theory and Coordination in Multiagent Systems*. Saarbruecken: German Research Center for Artificial Intelligence – Research Report RR-98-02, 1998.
- [14] Fischer, Wolfram and Lothar Dittrich: *Materialfluss und Logistik: Optimierungspotentiale im Transport- und Lagerwesen*. Springer Verlag, 1997.
- [15] Kelly, Kevin: *Out of Control: The Rise of Neo-Biological Civilization*. German ed. Bollmann Verlag, 1997.
- [16] Kossen, N.W.F. and N.M.G. Oosterhuis: *Modelling and Scaling-up of Bioreactors*. Chapter 24 of *Biotechnology Vol. 2* (eds. H.-J. Rehm and G. Reed): *Fundamentals of Biochemical Engineering*. Weinheim: VCH Verlagsgesellschaft, 1985.
- [17] Miller, James Grier: *Living Systems*. University Press of Colorado, 1995.
- [18] Nilsson, Nils J.: *Artificial Intelligence: a new synthesis*. San Francisco, CA: Morgan Kaufmann Publishers, 1998.
- [19] Parunak, H. Van Dyke: *Industrial and Practical Applications of DAI*. Chapter 9 of [23].
- [20] Pidd, Michael: *Computer Simulation in Management Science*. 2<sup>nd</sup> ed. Wiley, 1988.
- [21] Popper, Karl: *Alles Leben ist Problemlösen*. 4<sup>th</sup> ed. Munich: Piper Verlag, 1999.
- [22] Taylor, David A.: *Object Technology: A Managers Guide*. 2<sup>nd</sup> ed. Reading, MA: Addison-Wesley, 1997.
- [23] Weiss, Gerhard (ed.): *Multiagent Systems: A Modern Approach to Distributed Artificial Intelligence*. Cambridge, MA: MIT Press, 1999.
- [24] Zeigler, Bernard P.: *Theory of Modeling and Simulation*. Wiley, 1976.
- [25] Zeigler, Bernard P., Herbert Praehofer and Tag Gon Kim: *Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems*. 2<sup>nd</sup> compl. rev. ed. San Diego, CA: Academic Press, 2000.



## Negotiation Models for Agent Technologies in Logistics

<i>A. BRUN AND A. PORTIOLI-STAUDACHER: NEGOTIATION-DRIVEN SUPPLY CHAIN CO-ORDINATION FOR SMALL AND MEDIUM ENTERPRISES</i> .....	55
<i>F. LOPES, N. MAMEDE, A. Q. NOVAIS, H. COELHO: NEGOTIATION IN A MULTI-AGENT SUPPLY CHAIN SYSTEM</i> .....	61



# Negotiation-Driven Supply Chain Co-Ordination for Small and Medium Enterprises

Alessandro Brun<sup>1</sup> and Alberto Portioli-Staudacher<sup>2</sup>

**Abstract.** Leading Enterprise Resource Planning (ERP) vendors are offering new tools and solutions allowing co-ordination through centralized planning and control of the overall Supply Chain (SC); yet there are many SCs made up of Small and Medium Enterprises (SMEs) in which such an approach is not viable, since the components of the chain are not willing to give up their planning autonomy for co-ordination's sake; moreover, information privacy is also an issue.

In this paper, a Multi Agent System (MAS) architecture is presented, aiming to improve planning co-ordination in networks of independent enterprises.

## 1 PROBLEM SETTING

### 1.1 The need for Supply Chain co-ordination

For decades, management efforts have been for decades focused on *internal* competitive weapons, such as cost reduction and increased quality; such efforts resulted in better and more automated information systems, and in organizational projects such as set-up time reduction, variety reduction, group work.

Lately, the increased awareness that internal improvement are necessary, yet not sufficient, to survive in the fierce competitive scenario, fostered markets globalization, and concentration processes, as a way to exploit three *external* weapons which could result in improved service level and customer satisfaction: exploitation of scale and mix economies, creation of distributed technologies, and evolution of R&D. In particular, the management of many enterprises decided to develop strategic partnerships with suppliers and customers, and to increase their "degree of internationalization".

This process had brought to increased problems of co-ordination all over the Supply Chain. As a consequence, there is a need for a closer integration of the logistic functions, production planning and quality control, as witnessed by the AMR Research [1]: *recent studies in various industries have shown that Supply Chain performance could improve if trading partners were able to work more closely to assess expected customer demand and plan supply accordingly.*

Having understood the possibility offered by this evolution process, major ERP vendors are starting to offer commercial solutions for Supply Chain Planning and Control.

<sup>1</sup> Politecnico di Milano, Piazza L. da Vinci, 32 – 20133 – Milan - Italy – Email [Alessandro.Brun@polimi.it](mailto:Alessandro.Brun@polimi.it)

<sup>2</sup> Politecnico di Milano – Email [Alberto.Portioli@polimi.it](mailto:Alberto.Portioli@polimi.it)

### 1.2 User requirements

This paper reports an application of Agent Technology realised within the European Project Esprit 25226 - MUSSELS (Multi-Site and -Stage Enterprise Logistic control System).

The objective of MUSSELS is to provide a possible solution the co-ordination problems of SMEs, supporting an agile manufacturing framework, where logistics and production management systems of customers, suppliers and manufactures are managed in a coordinated fashion. Access to common information is supported by an open system based on Internet technology.

The *User Driven* software analysis and design, based on several Supply Chain case studies, developed thanks to the experience of a number of industrial partners, allowed the development of a tool tailored to the final users needs. The specific requirements of the industrial environment investigated in the case studies, were then extended; the result of this activity being a list of more general requirements, which could apply to a great variety of SC structures.

Most of the requirements identified are related to a series of shortcomings shared by Production Planning and Control (PP&C) and ERP systems of the SC analyzed in the Case Studies:

- most of Single- and Multi-Site MRP Systems do not consider resource-capacity constraints;
- they do not consider parallel production lines or sites;
- enormous amount of data are required.

On the one hand, there is a need for more accurate planning of most critical items and resources; on the other hand, omitting the trivial many planning operations and computations, it would be possible to abridge the often too cumbersome production planning and control system.

### 1.3 Pros and cons of existing SC solutions

Potential users of a SC Planning and Control System can be divided into three broad categories: multi-site enterprises, with a single ownership, Supply Chains characterized by the presence of a big node dominating the whole chain, and Supply Chains constituted by several independent SMEs.

In order to be able to take swiftly the right decisions, the management of a SME should have on-line access to some key information, such as the stock levels by the suppliers, or the delivery status of a particular order; in a word, they should be adequately supported by a sound information system, giving them visibility on the whole chain.

It is true that some recent ERP systems support a wide gamut of visibility and data sharing functionality, therefore overcoming some of the disadvantages presented in the previous sub-section. Yet, such ERP systems have two patent disadvantages: adopting such system is very expensive (in terms of license costs, hardware required, effort of employee and hired consultants, and so on); and

running such systems in an efficient way is not so simple (ERP systems often offer a huge amount of information, most of which are disregarded). Arguably, a SC of SMEs would rather adopt a lean (and, as a consequence, cheap) planning system, collecting and processing only essential information.

MUSSELS comparative innovation are presented in Table 1.

**Table 1.** Comparative Innovation in MUSSELS

Available Products (SAP's APO, I2's Rhythm, AspenTech's MIMI SCS, Numetrix/3 EP, ...)	MUSSELS
Medium to Large Enterprises	Small and Medium-sized Enterprises
Production network controlled by one Company	Network of independent enterprises
One goal across the network determined by dominant partner	Multiple co-operative goals: coordination instead of control
Central cockpit (point of control and information management)	Co-operative process: no single point of control, shared information
Single point of decision making	Distributed decision making, possible multi-agent approach
Optimize overall profit (supply chain solution)	Establish a feasible and agreed solution and maintain stability
Replacement approach: substitute for pre-existing PPC systems	Add-on approach: enhancement to pre-existing PPC systems
High entry costs	Low entry costs
Difficult to join and leave	Easy to join and leave

A particular effort was devoted during the system architecture design phase, due to the overall goal of addressing the most important requirements highlighted, while granting that no frills or useless procedures were included in the system. The result was a modular system featuring four main functionality:

- a simulation environment, called Logistic Chain Analyzer;
- SC planning, performed by the Multi Site Planning system;
- electronic *Kanban* supported by the Kanban Control System;
- Multimedia Training module, containing self-training tutorials.

This paper focuses on the Agent based architecture, which constitutes the engine of the Multi Site Planning (MSP) system.

## 2 THE ARCHITECTURE OF THE MULTI-SITE PLANNING SYSTEM

This module is responsible for the support of medium term planning of units within the supply chain, in term of assuring coherence in timing of production planning, and compliance with capacity constraints. Users of the MPS could enjoy feasible plans, both in terms of temporal and capacity constraints, which satisfy as much as possible the whole chain.

It sometimes is the case that a feasible solution could not be found, unless modifying the existing constraints. In such a case, a negotiation process among the various nodes would result in some nodes giving up a fraction of their requirements, and some other nodes paying to have their requirements satisfied, in order to sort out an overall feasible plan.

### 2.1 Functionality

- **Quick-response What-if Analysis:** the MSP aims at synchronizing the whole supply chain, considering that a quick response is mandatory for coping with the unexpected events that characterize real life operations.

- **Criticality identification:** in order to avoid computational overload, the MSP identifies – along several criteria – a subset of items, called *Critical Items*, and consider in its plans this reduced set of items.
- **Feasibility check:** the module automatically checks the consistency among the plans of the various nodes of the supply chain and the feasibility of the overall plan.
- **Priority management:** it manages order priorities, reflecting both conditions inside the plant (e.g. status of other orders, due date, etc.), and conditions of the other units in the chain.
- **Visibility:** for a better co-ordination of the activities of the nodes, this module gives visibility on the overall material flow.
- **Automatic Plan Adjustments:** in many cases lead-time in the chain is increased because of misalignments and inconsistencies in the behavior of the members of the chain. By performing automatic alignment of plans in different nodes of the supply chain, the MSP helps in reducing lead-time and WIP.

### 2.2 The Theoretical Background

When facing the problem of designing a planning system, the User Requirements made it clear that:

- the environment is characterized by a degree of uncertainty;
- the system is time constrained, since it has to provide the answer in a bounded amount of time;
- the problem is *inherently distributed*: there are multiple, conflicting objectives; decision autonomy must be preserved, since nodes are often owned by different companies;
- there is the need to locally adapt the system to the physical structure of the logistic chain, which may change in time.

As a clear consequence of the first and second observation, the possibility to use classic AI planning is excluded, since AI planning assumes a static environment, and known auction outcomes; and

since the whole plan is made and agreed to before action; as a consequence, since being P-space complete, AI planning is unusable in any time-constrained system.

Unlike a classic AI planning system, the possibility of adopting an agent-based system is not jeopardized by observations 1, 2, and 4. Yet, the major strength in adopting a MAS is that MAS suit well to distributed problems: in particular, in a MAS, each agent is a locus of self interest. This means that, by distributing both information and decision making among the agents, it is possible to effectively address the requirements of information privacy and decision autonomy.

Moreover, in those cases in which the request for an action and the action itself are two distinct logical moments, modeling the decision makers with agents would result in a more realistic representation of the problem.

### 2.2.1 Co-ordination

MAS allow to distribute data, decision making, and even control; this way, no agent has a complete picture of the problem: how is it possible to guarantee an organized behavior?

*Co-ordination* could be defined as *the process of managing interdependencies between activities* [6]. In a MAS, the high-level interaction abstraction designed to achieve co-ordination by ruling the interaction among agents is referred to as *co-ordination model* [3]. Co-ordination models could be centralized or decentralized; the co-ordination mechanism could be implicit or explicit.

Centralized co-ordination models require a single locus of data and decision making; main advantage is the possibility to reach optimality; yet such models rely on a central point of failure, and perform badly in dynamic environments; Decentralized co-ordination models works when data and decision making are spread among a number of agents; though being rarely optimal, such models are more robust, and grant information privacy and decision autonomy. The choice of preserving decision autonomy prevents from designing a centralized co-ordination model.

Implicit co-ordination mechanisms consists in the definition of a set of *social rules*, specifying legal actions for the agents; if such a restricting protocol is correctly designed, the action allowed to the agents result, by definition, in a coherent behavior, even without communication among the agents; in the case of explicit co-ordination, agents explicitly argues over who does what, how, and when; as a consequence, agents need to communicate their intentions, goals, results, and state. The effort required to design a complete set of social rules makes it preferable to go for an explicit one: co-ordination is achieved via communication among agents. This choice is based on the results of a previous work, in which different co-ordination models have been compared [2].

### 2.2.2 The communication protocol

The design of a communication protocol is influenced by the trade-off of computation versus communication: this is particularly the case when an agent needs a certain pieces of information, which could either be obtained via computation or via communication. Huhns [5] argues that communication is generally more expensive, slower (it could lead to prolonged negotiation), and less reliable (no

certainty to come to an agreement) than computation; yet, it is qualitatively superior.

Many different communication protocols have been proposed in recent years; in particular, the MSP relies on the so called Contract Net Protocol (CNP) [4]. It is possible to establish a parallel among the CNP and the process of (many-to-many) supply planning, by mean of a simple example (see Figure 1):

- due to a shortage in a certain raw material, the agent in charge of planning the plant D (agent D) announces the existence of a new requirement;
- the agents of plants A, B, and C evaluates the announcements; being able to supply the requested material, the agent A and B submit their bids;
- after receiving the bids, agent D selects the best bid (say, the one of agent B), and communicates to the bidders;
- agent B is then committed to supply to agent D the specified amount of raw material; it is possible that agent B has to modify its plan in order to abide by such new commitment.

The reader could notice, in the above example, the following minor terminology adaptations: node  $\Rightarrow$  agent; task  $\Rightarrow$  requirement; contractor  $\Rightarrow$  supplier. At any rate, a more substantial modification have to be made, if the CNP, originally conceived to manage the process of task distribution over a network of processors (nothing more than a few computers), is to work in a complex organizational domain.

A change of perspective should take place: the CNP was conceived to work in a *goal-oriented domain* (i.e. a work of black and white: each task is a goal; achieving it is good, not achieving it is bad) and it is now to work in a *worth-oriented domain* (i.e. a fuzzy domain: a value is associated to the achievement of each single task). The user requirements clearly stress that we are coping with a worth-oriented domain, since at least two objective functions are considered: minimizing planning nervousness, and minimizing additional production, holding, and transport costs. Needless to say, the above mentioned are not Boolean, *black-or-white* objectives.

Another problem associated with the CNP concerns nodes benevolence: in the original CNP, eligible nodes spontaneously offer themselves as perspective task executors. In real world problems, an agent is a locus of self interest, and it will perform a job only when an adequate recompense is expected.

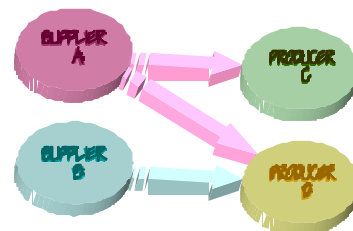


Figure 1: Example of many-to-many planning process

### 2.2.3 Modeling a Worth-oriented Environment: why a Price Based Mechanism?

The most natural solution to the problems spotted in the last subsections is the introduction of a Price-Based negotiation mechanism, the main reason behind this choice being the fact that assigning a price to each activity allows to trade-off among conflicting objectives. Moreover, a Price-Based mechanisms reveals information about scarcity and value of resources; which would be hardly known in case of a central planner.

Two are the main real market analogies which could be used to implement a Price-Based mechanism: the free market economy and the auction. Both analogies allow efficient allocation of goods. Yet, in case of the free market economy the convergence toward the set of *market clearing prices* may take a long time, while the duration of an auction depends on the particular protocol selected.

Therefore, in time-constrained contexts, it is preferable to implement an auction framework, making sure that the selected protocol guarantees fast response. The MSP is based on a series of independent multi-units auctions. [7].

## 2.3 The Architecture of the MAS

Each site in the supply chain is represented by three agents:

- a Procurement Agent (PA), which supports the functionality of the Purchase Department;
- a Sales Agent (SA), which interacts with customers; a customer might be the procurement agent of another site using the MSP, or a physical person, working in an internal department (e.g. the planning or the logistics department);
- a Production Management Agent (PMA), which helps the production management department, and receives information from the shop-floor and the inbound and outbound logistics.

There are only two kind of interactions among agents belonging to different plants of a Supply Chain: the SA of a Supplier interacts with the PA of the Customers; and the PA of the nodes at the same stage of the supply chain may interact with one another. Each agent is to a certain extent autonomous; in this way, a consistent part of the decision making process can be carried out without the need of user intervention. In particular:

- the PA acts as a broker of information, in that it gathers a lot of data via communication with the suppliers' SAs, and then communicates to the PMA only relevant information;
- the PMA is the interface among the Production Planning and Control system internal to the node and the MSP; it has (partial) authority on the plant facilities;
- the SA is committed towards the customers; in case of problems, it has the authority to decide which order to delay (according to some priority criterion set by the user); moreover, it represents its site in the negotiation process with other agents.

## 3. THE PILOT PLANT: PIRELLI LTD.

The SC in the case of Pirelli Ltd. is composed by one central manufacturer, located in Carlisle (UK), supplied by a small number of dependent companies, and a single distributor, managed by the Pirelli group central logistics. The area of interest for the MUSSELS project is limited to Carlisle site and to the main raw material supplier, located in Burton on Trent (UK). Carlisle plant is devoted to the production of car and light truck tires.

The production flow carried out in Carlisle plant can be represented as a five-step flow, as different levels of an overall continuous process, as illustrated in Figure 3:

1. Compounding: Banburies (mixing machines) merge rubber, oils, carbon black and chemical elements to produce inputs for semi-manufacturing lines. Different kinds of mixtures are realized in Burton and Carlisle or purchased. Also textile and metallic fabric creel rooms are involved at this level to rubberize raw fabrics. Burton is the main supplier of both bulk materials required by semi-finishing units, and raw material used in the compounding area of Carlisle plant.
2. Semi-finishing: at this level the main activities are calendaring, extrusion and cutting of mixtures; the number of components manufactured in a day is really huge (250 different items out of 300).
3. Assembling: 24 semi-automatic and 3 fully automated machines build up *green tires*.
4. Curing: green tires are vulcanized and molded in presses, in a synchronized flow towards finishing units.
5. Finishing: tests are performed on every item coming from curing, to detect potential defects.

In Carlisle plant production is managed by a pull system: two manual kanban loops are present in the process, one between semi-finishing and assembly, the other one between assembly and curing. Burton deliveries satisfy the most of Carlisle requirements. Burton plant is working 10 shifts in 6 day, out of 21 shift (without stopping) a week performed in Carlisle. Set-up times are also longer for Burton than for Carlisle process. As a consequence, it is not possible to establish a pull connection between the two plants. The communication of the weekly plan results in a push system.

The visibility provided by such a system is rather poor: visibility in semi-finishing and assembling is limited to just one production unit ahead, while inter-site visibility is limited to the weekly communication. The time required to get stock information is too long and subject to errors.

The MSP provides accurate on-line information about stock levels and production performance in the main points of the SC. The access to inter-site stock information is granted only to authorized sites and people. A web connection between sites improves controls on capacity levels and plant productivity. Data flows (event driven or on demand) could guarantee overall visibility and provide automated demand leveling, reducing problems of overproduction or stock-out.

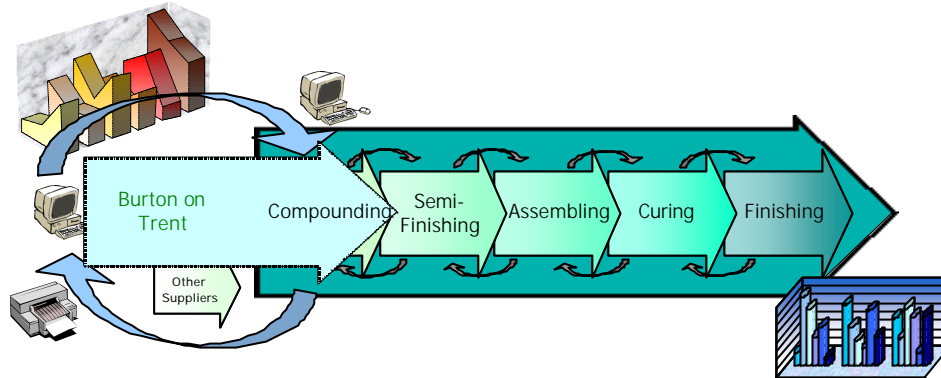
Burton plant could better distribute resources and production demand with full on-line visibility of accomplished orders to consider new orders (coming from other SC customers, for

instance), in a short time, derisory if compared with similar check performed by fax or telephone.

The production is not at all leveled. While an item run out of production process, another is run in production and, as a consequence, timely stops of production to empty pipeline are not managed. The MSP is able to communicate and calculate the quantity of material to be produced in the different stages to empty

the pipeline. Modifications of priority are planned to facilitate change of production in such a continuous process with no need of change kanban board.

In the first implementation, the Pirelli Pilot was restricted to three sites (Carlisle, Burton, and one external supplier). The MAS is therefore constituted by 9 different agents: PAS, PAC, PAB, PMAS, PMAC, PMAB, SAS, SAC, SAB.



**Figure 3:** Pirelli manufacturing process and Supply Chain

The automatic re-planning process is initiated whenever a PMA observes a situation exceeding some pre-defined threshold value. The objective of this process is to find a solution with the least negative effect, both in terms of previous plans disruption and additional production, holding, and transport costs. We shall explain the whole process with an example.

1. Due to a machine breakdown, the PMAB has highlighted the need of a re-planning.
2. It communicates the problem to the SAB, which has the authority to decide which job to delay.
3. The SAB asks to the PAC (its only customer) its priority list (plainly speaking, this is like asking a customer "I'm sending you all the stuff but one component; which one would you dislike less to be delivered late?").
4. Taking into account the plans issued by PMAC, PAC generates a priority list, and then sends it to SAB.
5. Working in collaboration with PMAB, SAB considers several unfeasible plans, and selects the best (i.e. the less bad) by considering both the just received priority list and its internal priority list.
6. It then communicates to PAC that certain deliveries have been put off, and the new (expected) delivery date.
7. PAC transfers this information to PMAC, which now has updated information on expected delivery terms.
8. After evaluating the new delivery dates, PMAC may ask to PAC to find an alternative supplier. In this case, an auction procedure is initiated.
9. the PAC auctions all the materials which causes unfeasibility, and communicate it to SAB and SAS;

10. SAB and SAS evaluate their eligibility by asking to the respective PMAs whether they are able to produce the auctioned components, and at which costs;
11. the PAC receives the bids; it then awards the best bidder, and communicates the choice to the other bidders also.

When an auction is open, the message circulated to all the SAs indicates the item type, quantity, due date. When submitting a bid, an agent has to indicate the maximum quantity it is able to supply, and the price. In case the auctioneer required more than one item, the bidder is not compelled to consider all the item in its bid.

Finally, the auctioneer selects the set of bids allowing it to minimise an internal cost function. This operation could be performed in a reasonable amount of time by exploiting a branch-and-bound algorithm.

## 5 CONCLUSIONS

A set of baselines has been defined by Pirelli in order to measure the improvement following the introduction of MSP, including increased daily output, widened product mix, a higher number of mould changes per day, shorter throughput times, lower stock levels and scrap rates, transport savings and so on. Improvement will be assessed by the end of the project (December 2000).

Expected improvements are such, that the investment in the MSP system will have a pay-back time shorter than 18 months.

At the moment the authors are conducting further simulations in order to assess the validity of the proposed model in case of three stages supply chain.

## REFERENCES

- [1] The report on Supply Chain Management, AMR Research, July 1998.

- [2] A. Brun and A. Portioli, 'Effective Supply Chain Coordination: an investigation', *Proceedings of the Second International Workshop on Intelligent Manufacturing Systems*, 411-419, (1999).
- [3] P. Ciancarini *et al.*, 'Coordination models for Multi-agent Systems', *AgentLink News*, **3**, 3-6, (1999).
- [4] R. Davis and R.G. Smith, 'Negotiation as a Metaphor for Distributed Problem Solving', *Artificial Intelligence*, **20**, 63-109, (1990).
- [5] M. Huhns, 'Multi-agent Systems: basics and interaction', *Proceedings of the First European Agent Systems Summer School*, (1999).
- [6] T.W. Malone and K. Crowston, 'The interdisciplinary study of coordination', *ACM Computer Surveys*, **26**, 87-119 (1991).
- [7] T. Sandholm, 'Automated Decision Making', *Proceedings of the First European Agent Systems Summer School*, (1999).

# Negotiation in a Multi-Agent Supply Chain System

Fernando Lopes<sup>1</sup>, Nuno Mamede<sup>2</sup>, A. Q. Novais<sup>1</sup>, Helder Coelho<sup>3</sup>

**Abstract.** The integration of isolated supply chain functions into a global system and the coordination of multiple functions across the system are two open problems. In this paper we address these problems by organizing the supply chain as a network of intentional agents with planning and negotiation competence. Intentional agents are autonomous computational processes with mental states composed of beliefs, goals, plan templates, and intentions. They generate and execute intentionally plans of action towards the achievement of their goals. This paper describes a mechanism for planning from second principles. The agents operate in multi-agent systems and situations often arise in which their plans conflict with the plans of other agents. Conflict resolution is crucial for achieving effective multi-agent coordination. Negotiation is the predominant tool for resolving conflicts of interests. This paper presents a generic negotiation mechanism, a new approach for defining a structure for negotiation problems, a method for generating negotiation proposals, and methods for generating counterproposals either by making or not making concessions.

## 1 INTRODUCTION

The study of autonomous agents has received a great deal of attention in the recent years and a number of different approaches have emerged as candidates for their design. An interesting approach, called mental state approach, views the agents as computational entities with internal structures composed of human-like mental attitudes, such as belief and desire [3; 13; 33]. This is the approach adopted in this work. The major reason for this choice is the direct prescription of the agent architecture given by mental state models.

*Intentional agents* are autonomous computational processes with internal structures (mental states) composed of *beliefs* (facts about the environment and the other agents), *goals* (world states to be achieved), *plan templates* (procedures for achieving specific world states), and *intentions* (commitments to achieve world states or to perform actions). The agents are belief-desire-intention or BDI agents. The beliefs and the plan templates represent their informational states. The goals and intentions capture, respectively, their motivational and deliberative states [31].

The agents are charged with generating plans for achieving their goals. Two major approaches to plan generation are discussed in the literature. Planning from first principles generates plans from scratch [1]. This approach has several limitations, notably the invariable nature of the planning process over time - to solve a problem equal to one previously solved there is a need to repeat exactly the same operations [18].

Planning from second principles tries to overcome the limitations of planning from scratch by reusing previously generated plans. This approach has been used successfully in the development of agents for real world applications [15; 16; 25]. This work seeks to develop agents for complex application domains (e.g., supply chain management). To this end, the approach of planning from second principles is adopted. A *plan* is defined as a collection of plan templates structured into a hierarchical and temporally constrained And-tree. Plan generation is based on the retrieval, adoption, interpretation, and decomposition of plan templates stored in plan libraries.

The agents try to execute their plans in an effective and competent way. They are not antagonistic and do not intentionally try to deceive or thwart the plans of other agents. However, situations often arise in which their plans conflict with the plans of other agents. A *conflict of interests* is defined as a social concept involving two or more agents that have plans requiring mutually exclusive world states to exist. These plans are called *incompatible* and cannot be executed together. Conflicts are detected individually by each agent using pre-specified axioms.

Negotiation is the predominant tool for solving conflicts of interests. The key issues associated with the development of negotiating agents can be structured into three categories [26]: (i) process category - deals with global negotiation mechanisms, (ii) language category - focus on negotiation protocols and languages, and (iii) internal category - handles the intra-agent part of negotiation. This article concentrates on the process and internal categories and presents: (i) a generic negotiation mechanism, (ii) an approach for defining a structure for negotiation problems, (iii) a method for generating and evaluating negotiation proposals, (iv) a method for generating counterproposals without making concessions, and (v) a set of negotiation tactics.

The first contribution of this article is to present a *negotiation mechanism* that handles *multi-party*, *multiple-issue* and *single or repeated* rounds. Each round consists of three main tasks: (i) definition of a structure for a negotiation problem and generation of proposals, (ii) improvement of rejected proposals or generation of new proposals without changing problem structure, and (iii) re-definition of current problem structure. The mechanism is generic and flexible (see subsection 5.1).

---

<sup>1</sup> INETI, Estrada do Paço do Lumiar, 1699 Lisboa Codex, Portugal, {flopes, anovais@dms.ineti.pt}

<sup>2</sup> IST, Avenida Rovisco Pais, 1049-001 Lisboa, Portugal, Nuno.Mamede@acm.org

<sup>3</sup> Faculdade de Ciências, Campo Grande, 1700 Lisboa, Portugal, hcoelho@di.fc.ul.pt

The second contribution is to describe an approach for defining a structure for negotiation problems. A *negotiation problem structure (NPstruct)* is defined as a hierarchical And-Or tree expressing a relationship between negotiation goals and issues. *NPstruct* is generated from an incompatible plan. It represents a natural link between planning and negotiation and allows the direct integration of these two cognitive capabilities into a control architecture for intentional agents. Also, problem structure generation acknowledges the role of conflict as a driving force for negotiation.

The third contribution is to describe a method for proposal generation. A *negotiation proposal* is defined as a set of facts (leaves of *NPstruct*) with logical values (*true*, *false*, or *any*). A fact is a tuple with two components, namely a negotiation issue and a value for the issue. Proposal generation is based on the resolution of *NPstruct* and supports *cooperative* and *non-cooperative* negotiation behavior.

The fourth contribution is to introduce a method for generating counterproposals without making concessions. The method, called *bargaining issue manipulation*, consists mainly of adding to a proposal negotiation issues considered superfluous, in the hope that the other parties will feel strongly about these issues.

The last contribution is to describe a set of *negotiation tactics* for generating counterproposals, by making concessions.

Intentional agents equipped with the negotiation mechanism are currently being implemented in Prolog. Our work follows an experimental line. The form of the mechanism, and the assumptions it makes, has been guided by our experiences in developing agents for the domain of supply chain management. In earlier work, we described intentional agents and presented part of the formal negotiation mechanism (see [23]). In this article, we define the concept of conflict of interests and continue the description of the negotiation mechanism. We also introduce the type of application domains we are interested in, by describing a simplified multi-agent supply chain system.

The remainder of this article is organized as follows. Section 2 describes a simplified multi-agent supply chain system. Section 3 presents the components of the mental state and the planning mechanism of intentional agents. Section 4 presents the concept of conflict of interests and describes axioms for conflict detection. Section 5 describes the main components of the negotiation mechanism. Finally, related work and concluding remarks are presented in sections 6 and 7 respectively.

## 2 MULTI-AGENT SUPPLY CHAIN SYSTEM

A *supply chain* is a network of facilities that performs the functions of procurement of raw materials from suppliers, transformation of these materials into intermediate goods and final products, and the delivery of these products to customers [8]. The *supply chain functions* range from the ordering and receipt of raw materials, to the distribution and delivery of final products, via the scheduling, production, warehousing, and inventory of intermediate goods and final products.

The *integration* of the multiple supply chain functions has received a great deal of attention in the recent years [35]. However, most work addresses only single functions, such as scheduling or production. To date there exist little work that addresses the

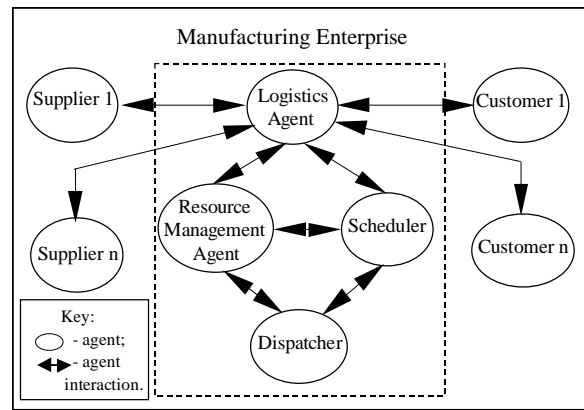


Figure 1. Simplified multi-agent supply chain system

problem of integrating such isolated functions into a complete, global supply chain.

The *coordination* of the supply chain functions has been another active area of research. Also, most research addresses the coordination of two or more supply chain functions, such as production-distribution and buyer-vendor coordination. Despite the importance of the results obtained, the coordination of multiple supply chain functions is still an open problem.

We sketch simplified solutions to the integration and coordination problems in this article. More specifically, we address these problems by organizing the supply chain as a collection of intentional agents that are able to coordinate their activities through negotiation.

### 2.1 System architecture

The supply chain of a manufacturing enterprise is modeled as a multi-agent system [7]. The system is composed of a collection of intentional agents, each responsible for performing one or more supply chain functions. The architecture of a simplified system is shown in Fig. 1. We are currently working on the following agents: logistics agent, scheduler, resource management agent, dispatcher, a number of suppliers, and a number of customers. A brief description of each agent follows.

The *logistics* agent manages the movement of raw materials from the suppliers, the manufacturing of intermediate goods and final products by the enterprise, and the distribution of the products to the customers. He receives customer orders, deviations in schedules which affects customer orders, and resource demands. He originates production requests and supplier orders. He also notices the acquisition of resources.

The *scheduler* is responsible for scheduling and rescheduling activities in the manufacturing enterprise. He receives production requests from the logistics agent, resource problems from the resource agent, and deviations of the current schedule from the dispatcher. He originates detailed schedules and sends them to the dispatcher and to the resource management agent. He also communicates the deviations of the current schedule to the logistics agent.

The *resource management* agent is responsible for managing dynamically the availability of resources in order to execute the scheduled activities. He receives the schedule from the scheduler and the consumption of resources from the dispatcher. He also receives information about the acquisition of resources from the logistics agent. He estimates resource demands and identifies

resource problems. He transmits resource availability to the dispatcher.

The *dispatcher* is responsible for executing the scheduled activities. This agent controls the real time functions of the factory floor. He receives the schedule and the availability of resources. He notices deviations of the current schedule and the consumption of resources.

The *suppliers* sell raw materials and the *customers* buy finished goods. The suppliers receive orders from the logistics agent and transmit their own alternative orders. The customers send orders to the logistics agent and receive alternative orders.

## 2.2 Multi-agent coordination via negotiation

The individual agents of the supply chain system must work in a tightly coordinated manner in order to effectively and efficiently achieve their goals. Coordination is achieved through negotiation. *Negotiation* is a process by which two or more parties verbalize contradictory demands and move toward agreement by a process of concession making and search for new alternatives [29].

Let us introduce an example to illustrate the negotiation process in the multi-agent supply chain system. The example concerns mainly the negotiation between a customer and the logistics agent, but also illustrates the negotiations between the agents within the manufacturing enterprise.

**Example.** Consider that a customer (C) intends to buy 20 computers and sends the following proposal order to the logistics agent (L):

C: I propose to buy 20 high-quality computers with the price of EUR 15000 and the due date for 30 March.

Upon receiving the proposal, the logistics agent negotiates with the scheduler the feasibility of scheduling the production of the order. He also negotiates with the resource management agent the feasibility of producing the desired quantity of computers. The latter negotiations may lead to subsequent negotiations with suppliers of raw materials. If an execution plan is agreed on, the logistics agent uses it to generate a response to the customer. He can respond either by: (i) accepting the received proposal, (ii) sending a counterproposal, or (iii) just ignoring the proposal and aborting the negotiation. Suppose that the logistics agent responds with the following counterproposal:

L: I propose to sell 20 high-quality computers with the price of EUR 20000 and the due date for 30 July.

The next step would be easy if the customer decides to accept the counterproposal. In that case, he just needs to send an acceptance message to the logistics agent. However, if the customer is not satisfied with the counterproposal, he can either send back another counterproposal or abort the negotiation. Suppose that the customer decides to make a concession - he prepares a counterproposal that meets his needs worse than the initial proposal. He decides to adjust the price of the computers and accepts the due date proposed by the logistics agent:

C: Okay, in that case, I propose the price of EUR 18000 and accept the due date for 30 July.

This exchange of proposals and counterproposals will go on until the agents decide to accept a proposal or quit. The negotiation

process can also end because of external events such as missing an agreement deadline.

## 2.3 Negotiation characteristics

Negotiation between agents in a multi-agent supply chain system and, we believe, a wide range of similar systems, exhibit the following characteristics:

1. *two or more parties* - negotiation may involve two parties (e.g., the logistics agent and a customer) or many parties (e.g., the logistics agent and the scheduler, the resource management agent, etc). Multilateral negotiation consists essentially of a set of mutually influencing bilateral negotiations (e.g., the logistics agent negotiates individually with the scheduler, or with the resource management agent, etc).
2. *multiple issues* - negotiation ranges over a number of interrelated issues (e.g., price, quantity, quality, date, etc);
3. *repeated rounds (encounters)* - more than one bargaining session may occur before reaching an agreement. So, the atmosphere at the end of one session can influence the atmosphere at the next session;
4. *cooperative or non-cooperative negotiation behavior* - negotiation may occur between agents within the same organization (e.g., between the logistics agent and the scheduler) or between inter-organizational agents (e.g., between the logistics agent and a customer). In the former case, negotiation is cooperative in nature. In the latter case, negotiation is purely competitive;
5. *time restrictions* - time is an important factor. The time needed to reach an agreement must be reasonable. Also, the mutually accepted due dates are often important.

## 3 INTENTIONAL AGENTS - INDIVIDUAL PERSPECTIVE

This section discusses a *single* intentional agent from an internal point of view, in terms of his mental state and planning process.

Let us introduce a formal notation in order to describe the concepts that follow. Let  $Agents = \{ag_1, \dots, ag_r\}$ ,  $r \in N$ , be a set of *intentional agents*. In the remainder of this article, we use  $ag_i$  (or only  $i$ ) to denote the generic intentional agent whose characteristics we are analysing. We use  $k$  and  $m$  to denote particular objects of  $ag_i$  (e.g.,  $g_{ik}$  denote the goal  $k$  of  $ag_i$ , and  $int_{ikm}$  denote the intention  $m$  contained in the plan  $k$  of  $ag_i$ ). We also use the standard notations  $\langle \rangle$ ,  $\{ \}$ , and  $[ ]$  to denote tuples, sets and lists, respectively.

### 3.1 Mental state and mental attitudes

The *mental state*  $M_i$  of agent  $ag_i \in Agents$  is a 5-tuple:

$$M_i = \langle B_i, G_i, PL_i, I_i, Ext_i \rangle$$

where  $B_i$  is a belief set,  $G_i$  a goal set,  $PL_i$  a plan template library,  $I_i$  an intention structure, and  $Ext_i$  an external description (see subsection 4.1).

Every intentional agent  $ag_i$  has a set  $B_i = \{b_{i1}, b_{i2}, \dots\}$  of *beliefs* representing facts about the world and the agent himself. We assume that beliefs are internally consistent, *i.e.*, individual beliefs do not conflict with one another. In addition, we assume that beliefs persist by default over time and are continuously updated to reflect changes in the world. Describing the belief updating process is beyond the scope of this work.

The agent  $ag_i$  has a set  $G_i = \{g_{i1}, g_{i2}, \dots\}$  of *goals* representing world states to be achieved. We consider only achievement goals. An achievement goal, denoted by  $g_{ik}$ , states that  $ag_i$  wants to achieve a world state where  $g_{ik}$  holds. Every achievement goal (goal, for short) has associated a number of factors, including a priority or importance and a degree of urgency. These factors are used by  $ag_i$  to select the most appropriate goal to accomplish. We assume that goals are internally consistent.

The agent  $ag_i$  has a *library*  $PL_i = \{pt_{i1}, pt_{i2}, \dots\}$  of *plan templates* representing known procedures for achieving specific goals. Plan templates can be seen as a subset of the agent's beliefs, namely beliefs about what procedures are useful for achieving which goals under specified conditions. A *plan template*  $pt_{ik} \in PL_i$ ,  $k \in N$ , is a 7-tuple:

$$\langle id, name, args, type, body, constrs, preconds \rangle$$

where  $id$  is a plan identifier,  $name$  is a plan name,  $args$  is a list of arguments,  $type$  is the type of the plan template (composite or primitive),  $body$  is a procedure for achieving the goal specified by the name and args,  $constrs$  is a list of constraints (imposing a temporal order on the members of the body, on the types of arguments, on the relations between arguments, etc), and  $preconds$  is a list of conditions that must hold before processing the plan's body.

The library  $PL_i = CPL_i \cup PPL_i$  has composite and primitive plan templates. A *composite plan template*  $cpt_{ik} \in CPL_i$  has a body containing one or more *body steps* describing the hierarchical decomposition of the goal specified by the name and args into more detailed subgoals. A *primitive plan template*  $ppt_{ik} \in PPL_i$  has a body containing an *action* (or a sequence of actions) directly performable by the agent.

The *intention structure*  $I_i = [it_{i1}, \dots, it_{in}]$ ,  $n \in N$ , consists of a list of intention threads. A single *intention thread*  $it_{ik}$ ,  $1 \leq k \leq n$ , is a conventional agenda of hierarchically related plan templates. More specifically, each intention thread contains the plan  $p_{ik}$  adopted by  $ag_i$  for achieving the goal  $g_{ik} \in G_i$ . This plan is constituted by *intentions* ( $int_{ik1}, int_{ik2}, \dots$ ). We assume that intentions are internally consistent and consistent with the beliefs. It should be stressed that the term "adopted plan" entails an agent's commitment to execute the plan [28]. The nature of this commitment is quite complex [3]. In this article, we just assume that the agent  $ag_i$  commits to the plans he adopts and undertakes to reconsider them when they conflict with the plans of other agents. In particular, he negotiates a mutually acceptable agreement that often leads to plan reconsideration.

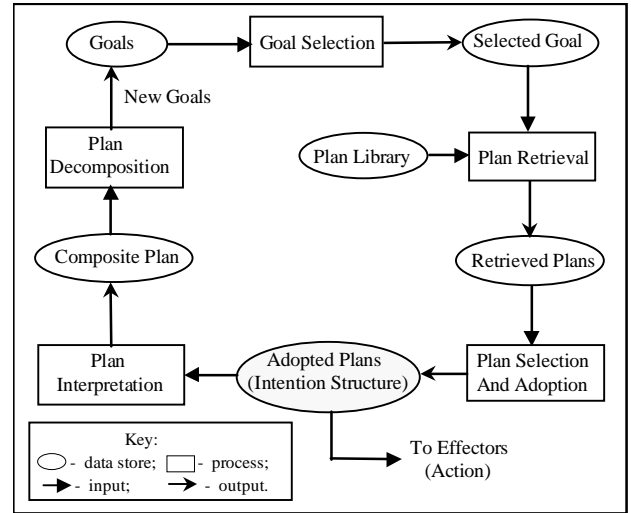


Figure 2. Intentional agent architecture (part related to plan generation)

### 3.2 Plan generation

Figure 2 is a block diagram describing part of the architecture of an intentional agent, namely the part related to plan generation. The mental attitudes just discussed are modeled as data structures and denoted by ovals. The processes that operate on these structures are denoted by rectangles. The belief data structure is not shown, but it provides input to all processes.

A *plan*  $p$  is a partially ordered collection of plan templates. The plan has a *structure*  $Pstruct$  consisting of a hierarchical and temporally constrained And-tree. The nodes of the tree are plan templates retrieved from the library and instantiated with specific values (henceforth, plan templates are just referred as plans). Plan generation is an iterative procedure of: (i) goal selection, (ii) plan retrieval, (iii) plan selection and adoption, (iv) plan interpretation, and (v) plan decomposition.

At any given time, the agent  $ag_i$  selects a goal  $g_{ik} \in G_i$ ,  $k \in N$ , to achieve and starts the generation of the plan  $p_{ik}$ . *Goal selection* is done accordingly to a number of factors, notably the priority and urgency. First,  $ag_i$  prefers to fulfill goals of higher priority over goals of lesser priority. Second, in cases where goals of similar priority are involved,  $ag_i$  prefers to fulfill goals of higher time urgency.

*Plan retrieval* consists of searching the plan library  $PL_i$  and finding any plan whose name and arguments match the representation of  $g_{ik}$ . When suitable matches are found, the plans are retrieved from the library and their arguments are unified with specific values from  $g_{ik}$  representation. These plans are called *retrieved* or *applicable plans*.

*Plan selection and adoption* involves the evaluation of the applicable plans  $(pt_{i1}, \dots, pt_{ik-1}, pt_{ik}, pt_{ik+1}) \in PL_i$ , the selection of the preferred one  $pt_{ik}$ , and the adoption of  $pt_{ik}$ . The evaluation and selection of the applicable plans is done by comparing their costs (scores) and choosing the plan  $pt_{ik}$  with the higher cost [14; 25]. The adoption of  $pt_{ik}$  consists of adding it to the intention structure  $I_i$ . More specifically,  $pt_{ik}$  is added to the top of the intention thread

$it_{ik}$ . In other words,  $pt_{ik}$  is made the root node of  $Pstruct_{ik}$  (structure of plan  $p_{ik}$ ). The description (name and args) of  $pt_{ik}$  is referred to *intention*  $int_{ik1}$  adopted by  $ag_i$ <sup>4</sup>. It should be stressed that the applicable plans  $(pt_{i1}, \dots, pt_{ik-1}, pt_{ik+1}) \in PL_i$  not adopted are also added to  $it_{ik}$  for possible future use (they are explicitly recorded in  $Pstruct_{ik}$  and placed alongside  $pt_{ik}$ ). These plans are called *alternative plans* and have a key role in the definition of a structure for a negotiation problem (see subsection 5.2).

*Plan interpretation* consists mainly of selecting a composite plan for subsequent decomposition. Typically, the plan  $pt_{ik}$  recently added to  $it_{ik}$  is a composite one and is selected. This plan is called the *current plan*.

*Plan decomposition* consists of processing the precondition list of the current plan  $pt_{ik}$ , and establishing a temporal order for its body members  $\{g_{ik1}, g_{ik2}, \dots\}$ . The temporal order is defined by the constraint list. The body members  $\{g_{ik1}, g_{ik2}, \dots\}$  of  $pt_{ik}$  are interpreted as subgoals of  $g_{ik}$  and are added to the goal set  $G_i$  (assuming consistency is maintained).

At this point a new cycle begins and all the tasks described are repeated. The first ordered subgoal  $g_{ik1}$  is selected and processed in the same manner as the initial goal  $g_{ik}$ . The new adopted plan  $pt_{ik1}$  is added to  $it_{ik}$  and placed in  $Pstruct_{ik}$  as the left-most child of the plan  $pt_{ik}$ . This is then repeated for all the other subgoals, with each new adopted plan being located to the right of any plan already placed in  $Pstruct_{ik}$ .

## 4 CONFLICT OF INTERESTS

This section defines formally the concepts of external description and conflict of interests, and describes axioms for conflict detection.

### 4.1 External description

Intentional agents operate in multi-agent environments and must generate their plans by taking into account the potential plans of other agents. They have a data structure called *external description* where the information about the other agents present in the environment is stored. This information is obtained by explicit communication and corresponds to the *beliefs, goals and intentions (plans)* every agent has regarding the other agents. This information may be incomplete and even incorrect.<sup>5</sup>

The external description  $Ext_i$  of an intentional agent  $ag_i \in Agents$  is defined formally as a list:

$$Ext_i = [Ext_i(ag_1), \dots, Ext_i(ag_n)],$$

where  $(ag_1, \dots, ag_n) \in Agents$ . The individual entry  $Ext_i(ag_j)$  stores information about agent  $ag_j \in Agents$  and is defined as a 3-tuple:

$$Ext_i(ag_j) = \langle B_i(ag_j), G_i(ag_j), I_i(ag_j) \rangle$$

where  $B_i(ag_j)$ ,  $G_i(ag_j)$  and  $I_i(ag_j)$  are respectively the beliefs, goals and intentions (plans)  $ag_j$  believes  $ag_j$  has.

### 4.2 Potential conflict of interests

Let  $ag_i \in Agents$  be an intentional agent and  $A = \{ag_1, \dots, ag_n\}$ ,  $A \subseteq Agents$ , be a set of agents that interacts with  $ag_i$ . Let  $I_i$  be the intention structure of  $ag_i$  and  $Ext_i = [Ext_i(ag_1), \dots, Ext_i(ag_n)]$  be his external description. Let  $p_{ik}$  be a plan of  $ag_i$  including intention  $int_{ikm}$ . Let  $PP = \{p_{ik}(ag_1), \dots, p_{ik}(ag_n)\}$  be a set of *possible plans* of the agents in  $A$ , i.e., plans that  $ag_i$  believes these agents have generated. Let  $PI = \{int_{ikm}(ag_1), \dots, int_{ikm}(ag_n)\}$  be a set of *possible intentions* of the agents in  $A$ , i.e., intentions that  $ag_i$  believes these agents have formulated as part of plans  $\{p_{ik}(ag_1), \dots, p_{ik}(ag_n)\}$ , respectively.

Let us assume that the intentions in  $PI$  represent commitments to achieve mutually exclusive world states. In this situation, the intentions are called *potentially incompatible* and represented by:

$$Pot-Incomp(int_{ikm}(ag_1), \dots, int_{ikm}(ag_n))$$

emphasizing the fact that they cannot be executed together. The plans in  $PP$  are also called *potentially incompatible* and represented by:

$$Pot-Incomp(p_{ik}(ag_1), \dots, p_{ik}(ag_n))$$

A *potential multi-agent conflict of interests* from the perspective of  $ag_i$  and with respect to plan  $p_{ik}$  is defined formally as follows:

$$\begin{aligned} Pot-Conf_{ik} = & \exists int_{ikm} \in I_i \wedge \exists int_{ikm}(ag_1) \in Ext_i(ag_1) \\ & \wedge \dots \wedge \exists int_{ikm}(ag_n) \in Ext_i(ag_n) \wedge \\ & Pot-Incomp(int_{ikm}, int_{ikm}(ag_1), \dots, int_{ikm}(ag_n)) \end{aligned}$$

It is important to note that potential conflict is defined as being subjective. In other words, the agent  $ag_i$  only needs to believe that the other agents in  $A$  intend to achieve specific world states, and does not need to know the real intentions of these agents.

### 4.3 Axioms for conflict detection

Conflict detection is done individually by each agent using pre-specified axioms  $Ax_i = \{ax_{i1}, ax_{i2}, \dots\}$ . The axioms have the general form:

$$int_{ikm} \& int_{ikm}(ag_1) \& \dots \& int_{ikm}(ag_n) \& conds \rightarrow false$$

<sup>4</sup> Intentions are goals not yet achieved and considered achievable - goals restricted to the existence of plans for achieving them [9].

<sup>5</sup> We assume throughout this work that the agents are able to communicate by message passing, that they have a common means of structuring their messages, and that they are able to reach a common agreement of the terms they exchange. We also assume that the agents will not necessarily reveal their beliefs, goals and intentions (plans) truthfully, if asked by other agents.

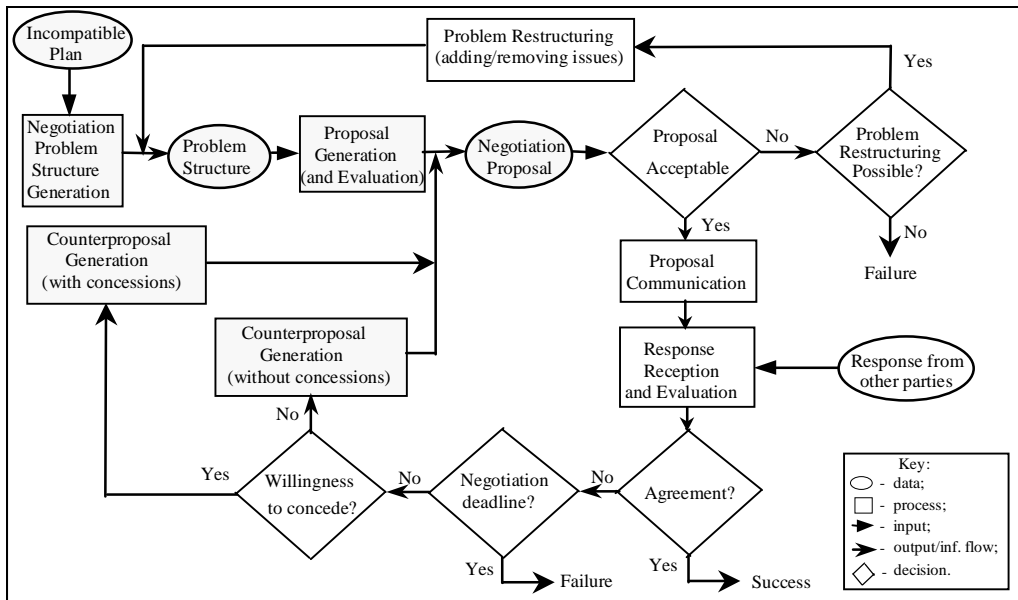


Figure 3. The negotiation mechanism (perspective of each agent)

where  $int_{ikm}$ ,  $int_{ikm}(ag_1)$  and  $int_{ikm}(ag_n)$  have the meaning just specified,  $conds$  is a list of conditions,  $false$  is a 0-ary predicate symbol,  $\&$  is the conjunction operator, and  $\rightarrow$  the implication operator. The axioms can be seen as a subset of the agent's beliefs, namely beliefs about which world states are mutually exclusive under specified conditions.

#### 4.4 True conflict of interests

Intentional agents check regularly their adopted plans in order to detect any potential conflict. Let us assume that  $ag_i$  detects a conflict. Following this, he identifies all the agents  $A = \{ag_1, \dots, ag_n\}$  involved and asks them to confirm the information used to detect the conflict – the possible intentions  $PI = \{int_{ikm}(ag_1), \dots, int_{ikm}(ag_n)\}$ . We assume that the agents are willing to reveal truthfully the correctness of the information necessary to confirm a conflict. However, as stated above, they do not necessarily reveal information not related to a conflict.

The confirmation of potential conflicts of interests leads to *true conflicts of interests*.

### 5 MULTI-AGENT NEGOTIATION

True conflicts of interests raise negotiation problems. Let  $A = \{ag_1, \dots, ag_n\}$ ,  $A \subseteq Agents$ , be a set of intentional agents. Let  $P = \{p_{1k}, \dots, p_{nk}\}$  be a set of plans of the agents in  $A$ . Let  $ag_i \in Agents$  be an intentional agent. Let  $p_{ik}$  be a plan of  $ag_i$  incompatible with the plans in  $P$ . A *negotiation problem* from the perspective of the individual agent  $ag_i$  is defined formally as a 4-tuple:

$$NP_{ik} = \langle ag_i, p_{ik}, A, P \rangle$$

This section presents a generic mechanism for designing and implementing intentional agents with competence for solving negotiation problems. In addition, this section presents a detailed description of four processes specified by the mechanism, namely

negotiation problem structure generation, proposal generation and evaluation, counterproposal generation without concessions, and counterproposal generation with concessions.

#### 5.1 The generic negotiation mechanism

The examination of the negotiation literature from the fields of psychology [29], economy [30], game theory [32], distributed artificial intelligence [21; 27], and computer support of negotiation [17] motivated the development of the mechanism shown schematically in Fig. 3. The mechanism handles *multi-party*, *multiple-issue* and *single or repeated* rounds, and supports the following primary aspects of negotiation:

1. iterative exchange of proposals and counterproposals;
2. application of non-concession and concession mechanisms for proposal generation;
3. learning – discovery of new negotiation issues;

The mechanism defines the essential steps each agent  $ag_i$  will perform in order to solve a negotiation problem  $NP_{ik}$ . First,  $ag_i$  generates a structure  $NPstruct_{ik}$  for the negotiation problem  $NP_{ik}$  (see subsection 5.2). Next, he generates the preferred negotiation proposal that satisfies the requirements imposed by  $NPstruct_{ik}$ . More specifically, he generates multiple alternative negotiation positions, evaluates them using an additive scoring function, and selects the preferred one. The selected position is then used to prepare the preferred negotiation proposal (see subsection 5.3). A *negotiation proposal*, broadly speaking, is a partial or a complete solution to the problem  $NP_{ik}$  that  $ag_i$  faces. Next,  $ag_i$  determines the feasibility of the proposal. If the proposal is acceptable,  $ag_i$  communicates it to the other parties (agents in  $A$ ) which, typically, respond with counterproposals. A *counterproposal* is just a proposal made in response to a previous proposal  $prop_{ikm}$  and more favorable to the responding agent than  $prop_{ikm}$ .

Following this,  $ag_i$  evaluates the counterproposals and either

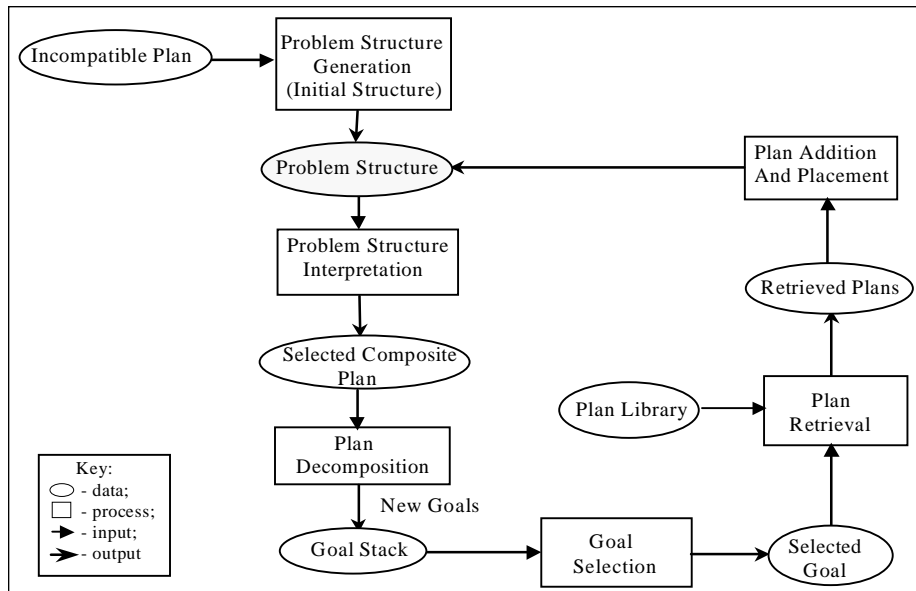


Figure 4. Negotiation problem structure generation

accept one of them or not. If all the counterproposals are unacceptable and the negotiation deadline is not reached,  $ag_i$  responds by sending back a counterproposal. He can generate a counterproposal by either: (i) “improving” the rejected proposal  $prop_{ikm}$  (see subsection 5.4), or (ii) preparing another proposal  $prop_{ikm+1}$  satisfying the requirements imposed by  $NPstruct_{ik}$  (see subsection 5.5). Both types of responses require no changes in  $NPstruct_{ik}$ .

The exchange of proposals and counterproposals continues until an agreement is found, the possibilities for preparing acceptable proposals have been exhausted, or a decision for introducing new issues has been taken. In the latter two cases,  $ag_i$  needs to restructure the negotiation problem in order to avoid a deadlock. Problem restructuring closes one round of negotiation. Negotiation proceeds to a new round in which all the tasks described above are repeated. The final outcome is an agreement or a deadlock.

## 5.2 Negotiation problem representation

The negotiation problem  $NP_{ik}$  defined above is represented by a hierarchical And-Or tree called *negotiation problem structure* ( $NPstruct_{ik}$ ). The description (name and args) of the root node of is called the *negotiation goal* ( $ng_{ik}$ ) and the descriptions of the leaf nodes are called *facts*.

Let  $p_{ik}$  be the plan of  $ag_i$  incompatible with the plans in  $P$ . The negotiation problem structure  $NPstruct_{ik}$  is generated from the structure of plan  $p_{ik}$ , by expanding all the alternative plan templates. More specifically, the generation of  $NPstruct_{ik}$  is an iterative procedure of: (i) problem structure interpretation, (ii) plan decomposition, (iii) goal selection, (iv) plan retrieval, and (v) plan addition and placement. Fig. 4 is a block diagram describing the generation of a structure for a negotiation problem.

First, an initial structure is generated for  $NP_{ik}$ . This structure is simply a copy of  $p_{ik}$ 's structure (And tree). Then the iterative procedure starts.

*Problem structure interpretation* consists mainly of selecting an *alternative plan*  $pt_{ik}$  from the structure of  $NP_{ik}$ . *Plan decomposition*, *goal selection* and *plan retrieval* were described above (see section 3.2). In brief, these processes involve the definition of an order among the body members of  $pt_{ik}$ , the addition of the body members (goals) to a particular goal stack, the selection of a goal, and the retrieval of all the plan templates stored in the library  $PL_i$  matching the goal description.

*Plan addition and placement* consists of placing the retrieved plan templates at appropriate points in the structure of  $NP_{ik}$  (also, according to the procedure described in subsection 3.2). The complete expansion of all alternative plans leads to  $NPstruct_{ik}$ .

## 5.3 Proposal generation and evaluation

Let us introduce some formal definitions needed for describing proposal generation and evaluation. Let  $F_{ik} = \{f_{ik1}, \dots, f_{ikz}\}$ ,  $z \in N$ , be the set of facts of  $NPstruct_{ik}$ . A *fact* is defined formally as a tuple:

$$fact = \langle issue, value \rangle$$

where *issue* is a *negotiation issue* and *value* is a value for the issue. Facts can have associated logical values taken from the set  $L = \{true, false, any\}$ . The set of issues under negotiation is represented by  $Is_{ik} = \{is_{ik1}, \dots, is_{ikz}\}$  and the generic value of the  $m$ th issue by  $x[is_{ikm}]$ ,  $1 \leq m \leq z$ . The issues are defined over finite ranges.

The range of values acceptable to agent  $ag_i$  for the generic issue  $m$  is represented as  $D_{ikm} = [min_{ikm}, max_{ikm}]$ . The agent assigns different weights to each issue reflecting their importance. The *weight* of the  $m$ th issue is represented by  $w_{ikm}$ . We assume the

weights are normalized, i.e.,  $\sum_{m=1}^z w_{ikm} = 1$ .

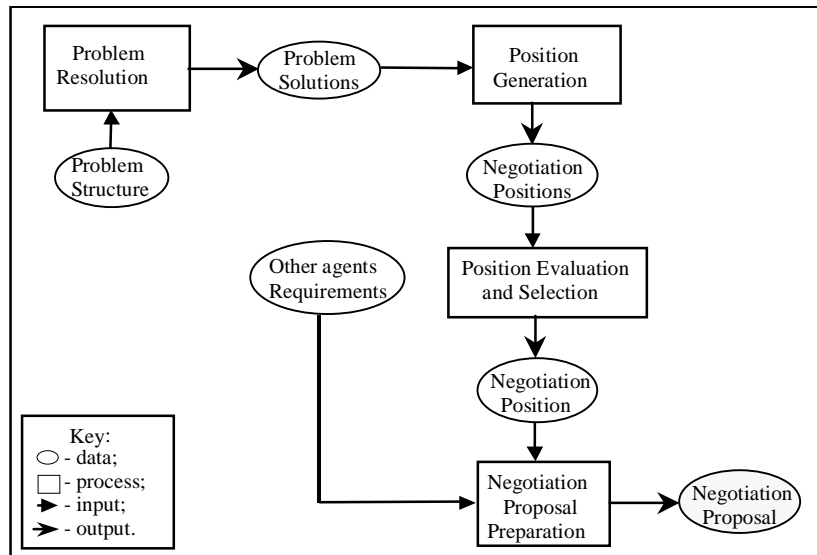


Figure 5. Proposal generation

Proposal generation involves: (i) problem resolution (problem solution generation), (ii) position generation, (iii) position evaluation and selection, and (iv) proposal preparation. Fig. 5 is a block diagram describing the generation of a negotiation proposal.

A negotiation problem  $NP_{ik}$  has usually several solutions. A solution  $S_{ikm} = \{f_{ik1}, \dots, f_{ikp}\}$ ,  $S_{ikm} \subseteq F_{ik}$  is a set of facts with the logical value *true* satisfying the negotiation goal  $ng_{ik}$  [17]. The solution  $S_{ikm}$  is obtained from  $NP_{struct_{ik}}$  by:

1. selecting exactly one disjunct from each Or node;
2. assigning the value *true* to the facts  $(f_{ik1}, \dots, f_{ikp})$  of the tree resulting from step 1.

The facts  $(f_{ik1}, \dots, f_{ikp})$  must have the value *true* in order to achieve  $ng_{ik}$ . The issues  $(is_{ik1}, \dots, is_{ikp})$  associated with these facts represent the *inflexible issues of negotiation*.

A negotiation position  $Pos_{ikm} = \{f_{ik1}, \dots, f_{ikp}, f_{ikp+1}, \dots, f_{ikz}\}$  is a solution  $S_{ikm} = \{f_{ik1}, \dots, f_{ikp}\}$  increased with the remaining facts  $FF_{ikm} = \{f_{ikp+1}, \dots, f_{ikz}\}$  of  $NP_{ik}$ . The facts in  $FF_{ikm}$  are not important for achieving  $ng_{ik}$  and get the logical value *any*. The issues  $(is_{ikp+1}, \dots, is_{ikz})$  associated with these facts are called *bargaining issues* and represent the *flexible issues of negotiation*.

The agent  $ag_i$  generates all the negotiation positions and selects the preferred one using an *additive scoring function* [30]. Let  $V_{ikm}: D_{ikm} \rightarrow \mathbb{R}$  be the *component scoring function* that gives the score (or value)  $ag_i$  assigns to a value  $x[is_{ikm}] \in D_{ikm}$  of issue  $m \in Is_{ik}$ . The score for contract  $x_{ikm} = (x[is_{ik1}], \dots, x[is_{ikp}])$  corresponding to solution  $S_{ikm}$  is given by:

$$V_i(x_{ikm}) = \sum_{j=1}^p w_{ikj} V_{ikj}(x[is_{ikj}])$$

The score for values  $(x[is_{ikp+1}], \dots, x[is_{ikz}])$  of the bargaining issues is assumed to be null. As a result, position  $Pos_{ikm}$  has the

same score as contract  $x_{ikm}$ .

A negotiation proposal  $prop_{ikm} = \{f_{ik1}, \dots, f_{ikp}, f_{ikp+1}, \dots, f_{ikz}\}$  is a revised negotiation position obtained by changing the flexible facts with the value *any* to *true* or *false*. The agent  $ag_i$  selects the position  $pos_{ikm}$  with the higher score and changes the values of the facts in  $FF_{ikm}$  to *true* or *false*. He can adopt a *cooperative* or a *non-cooperative* behavior by assigning values to these facts according to or against the interests of other agents, respectively. He also may wish to maintain all the flexible facts in the proposal or to withhold judgement on some of them. In the latter case, he selects a subset  $CFF_{ikm} = \{f_{ikp+1}, \dots, f_{ikq}\}$ ,  $CFF_{ikm} \subseteq FF_{ikm}$  of the flexible facts and includes only these facts in the proposal  $prop_{ikm} = \{f_{ik1}, \dots, f_{ikp}, f_{ikp+1}, \dots, f_{ikq}\}$ ,  $q \leq z$ . This proposal is called a *negotiation offer*. The facts in  $CFF_{ikm}$  are called *communicated flexible facts* and the remaining facts, i.e., the facts in  $RFF_{ikm} = FF_{ikm} - CFF_{ikm} = \{f_{ikq+1}, \dots, f_{ikz}\}$  are called *retained flexible facts*.

#### 5.4 Counterproposal generation without concessions - bargaining issue manipulation

The agent  $ag_i$  communicates the negotiation proposal (or offer)  $prop_{ikm}$  to the other parties which, in turn, evaluate it and typically respond with counterproposals. Upon receiving the counterproposals,  $ag_i$  may decide either to accept one of them or to send back another counterproposal, i.e., an alternative proposal to  $prop_{ikm}$ .

Counterproposal generation without restructuring the negotiation problem can be done by using: (i) non-concession, and (ii) concession mechanisms. *Non-concession mechanisms* generate counterproposals that meet successively the needs of  $ag_i$  without any significant loss of acceptability. The new counterproposals have a score similar to the score of the previously offered counterproposals. This subsection introduces a significant non-concession

mechanism that we call *bargaining issue manipulation*. Concession mechanisms will be the subject of the next subsection.

Real world negotiations often involves strategic misrepresentations. For instance, bargainers are often advised that they should purposely add to the negotiation agenda issues that they do not really care about, in the hope that the other parties will feel strongly about one or more of these issues - strong enough to be willing to make compensating concessions [30]. To this end, *bargaining issue manipulation* is a mechanism that allows one party to act strategically by exaggerating the importance of some issues (bargaining issues) in order to extract concessions from the other parties. More specifically, this mechanism allows one party  $ag_i$  to "improve" a rejected proposal  $prop_{ikm}$  by selecting one or more bargaining issues with specific values (flexible facts) from  $RFF_{ikm}$  and adding them to the proposal. The "improved" proposal  $prop_{ikm+1}$  and the rejected proposal  $prop_{ikm}$  have the same score (hence,  $ag_i$  does not make a concession).

Bargaining issue manipulation is formalized by two functions: select and add. The function *select*:  $2^{RFF_{ikm}} \rightarrow RFF_{ikm}$  assists  $ag_i$  in selecting a fact from  $RFF_{ikm}$ . This function takes  $RFF_{ikm}$  as input and returns a retained flexible fact  $f_{ikx} \in RFF_{ikm}$ . The function *add*:  $2^{prop_{ikm}} \times RFF_{ikm} \rightarrow 2^{prop_{ikm+1}}$  maps a proposal  $prop_{ikm}$  and a fact  $f_{ikx} \in RFF_{ikm}$  into a new proposal  $prop_{ikm+1}$  containing  $f_{ikx}$ , i.e.,

$$add(prop_{ikm}, f_{ikx}) = prop_{ikm+1} \cdot f_{ikx}$$

where  $\cdot$  stands for concatenation. It should be stressed that several select functions can be defined reflecting the negotiation behavior of  $ag_i$ . One such function is:

$$select(RFF_{ikm}) = f_{ikx} \mid w_{ikx} = \max_{q+1 \leq j \leq z} w_{ikj}$$

where  $w_{ikj}$ ,  $(q+1) \leq j \leq z$ , is the weight of the bargaining issues associated with the retained flexible facts. This function selects the retained flexible fact  $f_{ikx}$  in  $RFF_{ikm}$  corresponding to the bargaining issue with the higher weight or importance.

## 5.5 Counterproposal generation with concessions – negotiation tactics

*Concession mechanisms* generate counterproposals that have successively lower scores than previous counterproposals. Negotiation tactics are the predominant concession mechanisms.

*Negotiation tactics* are functions that compute values for single issues at every instant in the negotiation. Let  $prop_{ikm}$  be a negotiation proposal submitted by  $ag_i$  to the other parties and rejected. Let  $x[is_{ikj}]_{old}$  be the value of the issue  $is_{ikj}$  offered in  $prop_{ikm}$  by  $ag_i$ , and  $D_{ikj}$  be the range of values acceptable to  $ag_i$  for  $is_{ikj}$ . Let  $V_{ikj}$  be the component scoring function of  $ag_i$  for issue  $is_{ikj}$ . This function is assumed to be either monotonically increasing or monotonically decreasing. Let  $prop_{ikm+1}$  be the new

proposal that  $ag_i$  decided to prepare as a response to the rejection of  $prop_{ikm}$ . Generally speaking, a negotiation tactic is a function *tactic*:  $D_{ikj} \rightarrow D_{ikj}$ , that takes the value  $x[is_{ikj}]_{old}$  of  $is_{ikj}$  as input and returns the new value  $x[is_{ikj}]_{new}$  of  $is_{ikj}$  to be offered in  $prop_{ikm+1}$  by  $ag_i$ , with  $V_{ikj}(x[is_{ikj}]_{old}) \geq V_{ikj}(x[is_{ikj}]_{new})$ .

In this work, we consider five generic tactics that model different concessions on a single issue  $is_{ikj}$  at each point of the negotiation process. A *concession* on an issue  $is_{ikj}$  is defined as a change in the value of  $is_{ikj}$  that reduces the level of benefit sought (or score value). The description of the five tactics is as follows:

1. *Stalemate* - models a *null* concession on  $is_{ikj}$ , i.e., the agent  $ag_i$  does not change the value of  $is_{ikj}$ ;
2. *Tough* - models a *small* concession on  $is_{ikj}$ ;
3. *Moderate* - models a *moderate* concession on  $is_{ikj}$ ;
4. *Soft* - models a *large* concession on  $is_{ikj}$ ;
5. *Compromise* - models a *complete* concession on  $is_{ikj}$ , i.e., the new value  $x[is_{ikj}]_{new}$  of  $is_{ikj}$  is the value proposed by another agent for  $is_{ikj}$  (or the reservation value for  $is_{ikj}$  - see below).

Let us introduce some definitions needed to formalize these tactics. The *limit* or *reservation value*  $RV_{ikj}$  for issue  $is_{ikj}$  is the minimum (or maximum) value that  $ag_i$  is willing to accept before he definitely breaks off negotiation - depending on whether  $ag_i$  is maximizing or minimizing  $is_{ikj}$  [30]. The *demand*  $dem_{ikj}$  of  $ag_i$  for  $is_{ikj}$  is the value of  $is_{ikj}$  proposed by  $ag_i$  to the other parties at a specific point in the negotiation. The *negotiation interval*  $NI_{ikj}$  for  $is_{ikj}$  is the range of values between  $RV_{ikj}$  and  $dem_{ikj}$ . Typically,  $NI_{ikj}$  diminishes during the course of negotiation [29, 30]. The new value  $x[is_{ikj}]_{new}$  of  $is_{ikj}$  is computed by the following expression:

$$x[is_{ikj}]_{new} = x[is_{ikj}]_{old} + (-1)^w F |RV_{ikj} - x[is_{ikj}]_{old}|$$

where  $w = 0$  if  $V_{ikj}$  is monotonically decreasing or  $w = 1$  if  $V_{ikj}$  is monotonically increasing, and  $F \in [0, 1]$  is a factor. This expression assures that  $x[is_{ikj}]_{new} \leq RV_{ikj}$ . It also models the following experimental conclusions [29]: (i) a higher limit should produce larger demands and slower concessions, and (ii) demand will be closer to limit the higher the limit. In addition, the expression assures that  $x[is_{ikj}]_{new} \geq x[is_{ikj}]_{old}$ , i.e., once a value is offered for  $is_{ikj}$  it is not reversed. This is a basic principle of negotiation [29, 30] (tradeoffs and problem restructuring are the two notable exceptions - however, these mechanisms are distinct from negotiation tactics).

The factor  $F$  can be simply a constant [19]. The five tactics are then formalized by considering different values for  $F$ . For instance, the stalemate tactic is formalized by setting  $F=0$ , the tough tactic by  $F \in ]0, 0.5[$ , the moderate tactic by setting  $F=0.5$ , the soft tactic by  $F \in ]0.5, 1[$ , and the compromise tactic by  $F=1$  or  $F = |(x[is_{nkj}] - x[is_{ikj}]_{old}) / (RV_{ikj} - x[is_{ikj}]_{old})|$ , where  $x[is_{nkj}]$  is the value proposed by other party  $ag_n$  to the issue  $is_{ikj}$ .

Alternatively, the factor  $F$  can vary throughout the negotiation and be a function of a single variable or criteria [6]. In this article, we concentrate on the *relative concession criteria*. Let  $x[is_{ikj}]_1, x[is_{ikj}]_2, \dots, x[is_{ikj}]_{n-1}, x[is_{ikj}]_n$ , be the values of  $is_{ikj}$  successively offered by  $ag_i$  to the other parties, with  $V_{ikj}(x[is_{ikj}]_{i-1}) \geq V_{ikj}(x[is_{ikj}]_i)$ ,  $1 \leq i \leq n$ . Let  $C = |x[is_{ikj}]_{i-1} - x[is_{ikj}]_i|$ ,  $1 \leq i \leq n$ , be the value of a concession made by  $ag_i$  on  $is_{ikj}$  at a specific point in negotiation. Let  $C_{total} = |x[is_{ikj}]_0 - x[is_{ikj}]_n|$  be the value of the total concession made by  $ag_i$  on  $is_{ikj}$ . We distinguish two functions for modelling  $F$ :

$$(1) F = 1 - \gamma e^{-|C/C_{total}|}, \text{ and } (2) F = \gamma e^{-|1 - C/C_{total}|}$$

where  $\gamma \in R^+$ . These functions model the typical monotone decreasing pattern of concessions - the value of concessions becomes successively smaller as the negotiators move closer to their limit [30]. They also model the typical shape of the demand curve over time - demand declines rapidly at first and then more and more slowly as time goes on [29]. The five negotiation tactics are now formalized by choosing different values for the parameter  $\gamma$ . For function (1), the stalemate tactic is formalized by setting  $\gamma = e^{-|C/C_{total}|}$ , the tough tactic by  $\gamma \in ]1, e^{-|C/C_{total}|}$ , the moderate tactic by setting  $\gamma = 1$ , the soft tactic by  $\gamma \in ]0, 1[$ , and the compromise tactic by  $\gamma = 0$  or  $\gamma = |(RV_{ikj} - x[is_{nkj}]) / (RV_{ikj} - x[is_{ikj}]_{old})| * e^{-|C/C_{total}|}$ .

## 6 RELATED WORK

Negotiation is a rich, multidisciplinary research area. As a result, we highlight in this section just the negotiation work most related to our own work.

Bussmann and Muller [5] present a generic negotiation mechanism. The mechanism is rich, but lacks a rigorous theoretical underpinning and assumes that agents are inherently cooperative. Laasri *et al.* [22] describe a similarly rich mechanism. Again, the agents are assumed to be cooperative and to pursue common goals.

Sycara [34] presents a negotiation mechanism that supports problem restructuring and is based on persuasive argumentation. The mechanism can be employed by non-cooperative agents, but assumes the existence of a centralized mediator. The work of Sycara was extended by Kraus *et al.* [20], who formalized the process of argumentation. However, their work deals only with some particular types of argument and no consideration was given to dynamically introduce new negotiation issues.

Faratin *et al.* [6] present a multi-party, multi-issue, single encounter negotiation mechanism. Again, the mechanism is rich, but no consideration was given to integrate it into a complete agent architecture.

Rosenschein and Zlotkin [32] used game theory to investigate the properties of negotiation mechanisms. Their work has produced significant results, but assumes that agents have complete knowledge of the payoff matrix. Most of the research also assumes a single encounter.

We are interested in negotiation among self-motivated agents. Our structure for representing negotiation problems allows the direct integration of planning and negotiation into a complete

agent architecture. This structure is similar to decision trees [12], and goal representation trees [17], but there are important differences. Our approach does not require the quantitative measures typical of decision analysis. In addition, our approach is based on plan templates and plan expansion, and not on production rules and forward and backward chaining. Also, our formula for modeling negotiation tactics is similar to the formulae used by Faratin *et al.* [6] and Koperczak *et al.* [19]. Again, there are important differences. Our formula assures that the new value of an issue ranges between the reservation value and the previous value of the issue. In addition, our formula models important experimental conclusions about limit, demand, and concession. Finally, the relative concession criteria is not used by other researchers.

## 7 DISCUSSION AND FUTURE WORK

This article has described the development of intentional agents with negotiation competence. In particular, it has introduced a negotiation mechanism, and methods for negotiation problem structuring, proposal generation, and counterproposal generation either with and without making concessions.

There are several features of our work that should be highlighted. First, the negotiation mechanism is generic and can be used in a wide range of domains. Secondly, the mechanism supports problem restructuring, ensuring a high degree of flexibility. Problem restructuring facilitates the removal of deadlocks and increases the parties' willingness to a compromise [34]. Thirdly, the structure of a negotiation problem represents a natural link between the individual and social behavior of agents [10]. Also, the structure defines the set of negotiation issues. In addition, problem structure generation acknowledges the role of conflict as a driving force for negotiation. Finally, proposal generation supports cooperative and non-cooperative negotiation behavior.

Our aim for the future is: (i) to extend the negotiation mechanism to consider proposal feasibility and problem restructuring, and (ii) to validate experimentally the mechanism.

## REFERENCES

- [1] J. Allen, J. Hendler, A. Tate, *Readings in Planning*, Morgan Kaufmann, San Mateo, CA, 1990.
- [2] R. Axelrod, *The Evolution of Cooperation*, Penguin Books, London, UK, 1984.
- [3] M. Bratman, *Intention, Plans, and Practical Reason*, Harvard University Press, Massachusetts, 1987.
- [4] M. Bratman, D. Israel, M. Pollack, Plans and Resource-Bounded Practical Reasoning, *Comp. Intell.*, Vol. 4 (4), pp. 349-355, 1988.
- [5] S. Bussmann, H. Muller, *A Negotiation Framework for Cooperating Agents*, in Proceedings of Coop. Knowl. Based Syst., SIG Workshop, pp. 1-18, University of Keele, UK, 1992.
- [6] P. Faratin, C. Sierra, N. Jennings, Negotiation Decision Functions for Autonomous Agents, *Journal of Robotics and Autonomous Systems*, Vol. 24(3-4), pp. 159-182, 1998.

- [7] M. Fox, J. Chionglo, M. Barbuceanu, *The Integrated Supply Chain management System*, Internal Report, Department of Industrial Engineering, University of Toronto, 1993.
- [8] R. Ganeshan, T. Harrison, *An Introduction to Supply Chain Management*, Internal Report, Department of Management Science and Information, Penn State University, USA, 1995.
- [9] G. Gaspar, H. Coelho, *Where do Intentions Come From?: A Framework for Goals and Intentions Adoption, Derivation and Evolution*, in Progress in Artificial Intelligence, C. Pinto-Ferreira and N. Mamede (eds.), Madeira, Portugal 1995 (LNAI 990).
- [10] L. Gasser, Social Conceptions of Knowledge and Action: DAI Foundations and Open Systems Semantics, *Artif. Intell.*, Vol. 47, pp. 107-138, 1991.
- [11] M. L. Ginsberg, Aproximate Planning, *Artif. Intell.*, Vol. 76(1-2), pp. 89-123, 1995.
- [12] P. Goodwin, G. Wright, *Decision Analysis for Management Judgement*, John Wiley and Sons, 1991.
- [13] A. Haddadi, K. Sundermeyer, *Belief Desire Intention Agent Architectures*, in Foundations of Distributed Artificial Intelligence, G. O'Hare and N. Jennings (eds.), John Wiley, 1996.
- [14] J. Horty, M. Pollack, *Evaluating Options in a Context*, in Proceedings of the 7<sup>th</sup> Conference on Theoretical Aspects of Rationality and Knowledge (TARK-98), Chicago, 1998.
- [15] F. Ingrand, M. Georgeff, A. Rao, An Architecture for Real-Time Reasoning and System Control, *IEEE Expert*, 7(6), pp. 33-44, Dec. 1992.
- [16] N. Jennings, *Cooperation in Industrial Multi-Agent Systems*, World Scientific Publishing Co., Londons, UK, 1994.
- [17] G. Kersten, W. Michalowski, S. Szpakowicz, Z. Koperczak, Restrutable Representations of Negotiation, *Manage. Sci.*, Vol. 37(10), pp. 1269-1290, 1991.
- [18] J. Koehler, Planning from Second Principles, *Artif. Intell.*, Vol. 87, 1996.
- [19] Z. Koperczak, S. Matwin, S. Szpakowicz, Modelling Negotiation Strategies with Two Interacting Expert Systems, *Control and Cybernetics*, Vol. 21(1), pp. 105-130, 1992.
- [20] S. Kraus, K. Sycara, A. Evenchik, Reaching Agreements Through Argumentation: a Logical Model and Implementation, *Artif. Intell.*, Vol. 104, pp. 1-69, 1998.
- [21] S. Lander, *Distributed Search and Conflict Management Among Reusable Heterogeneous Agents*, Ph.D. Thesis, University of Massachusetts Amherst, Computer Science Technical Report 94-32, 1994.
- [22] B. Laasri, H. Laasri, S. Lander, V. Lesser, A Generic Model for Intelligent Negotiation Agents, *Int. J. Intell. Coop. Inf. Syst.*, Vol. 1(1), pp. 291-318, 1992.
- [23] F. Lopes, N. Mamede, H. Coelho, A. Q. Novais, *A Negotiation Model for Intentional Agents*, in Proceedings of the IFAC Workshop on Multi-Agent Systems in Production (MAS'99), P. Kopacek (ed.), pp. 211-216, Vienna, Austria, 1999.
- [24] F. von Martial, *Coordinating Plans of Autonomous Agents*, Springer-Verlag, Berlin 1992 (LNAI 610)
- [25] J. Muller, *The Design of Intelligent Agents*, Springer-Verlag, Berlin 1996 (LNAI 1177).
- [26] J. Muller, *Negotiation Principles*, in Foundations of Distributed Artificial Intelligence, G. O'Hare and N. Jennings (eds.), pp. 211-229, John Wiley, 1996.
- [27] S. Parsons, C. Sierra, N. Jennings, Agents that Reason and Negotiate by Arguing, *Jornal of Logic and Computation*, Vol. 8(3), pp. 261-292, 1998.
- [28] M. Pollack, Overloading Intentions for Efficient Practical Reasoning, *Noûs*, Vol. 25(4), pp. 513-536, 1991.
- [29] D. Pruitt, *Negotiation Behavior*, Academic Press, London 1981.
- [30] H. Raiffa, *The Art and Science of Negotiation*, Harvard University Press, Cambridge, 1982.
- [31] A. Rao, M. Georgeff, *BDI Agents: From Theory to Practice*, in Proceedings of First International Conference on Multi-Agent Systems, V. Lesser(ed.), pp. 312-319, AAAI/MIT Press, 1995.
- [32] J. Rosenschein, G. Zlotkin, *Rules of Encounter*, The MIT Press, Cambridge, USA, 1994.
- [33] Y. Shoham, Agent-Oriented Programming, *Artif. Intell.*, 60, 51-92, 1993.
- [34] K. Sycara, Problem Restructuring in Negotiation, *Manage. Sci.*, Vol. 37 (10), pp. 1248-1268, 1991.
- [35] C. Vidal, M. Goetschalckx, Strategic Production-Distribution Models: A Critical Review with Emphasis on Global Supply Chain Models, *European Journal of Operational Research*, Vol. 98, pp. 1-18, 1997.



## Application and Perspectives in Different Domains

<i>Q. MAIR AND Z. HAAG:</i> AGENT-BASED PROCESS MODEL INTEGRATION IN VIRTUAL SOFTWARE COOPERATIONS .....	75
<i>H. KNUBLAUCH AND T. ROSE:</i> APPLICATION SCENARIOS OF AGENT-BASED INFORMATION LOGISTICS IN CLINICAL AND ENGINEERING DOMAINS .....	85
<i>M. BEETZ, J., A. B. CREMERS, B. HELLINGRATH, C. MAZZOCCO:</i> PERSPECTIVES ON PLAN-BASED MULTIAGENT SYSTEMS FOR DISTRIBUTED SUPPLY CHAIN MANAGEMENT IN THE STEEL INDUSTRY .....	89



# Agent-based Process Model Integration in Virtual Software Corporations

Quentin Mair<sup>1</sup> and Zsolt Haag<sup>2</sup>

**Abstract.** This paper presents initial results for defining a method to integrate process models and supporting tools for the support of software development in Virtual Software Corporations (VSCs). VSCs are emerging as the organisational form enabling the development of complex, large scale software. However, due to the heterogeneous and temporary nature of VSC interactions, no dedicated software engineering environments exist that would support VSC processes. In this paper we present an approach for specifying and reusing the processes of VSC member companies; additionally we use e-mail based agents to integrate distributed process models. The practical applicability of the framework is validated for an experimental VSC. The resulting process model is enacted using a commercial e-mail based workflow management package. NOTE: This work is part of the VISCOUNT Esprit Project (Project Number 25754).

## 1 INTRODUCTION

The environment in which software is being developed has changed dramatically over the past ten years. The changes are a consequence of developments in network technologies and the increasing size of developer teams required for large-scale software projects. Several organisational forms, such as outsourcing and multi-national corporations, are trying to address these environmental changes in order to provide the necessary competitive edge on the market. However, an emerging form, that of Virtual Software Corporations (VSCs), is enabling the development of large scale, complex software while taking full advantage of economies of scale, access to scarce resources and adaptability to market requirements [1]. A VSC is characterised by an often loose and flexible alliance of geographically distributed teams and organisations, with particular specialisms, who come together for a particular, often one-off, software development [2]. Such environments, characterised by a collection of heterogeneous companies, require tool support for process [3] and agent-based distributed software configuration management (SCM). Support is thus required to address two fundamental areas:-

1. Exchange of SCM Configuration Items (CIs) between organisations i.e. an emphasis on *product*

2. Integration and interoperation of existing and new process models brought by organisations forming the VSC i.e. an emphasis on *process*

### 1.1 Software Configuration Management

Software Configuration Management (SCM), “the art of identifying, organizing, and controlling modifications to the software being built by a programming team” [4] has long been a critical and fundamental part of good software development and evolution practices. In recent years, its role and importance, in contributing to the development of high quality software, has been further emphasised by its consideration as a pre-requisite and key process area for the internationally recognised standards for Software Quality Management [5] and Software Process Improvement frameworks of CMM [6] and SPICE [7]. Due to the complexity of software systems, SCM requires automated support in all but the smallest projects. SCM tools have been available for many years now and are comparatively quite mature, handling all of the software process artefacts (not just code versions), with many providing sophisticated process management [8].

### 1.2 Process Modelling

The deployment of process management support tools in a VSC context will have to accommodate the different approaches to specify, model and enact processes in the member companies. The legally independent nature of the member companies and the necessity of applying a less centralised approach in managing processes generate further issues [9]. These aspects define a set of requirements for process models within VSCs and will be discussed in the section 2. The requirements have identified the need for the formal definition of an atomic process element, labelled a VSC Component Process (VCP).

The practical use of the formalism has to be assessed, and thus the processes for a real VSC had to be captured. This was achieved using use case diagrams describing the desired interaction between process actors and a support environment in a software development scenario. The selection of use cases is presented in section 3.

The requirements identified in section 2 are the basis for defining a formal notation to specify processes. The purpose of this notation is to address the modelling requirements and to

<sup>1</sup> Department of Computing, Glasgow Caledonian University, Glasgow, G4 0BA, UK, email: [qma@gcal.ac.uk](mailto:qma@gcal.ac.uk)

<sup>2</sup> Department of Computing, Glasgow Caledonian University, Glasgow, G4 0BA, UK, email: [z.haag@gcal.ac.uk](mailto:z.haag@gcal.ac.uk)

provide a common representation for exchanging processes, process data, CIs and CI meta-data between the different support tools. The formal notation is presented in section 4 and it is built on the process specification language (PSL) [10] and the resource definition framework (RDF) [11]. This section also presents a sample representation in PSL/RDF for use cases.

The formalised use cases are enacted using a commercial email based workflow management package which provides an agent-based interface to a proprietary SCM tool. The approach taken and the results of this experiment are presented in section 5.

The identified requirements, the definition of VCPs, the formal notation for processes and the experiences of the enacted example represent the approach for specifying, reusing and integrating processes in a VSC environment. The final section of the paper presents the conclusions of this work and indicates directions on how to further this research.

## 2 SUPPORTING PROCESSES WITHIN VSCs

VSCs, as defined in [2] are a temporary alliance of independent corporations which use the benefits of economies of scale and access to scarce resources to accommodate the changing needs of a particular market, such as large scale, mission critical software. The temporary and heterogeneous nature of VSCs implies that interaction between partners will have to be supported by tools to a greater extent than in traditional software development [12]. The role of these tools is to provide the context users require to carry out their tasks and to coordinate their activities [13].

It has been suggested that Wide Area Workflow Management can support processes in virtual corporations [9]. However, when considering VSCs, in particular VSCs involved in the development of mission critical systems, there is the added issue of supporting companies at different levels of maturity, as defined in [6], while complying with relevant quality standards.

Therefore, to allow companies to use, to a certain extent, their existing practices and to ensure the quality of the overall development process, support is required both for the *top-down* and *bottom-up* definition of processes. The *top-down* definition caters for critical aspects requiring an overall control, such as compliance with specific quality standards for projects. The *bottom-up* approach is used when integrating processes between companies, for example a configuration audit process can be included in a release manufacturing process, with the processes being performed by different companies.

Supporting both approaches to process definition lends itself to componentisation and we introduce the term of **VSC Component Process** (VCP) to represent an atomic process that is performed within one organisation. The use of VCPs, by encapsulating elementary process definitions, enables the

integration of processes between VSC members and provides support for sharing and reusing processes. This aspect is important in the context of companies using different process support tools. Each VCP is implemented by a **VSC agent**.

However the fact that the VSC contains VCP instances which do not communicate with other VCP instances during internal execution means that synchronisation will take place at defined points i.e. at the start and finish of the VCP instances. Therefore, once VCPs have been defined within a VSC, the interfaces, the interaction between processes has to be defined and controlled. In [9] several forms of control are presented, from these equal partnership is best suited for peer partners in a VSC. In essence this means that interfaces between VCPs are defined and agreed at VSC inception; any changes have to be renegotiated. This joint modelling exercise is similar to the process undertaken in generating the use cases and use case diagrams in defining the requirements for the VISCOUNT project, and will be discussed in a later section.

Another requirement for supporting processes within VSCs is a result of having to accommodate distinct tools used by member companies. VSC partners will bring with them their own software process tools and the need for these applications to inter-operate has become increasingly important. It is now well established that canonical neutral representations are a better way to enable communication than bespoke translators between pairs of tools. Technologies that implement transparent communication at the *syntax level* e.g. HTTP, CORBA are now mature. However communication at the *semantics level* remains a challenge with various initiatives attempting solutions in some areas with various techniques, but none with universal uptake e.g. STEP, CDIF, PIF, Ontolingua.

The Process Specification Language (PSL) [10] at the National Institute of Standards and Technology (NIST) is creating a neutral, standard language for process specification to serve as a semantic glue to integrate multiple process-related applications throughout the manufacturing life cycle. There is nothing to preclude PSL's application to software development processes, but currently this has not been attempted. Section 4 will suggest a possible application of PSL to modelling VCPs.

The following section presents use cases defined by partner companies of the VISOUNT project. The use cases are the basis of defining VCPs and implementing a prototype process support environment.

## 3 USE CASES OF A VSC PROCESS

Use cases defined in this section serve the purpose of identifying VCPs for an existing VSC and formalising the VCPs using PSL.

The VISCOUNT project is defining the interaction between VSC member companies and a VSC support system based on use cases. Use cases are interconnected in UML use case diagrams, each diagram representing a process model functional area. This section builds a scenario from two inter-

linked use case diagrams that would be carried out by two distinct companies.

The use cases and use case diagrams have been defined by partner companies in the VISCOUNT project and represent desired software configuration management (SCM) practices [14]. The use cases were defined for the following functional areas:-

- ?? SCM teamwork, use cases related to culture, environment, switching, communication, security and management
- ?? SCM planning, use cases related to current state analysis, activities, roles, applications, control and standards
- ?? SCM core activities, use cases related to configuration identification, configuration status accounting, release manufacturing and management, change control and configuration audit.

While processes falling in the SCM teamwork and SCM planning category are processes that can be identified within any virtual corporation [3], the SCM core activities are specific for VSCs. This is the reason for selecting use cases and use case diagrams from this category to be modelled and provide support for their enactment.

The first use case diagram defined by a VISCOUNT partner describes the interaction between organisational roles and the support system when a release is prepared and delivered to a customer (**Release Manufacturing and Management Diagram**). Appendix 1 is the UML representation for this use case diagram. The roles involved in this use case diagram could be located within distinct partners of a VSC and could be enacted by one or more persons.

The element that makes this use case diagram important for testing the applicability of our modelling approach is the inclusion of configuration audits, the process for which is defined in the second use case presented in Appendix 2 (**Request QA Approval Use Case Diagram**). Configuration audits are defined based on practices different from the first use case diagram. This has as result a different set of pre- and post-conditions for the use case diagram, and the need for the VSC members to negotiate how best to interface the processes.

The two use cases diagrams considered in this section represent a process fragment with typical issues for a VSC: roles being distributed, interaction between VCPs have to be negotiated, VCPs may be reused by partners other than the partner who defined the VCP. Modelling the resulting VCPs is discussed in the next section.

## 4 MODELLING VSC PROCESSES IN PSL/RDF

This section presents our formalism based on PSL and RDF. This formalism is used for modelling VCPs and to integrate the VCPs into a VSC process model.

### 4.1 PSL

The foundation of PSL is an extendable ontology, which provides rigorous and unambiguous definitions of the concepts necessary for specifying manufacturing processes to enable the exchange of process information. The PSL ontology is represented using the Knowledge Interchange Format (KIF) specification. KIF provides the level of formal rigour necessary to unambiguously define concepts in the ontology, a necessary characteristic to exchange manufacturing process information using the PSL ontology. Although defined in KIF, PSL is decoupled from any syntactical or structural representation, and so equivalent representations for process models can be defined e.g. for XML/RDF discussed below.

The PSL ontology is based upon a small set of primitive concepts: *activity*, *object*, *time point*, and *relationship* which can be extended.

Currently NIST's pilot studies apply PSL to the translation of complete manufacturing process models from one representation to another [15]. We use PSL in a slightly different manner to define VCPs in a form which is independent of any enactment tool. It will also be used to define VSC process models by specifying a network of VCPs and the communication between them. We consider that this work is complementary to the PSL.

### 4.2 PSL/RDF

Resource Description Framework (RDF) is a standard from W3C [10]. Its main aim is to provide interoperability between tools which exchange information over the Web. It does this by defining the general structure by which meta-data can be described, and by providing a mechanism (RDFS [16]) for meta-data schema definition for particular domains.

We perceive four advantages in using RDF to define PSL:-

1. RDF's support for structure in defining meta-data vocabularies whilst retaining understandability
2. RDF uses XML syntax and therefore the existing tool base for processing XML and RDF, including Java based parsers, can be leveraged
3. RDF and XML are text based and easy to transmit and receive in particular for transmitting data and meta-data content between executing VCPs
4. RDF's ability to combine vocabularies in our case in particular SCM meta-data

An initial experiment of representing PSL as RDF/XML has been carried out by NIST [17]. Although this shows the basic ideas it is not based on any defined schema for PSL and is *ad hoc*.

We have defined, which we consider to be more rigorous, a subset of PSL as an RDFS schema. This allows PSL semantics to be properly captured in RDF.

The RDFS schema is based on the PSL Informal Documentation [10]. The documentation defines the semantics of PSL *Kinds* (similar to the idea of a class) and *Relations*. In essence we map the *Kinds* to RDFS Classes and *Relations* to RDF Properties. We do not include any reasoning about the semantics of terms and the relationship between them, just encapsulate the same information. Part of the schema is shown

```

<!-- PSL Core -->

<!-- p5 -->

<rdfs:Class rdf:ID="Activity">
  <rdfs:comment>Activity class</rdfs:comment>
  <rdfs:subClassOf
    rdf:resource="http://www.w3.org/TR/1999/PR-rdf-
    schema-19990303#Resource"/>
</rdfs:Class>

<rdfs:Class rdf:ID="ActivityOccurence">
  <rdfs:comment>Activity class</rdfs:comment>
  <rdfs:subClassOf
    rdf:resource="http://www.w3.org/TR/1999/PR-rdf-
    schema-19990303#Resource"/>
</rdfs:Class>

<rdf:Property ID="occurenceOf">
  <rdfs:comment>occurence of</rdfs:comment>
  <rdfs:range rdf:resource="#Activity"/>
  <rdfs:domain rdf:resource="#ActivityOccurence"/>
</rdf:Property>

<rdf:Property ID="isOccuringAt">
  <rdfs:comment>ocurrence at timepoint</rdfs:comment>
  <rdfs:range
    rdf:resource="http://www.w3.org/datatypes#Number"/>
  <rdfs:domain rdf:resource="#ActivityOccurence"/>
  ...

```

in Figure 1.

Figure 1. Part of PSL/RDF Schema

SCM meta-data can also be modelled in RDF. A RDF schema allows for example *Unapproved* and *Approved* states of a CI to be described and the changes modelled as the process model is enacted (see Figure 4).

### 4.3 Defining a VSC with PSL/RDF

As an example of defining a VSC, we take a subset of the use case requirements defined by the SCM process modelling requirements phase of the VISCOUNT project. This is a *bottom-up* VSC as the requirements have been defined from scratch. As an example we use two use case diagrams and map

each to a VCP: the **Request QA Approval Use Case Diagram** which is initiated in the **Release Manufacturing and Management Diagram** (see Appendix 1 and 2) although the emphasis within our explanation below will be on the former. (This is an interesting example as it embeds one VCP within another, but there is nothing to preclude sequentially connecting VCPs with the appropriate PSL.)

There are five parts to the PSL declaration (full code for the examples can be found at [18]):-

1. **PSL VCP Declaration:** specifies layout of the process by defining a network of activities. From the Request QA Approval Use Case Diagram each use case becomes a PSL *Activity* as illustrated in Figure 3. The control flow sequence is embedded in the layout of the representation (*XorSplit* and *xorSplitSubactivity* are non-elegant but consistent with PSL). *Activity* etc. are defined as belonging to a XML namespace *psl*, in the PSL schema (not shown in diagram). In the *xorSplitSubactivity* relation only one subactivity will be executed. Which one is defined by different state transitions specified in the *ActivityOccurence* (Figure 2).
2. **VSC Process model:** A VSC will normally be a network of VCPs connected together. This is defined in exactly the same formalism as the VCPs above e.g. plugging VCPs together sequentially is done by simply arranging VCP activities sequentially (see below for discussion on pre- and post-conditions). Note that in our examples in this paper we do not show a VSC Process model; rather we have one VCP definition which initiates another VCP; this illustrates the same points and allows us to illustrate synchronous subcontractor-like behaviour.
3. **VSC Instance data:** we need also to model information that is in the use case tables e.g. the actors and roles (locations, e-mail accounts etc.), information for an instantiated VCP and VSC process model (including CI and actor instances) and mapping information.
4. **SCM Configuraton Items:** a schema is defined to indicate the state of the CI as it progresses through the process. This is defined in RDF with the states: *Approved*, *NotApproved* and *BeingApproved*. SCM CIs form the bulk of the data that is transferred both within and between VCP instances. It is therefore important that they are defined in a form which can be transferred between heterogeneous tools. Part of the CI schema is shown in Figure 4 (*Fluent* is defined in the RDFS PSL schema – a *Fluent* is PSL's notion of state). Two views are thus incorporated: the process model showing activity sequences, and state transitions of the CI item. The extendibility of RDF allows extra vocabularies such as these to be easily combined.

5. **Instance data:** as illustrated in Figure 2 this binds the VCP to the VSC Instance data and CI items. *ActivityOccurrences* are used in PSL to denote instances of activities i.e. each *ActivityOccurrence* is in turn bound to a particular actor as defined in the VSC instance data. We have added the ability to reason about pre- and post-conditions (currently on single CIs) by adding the notion of *requires* and *provides*. This enables the facility of interconnecting opaque VCPs by negotiating their interfaces. In addition we can model state (*Fluent*) changes e.g. using *stateAfter* and determine alternative process execution paths, denoted by *xorSplitSubactivity* subactivities, based on state values. An abridged version of this instance data is shown Figure 2, where *qa* is a XML namespace for the PSL VCP declaration shown in Figure 3. *Lifespan1* is a *pseudo* PM actor for the LIFESPAN SCM tool we currently use [19] (*pseudo* as the SCM tool and the process modelling functionality are here together treated as one functional unit; but are in fact in the experiment described below two software entities).

```

ActivityOccurrence rdf:ID="A_FAU_UC3">
<occurrenceOf>qa:A_FAU_UC3</occurrenceOf>
<boundTo>#John</boundTo>
<requires>#ci1</requires>
<stateBefore>ci:BeingApproved</stateBefore>
<provides>#ci1</provides>
<stateAfter>ci:Approved</stateAfter>
</ActivityOccurrence>

<ActivityOccurrence rdf:ID="A_FAU_UC4">
<occurrenceOf>qa:A_FAU_UC4</occurrenceOf>
<boundTo>#John</boundTo>
<requires>#ci1</requires>
<stateBefore>ci:BeingApproved</stateBefore>
<provides>#ci1</provides>
<stateAfter>ci:Unapproved</stateAfter>
</ActivityOccurrence>

<ActivityOccurrence rdf:ID="A_FAU_SUC3">
<occurrenceOf>qa:A_FAU_SUC3</occurrenceOf>
<boundTo>#Lifespan1</boundTo>
<requires>#ci1</requires>
<provides>#ci1</provides>

```

Figure 2. Partial VCP Instance Data

Figure 3: PSL/RDF VCP Description

#### 4.4 Mapping PSL/RDF to Enactable Process Models

A VSC process model consists of a network of VCPs (or equivalently as in our example), definitions of the VCPs

```

<Activity rdf:ID="QA_Request_VCP">
<name>Configuration Audit VCP</name>
<subactivity>
<Activity rdf:ID="A_FAU_UC1">
<name>Request of QA Approval</name>
</Activity>
<Activity rdf:ID="A_FAU_SUC1">
<name>Log QA Request with SCM</name>
</Activity>
<Activity rdf:ID="A_FAU_SUC2">
<name>Automatic Notification</name>
</Activity>
<XorSplit rdf:ID="A_FAU_UC2">
<name>Build Inspection</name>
<xorSplitSubactivity>
<XorSplit rdf:ID="A_PAU_UC1">
<name>Formal Test as InspApp</name>
<xorSplitSubactivity>
<Activity rdf:ID="A_FAU_UC3">
<name>Grant QA Approval</name>
</Activity>
<Activity rdf:ID="NULL">
<name>Null Activity</name>
</Activity>
</xorSplitSubactivity>
</XorSplit>
<Activity rdf:ID="A_FAU_UC4">
<name>QA Unapproved</name>
</Activity>
</xorSplitSubactivity>
</XorSplit>
<Activity rdf:ID="A_FAU_SUC3">
<name>Log QA Approval with SCM</name>
</Activity>
<Activity rdf:ID="A_FAU_SUC4">
<name>Automatic Notification</name>
</Activity>
</subactivity>
</Activity>

```

themselves and various aspects of VSC instance data. These descriptions are not in a form which is directly enactable. As illustrated in Figure 5 two transformations are necessary to this representation to obtain an enactable model.

The first transformation maps the VCPs to a set of VCP instances. The VSC process model is also instantiated using the same VSC instance data to describe communication between the VCP instances (although this paper does not discuss this directly). Information about the communication mechanism is also required i.e. *send* or *share* model. The first transformation therefore results in a set of partitioned VCP instances bound to actor instances.

```

<rdfs:Class ID="ApprovedState">
<rdfs:subClassOf
psl:EnactmentClass="http://bankfoot.gcal.ac.uk/psl/psl01#
Fluent"/>
</rdfs:Class>

<ApprovedState ID="Unapproved"/>
<ApprovedState ID="BeingApproved"/>
<ApprovedState ID="Approved"/>

```

Figure 4. Partial CI Schema

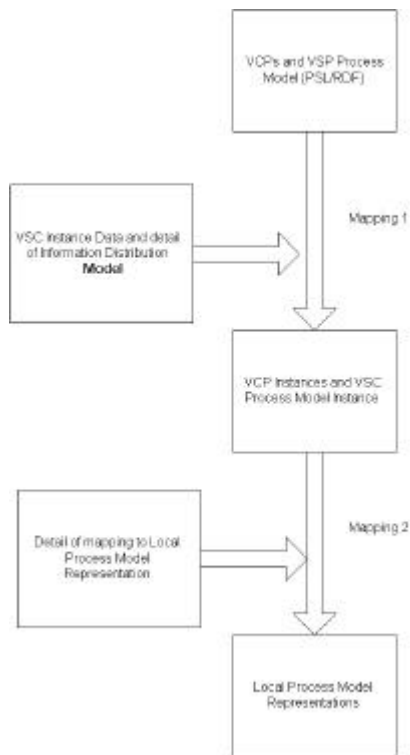


Figure 5. Mapping definitions for enactment

```

<Node rdf:ID="A_FAU_SUC2">
  <name>Automatic Notification</name>
  <actor>common:Lifespan1</actor>
  <Message rdf:ID="A_FAU_SUC2">
    <to>common:John</to>
    <attaches>common:ci1</attaches>
    <state>ci:BeingApproved</state>
    <subject>Automatic Notification</subject>
  </Message>
</Node>

<Node rdf:ID="A_FAU_UC2">
  <name>Build Inspection</name>
  <actor>common:Lifespan1</actor>
  <subject>Build Inspection</subject>
  <rdfs:Alt rdf:ID="XOR">
    <Message rdf:A_PAU_UC1">
      <to>common:John</to>
      <attaches>common:ci1</attaches>
      <condition>Formal Test as Inspection
Approved</condition>
    </Message>
    <Message rdf:ID="A_FAU_UC4">
      <to>common:John</to>
      <attaches>common:ci1</attaches>
      <condition>QA Unapproved</condition>
    </Message>
  </rdfs:Alt>

```

Figure 6. Partial Result of First Mapping

In terms of our results presented below, which use an *send* model e-mail based workflow tool, the output generated is in terms of a generic node and arc based representation. As well as specifying the control flow it also specifies the information flows, described as messages between the actor instances in terms of an e-mail interface e.g. attachments and subject fields. Conditional flows are also modelled. An edited example of a VCP instance for the Request QA Approval VCP and VCP instance data previously discussed is shown in Figure 6. The *actor* field in this case represents the actor which enacts the VCP instance; in this case it is *Lifespan1*. The specification determines next node in the sequence, message specifications and conditional control. The conditional control can be implemented by sending e-mail questions to the other actors. This form is suitable as a specification for designer use and is used as a template for workflow design CAD tools. We anticipate that the existing tool support for XML and RDF can be applied to creating automated translation tools for this transformation. Currently it is undertaken by hand by applying a set of rules. These rules are yet to be documented.

The second transformation involves taking the first output generated and mapping it to specific process model or workflow representations. This is suitable for tools requiring language-type workflow descriptions.

## 5 A PROTOTYPE IMPLEMENTATION

As validation of our work we have implemented a partial VSC to support instantiation of the case study supported by the two use case diagrams. The implementation has the following components and topology:-

1. We use two instances of GFI's Emailflow [20], each implementing a **VCP agent**. Each VCP agent simulates the execution of SCM process internal to a separate VSC company. We define this architectural viewpoint as a **VSC domain**. Note that this means personnel from either company can participate in the process of either VCP instance; but that the enacted VSC process is partitioned. This naturally fits the working practices of temporary VSC alliances. The primary reason for choosing Emailflow was its provision for interfacing with other tools, thus providing an agent-type interface. The definition of workflows follows a node-arc model albeit not based on any formalism. Currently meta-data information is communicated in e-mail messages as name-value pairs (which Emailflow provides a simple mechanism for parsing). We anticipate that this will be restructured as equivalent XML in a second prototype. Workflows can be started by sending e-mails which facilitates VCP instances starting others. An example of the workflow development tool showing the design of the workflow for the Request QA Approval VCP is shown in Figure 7.
2. Each Emailflow instance is "back-ended" into an instance of BAE's LIFESPAN [19]. LIFESPAN is an industrial strength SCM tool with support for ISO 9001 and other quality standards. Although Lifespan comes with non-distributed process model support, it does not address issues such as process interoperability, and we use Lifespan only as a CI and CI meta-data repository. The meta-data includes roles, modules and packages, a design control mechanism, and reporting mechanism. It is the intention that, within the context of the VISCOUNT project, this work is viewed as an feasibility study for future versions of Lifespan with improved process support. Lifespan supports Microsoft Windows and Motif user interfaces, a C++ API and client UNIX and NT command line interfaces.

The interface between the VCP agent and Lifespan is bespoke for particular VCP descriptions. This means that we have specific interfaces for system use cases and not a generalised API. Currently our platform uses Emailflow on two NT boxes with Lifespan also on one of the NT boxes and also on HP-UX. Emailflow can maintain run-time data in a number of different repositories - we use Microsoft Access. The interface between Emailflow and Lifespan involves scripts written VB-Script,

which allows manipulation of the run-time data. This synchronously calls a batch file on NT to manipulate the Lifespan client command line interface, passing data both ways as necessary.

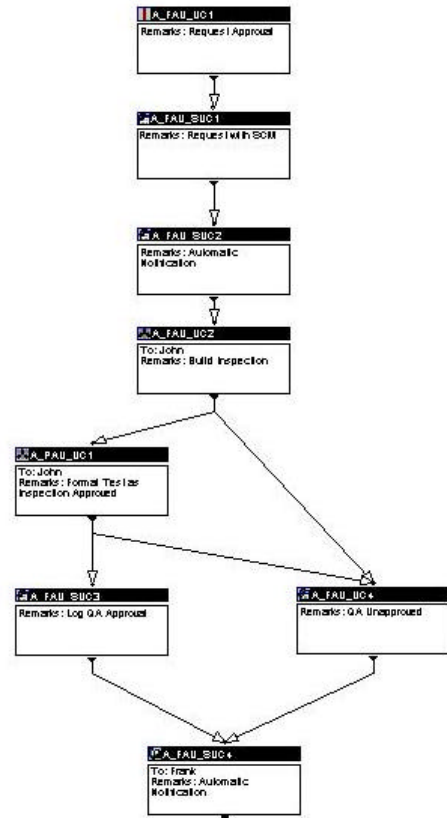


Figure 7. VCP Agent Implementation of Request QA Approval VCP

## 6 RESULTS, CONCLUSIONS AND FURTHER WORK

So far we have successfully demonstrated our approach of defining a specification of communicating VCP agents from user requirements. However we have only a partial solution; it is deficient in the following general areas:-

1. homogeneity: we require a second type of process modelling tool to validate heterogeneous communication.

2. lack of an industrial case-study: a more robust field trial in industry is required.

Our intention is that our next efforts will be aimed at proving a more general VCP agent solution covering the above points.

Specific areas we would like to address in future are:-

1. multiple CIs: so far we pass only the contents of single CIs; we wish to extend this to a more general case especially for pre- and post-conditions.
2. structured meta-data: replace name-value pairs with XML/RDF.
3. a comparative mapping and experiment to support interaction based on a *share* model.

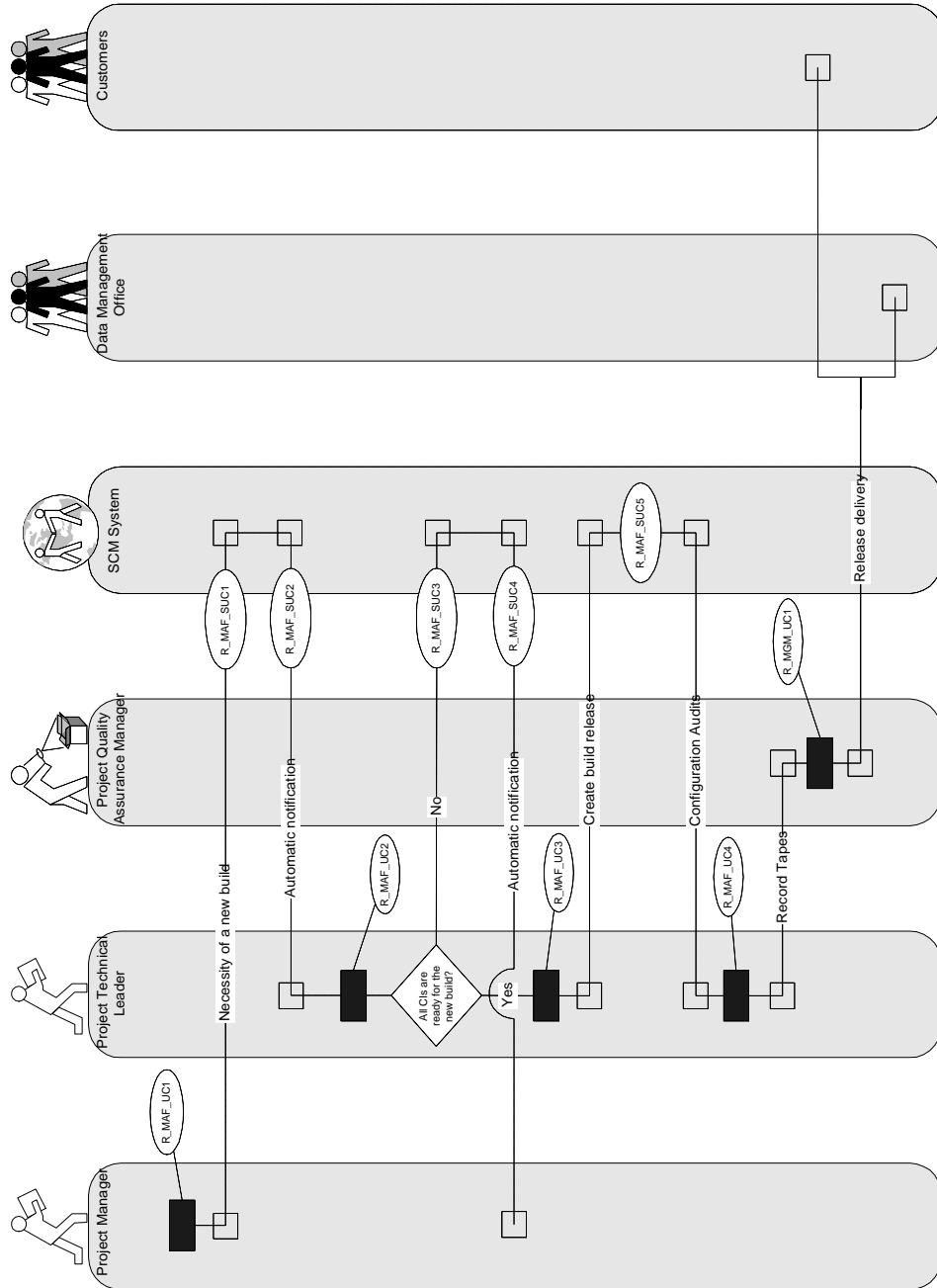
More generally and in the longer term, we anticipate work investigating:-

1. heterogeneous SCM tool integration through VCP agents: translation of SCM CI and meta-data perhaps incorporating an ontology for SCM.
2. development of a methodology for negotiating the set-up of VSCs. We anticipate that this would focus on top-down agreement of pre- and post-conditions, perhaps with the addition of agent-type interfaces.
3. iterative process improvement whereby partner VCPs can be measured and improved whilst still participating in the VSC.
4. automatic assessment to determine if specified VCPs meet designated international standards e.g. ISO 9001. Potentially also a library of VCP specifications which meet these standards and can be reused.
5. VSC topology models i.e. to assess the impact of subcontractor or peer-to-peer VSC domains on process models.

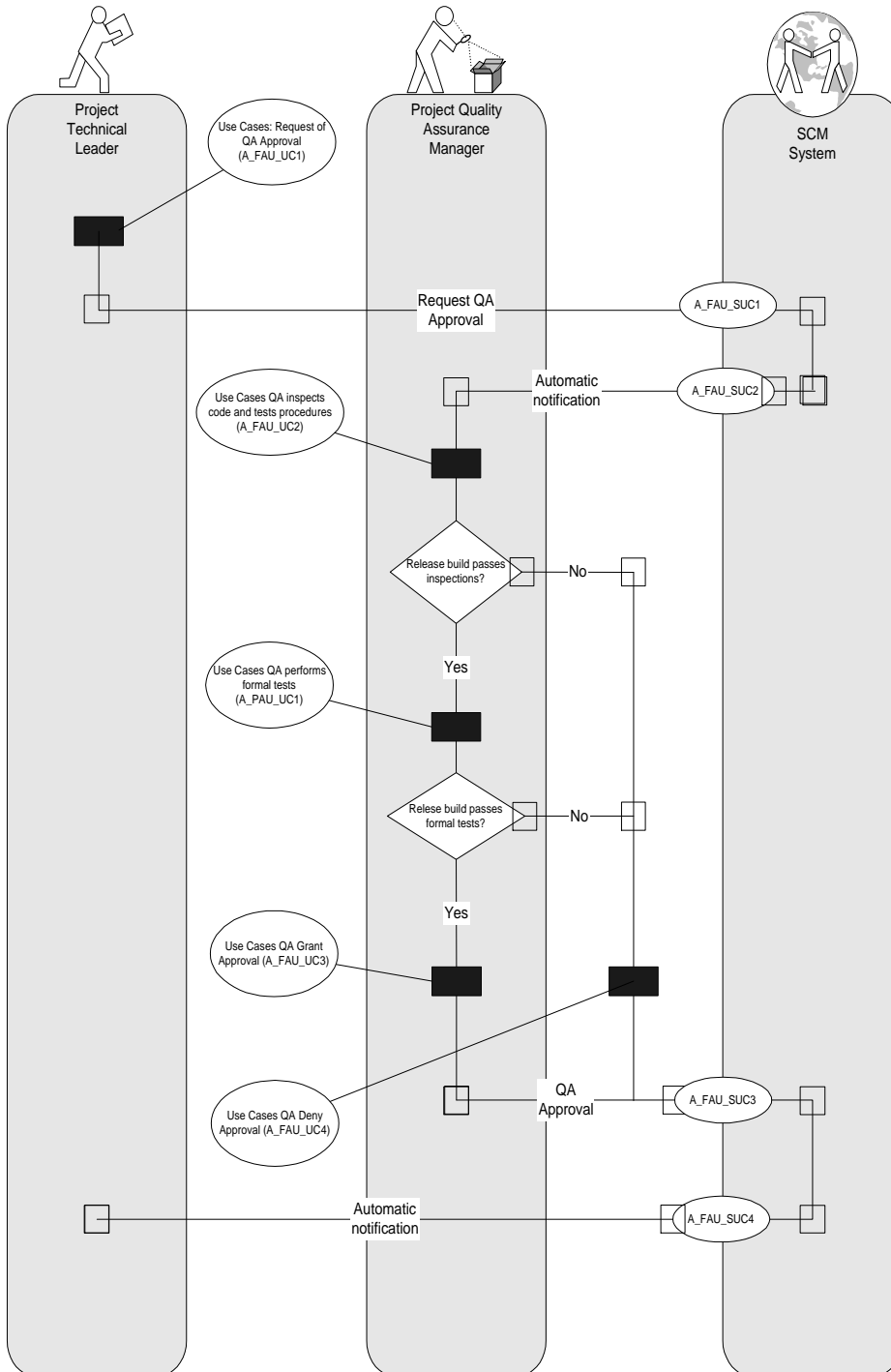
## 7 REFERENCES

- [1] Boldyreff, C., Newman, J. & Taramaa J. (1996): Managing Process Improvement in Virtual Software Corporations, Proceedings, IEEE 5<sup>th</sup> Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE '96), June 19-21, 1996, Stanford University Californian, USA
- [2] Byrne, J. A. (1993), The virtual corporation, Business Week (BWE), Issue: 3304, date: Feb 8, 1993, p98-102.
- [3] Hardwick, M., Bolton, R.: The Industrial Virtual Enterprise, in: Communications of the ACM, 40 (1997) 9, p59-60
- [4] Software Configuration Management; Addison-Wesley, 1986
- [5] Notes for Guidance on the Application of the ISO9001 Standard to Software Development, ISO, 1991
- [6] Mark C. Paulk - The Evolution of the SEI's Capability Maturity Model for Software - Software process - Improvement and practice 1995 Pilot Issue p3-15
- [7] Rout T.P.; SPICE A Framework for Software Process Assessment, Software Process Improvement and Practice (pilot issue); pp57-66, 1995
- [8] Mosley V., Brewer F., Heacock R., Johnson P, LaBarre G., Mazz V. & Smith T.; Software Configuration Management Tools: Getting Bigger, Better, and Bolder; Crosstalk, January 1996, pp6-11
- [9] Riemp, G. (1998): Wide Area Workflow Management: Creating Partnership for the 21<sup>st</sup> Century. Springer
- [10] PSL Informal Documentation Version 0.1; <http://www.mel.nist.gov/psl/>
- [11] Resource Description Framework Model and Syntax Specification Recommendation; <http://www.w3.org/TR/1999/REC-rdf-syntax-19990222> (1999)
- [12] [Zs. Haag, R. Foley, J. Newman - Software Process Improvement in Geographically Distributed Software Engineering: An Initial Evaluation - Proceedings of The 23<sup>rd</sup> Euromicro Conference, Budapest September 1997, Hungary, IEEE-CS Press
- [13] Davidow, W.H., Malone, M.S. 1992. *The Virtual Corporation: Structuring and Revitalizing the Corporation for the 21st Century*. Harper Business, New York
- [14] Second Revision of User Requirements. Esprit project number: 25754 (1999)
- [15] Polyak, S. T., Aitken, S.; Manufacturing Process Interoperability Scenario; AIAI-PR-68; University of Edinburgh (1998)
- [16] [Resource Description Framework Schema Specification Recommendation; <http://www.w3.org/TR/1999/REC-rdf-syntax-199903030> (1999)
- [17] Lubell, J.; Representing PSL as XML; <http://www.mel.nist.gov/psl/xml/> (1999)
- [18] Mair, Q; <http://bankfoot.gcal.ac.uk/psl/> (1999)
- [19] Lifespan Technical Overview, BAeSEMA Ltd., +44 181 942 9661 (1999)
- [20] Emailflow, <http://www.gficomms.com/> (1999)

## 8 APPENDIX 1: RELEASE MANUFACTURING AND MANAGEMENT USE CASE DIAGRAM



## 9 APPENDIX 2: REQUEST QA APPROVAL USE CASE DIAGRAM



# Application Scenarios of Agent-Based Information Logistics in Clinical and Engineering Domains

Holger Knublauch and Thomas Rose<sup>1</sup>

**Abstract.** Coordinated information flow and situated information provision surface as challenging requirements for many processes that operate in distributed environments. The distribution of processes may be founded in organizational or spatial distribution and in the separation of concerns, knowledge, information and data repositories. In this paper, we are focusing on two application domains that show typical requirements for information logistics: (a) Clinical resource and patient management, and (b) engineering processes in simultaneous engineering teams in the automotive domain. We argue that a multi-agent approach, providing methodologies for autonomous, situated, social and pro-active information services, is a well-suited platform for the coordination of information flow and pro-active information delivery in these domains.

## 1 Introduction

Economic pressure and the rapidly growing amount of knowledge have led to a situation where more and more tasks need to be distributed among specialized, cooperating agents. *Cooperation* can be defined as *communication* between two or more participants with the purpose of work carried out on artifacts [1]. Thus, the communication among distributed agents is a key factor in efficient processes. The effectiveness of communication depends on *coordination* and thus on an efficient logistics of information. As a result, the demand for automation and computerization of this information logistics task is increasing.

In this position paper, we describe two application domains that show a potential for optimization by means of computerized information logistics: (a) Clinical resource and patient management, and (b) engineering processes in simultaneous engineering teams in the automotive domain. The processes in these domains share various attributes, because they both operate in networks of organizationally or spatially distributed units and rely on a separation of concerns, knowledge, information and data repositories. We show that multi-agent technology, suggesting the development of situated, autonomous and flexible software components, can serve as a suitable platform for supporting the task of information logistics.

This paper is organized as follows. In the following section, we describe the attributes of the processes in the two application scenarios and point at their common coordination requirements. In section 3, we show the potential benefits of utilizing agent technology by presenting information logistics tasks that can be solved by agents. Finally, in section 4, we briefly give some conclusions and discuss some of the open issues that will be dealt with in our future work.

---

<sup>1</sup> Research Institute for Applied Knowledge Processing (FAW), Helmholtzstr. 16, 89081 Ulm, Germany, email: {Holger.Knublauch|Thomas.Rose}@faw.uni-ulm.de

## 2 Coordination requirements in organizational networks

### 2.1 Clinical resource and patient management

Modern hospitals are highly distributed environments. Patient health care is performed by a network of various clinical units, including administration, surgery, anesthesiology, laboratories, radiology, intensive-care units, wards, storage and others. These units are not only spatially distributed, but also in terms of concerns and organization. For example, whereas the laboratories mostly act as a mere service provider responsible for the restricted task of analyzing blood samples, anesthetists need to make decisions that can have a critical impact on the patient's health. Furthermore, these decisions depend on a broad range of information about the patient's history, the surgical procedure, test results, the available resources and other aspects. Due to this complexity, anesthesia is a good example for studying the coordination requirements in distributed environments.

An overview of the clinical personnel and the information flow between them is shown in figure 1. The task of operating theatre management, i.e. assigning patients, anesthetists, nurses and material to the available operating rooms, is performed by an attending anesthetist. In the following, we describe a typical scenario from theatre management. The scenario is a result of our work aiming at the development of a multi-agent system for information management in anesthesia [5].

In this scenario, each of the surgical departments has a fixed amount of operating room time per day. Each room is assigned a list of patients by the attending surgeon. This list is given to the attending anesthetist (AA), who sends out available anesthetists for pre-operative visitations to the wards. These visitations result in initial patient information sheets. Based on this information and the available resources (personnel, theatre occupation), the AA either accepts the proposed operation time or notifies the attending surgeon about potential difficulties. After the two managed to negotiate an operation time, the AA assigns the case to one of the suitable anesthetists. The latter is responsible for having the operating theatre prepared with material and drugs.

When the planned operating theatre is about to become available, the patient is requested from the ward and delivered by a nurse. Then, the patient is brought to the induction room, where he is anesthetized. After he is prepared for surgery, the patient is moved to the operating theatre and finally to the recovery room. A successful surgical operation ends with the patient being returned to the ward. The task of the AA is to supervise the ongoing surgical procedures and to offer help where needed. For theatre management, he also has to assess the remaining operation times. Before and even during the operations, the personnel can access services from other units, such as the laboratory

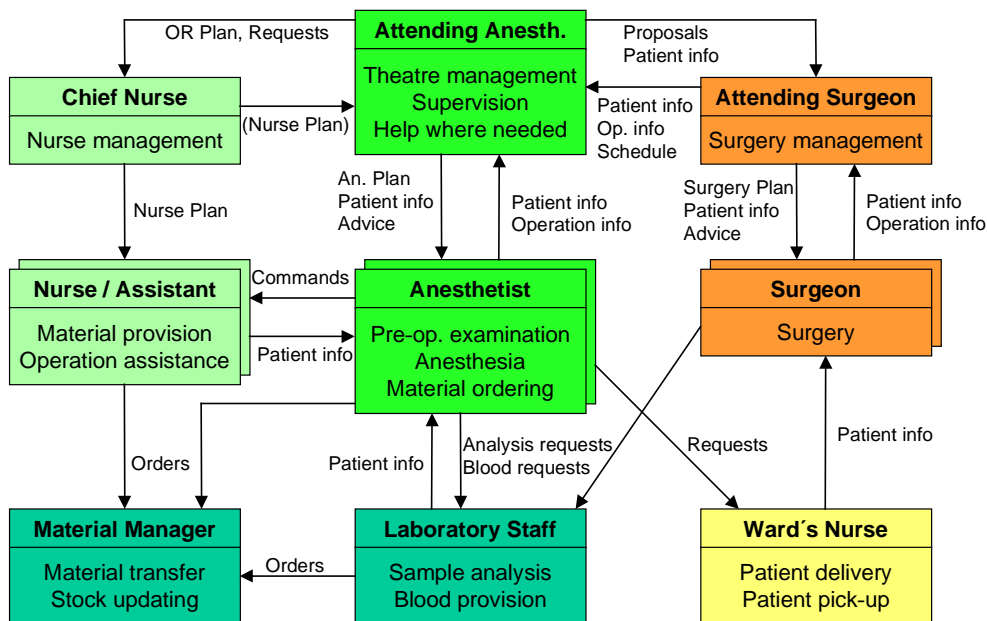


Figure 1. Some of the human agents involved in the context of anesthesia, their tasks and the information flow between them.

and the radiological department. The laboratory processes a queue of blood samples for analysis or is asked to deliver blood conserves. In emergency cases, the lab is expected to modify its task schedule accordingly. Emergency patients also require the AA to quickly shift the use of the available resources, e.g. by postponing planned operations.

## 2.2 Distributed engineering

The competitive market conditions in the automotive sector have put more weight on the development departments to constantly rethink their strategies and apply new methods for improving both their products and business processes. The question arises of how to optimize engineering processes in the early phases of product development. In order to improve engineering performance, Simultaneous Engineering (SE) was adopted by many enterprises in the automotive industry as an organizational concept for product development. SE is defined as “a systematic approach to the integrated, concurrent design of products and their related processes, including manufacture and support. This approach is intended to cause the developers, from the outset, to consider all elements of the product life cycle from conception, through disposal, including quality, cost, schedule, and user requirements” [8]. SE teams have been established to implement simultaneous engineering. These teams comprise members of the various functional departments and representatives from suppliers. Due to these and other developments, cooperative work in teams is the most dominant work organization in product development [7].

As an operational consequence of these kinds of distribution, SE teams are often confronted with incomplete information due to the state of the process, the type of coordination or other environmental reasons. Efficient information logistics is instrumental to support SE teams in their operations. Lack of task-specific information surface as a major obstacle to the success of such SE teams. This lack might be caused by several reasons:

- The available documents containing task-specific information are outdated.
- Information is incomplete due to open decisions.
- Various distributed information sources are not synchronized, i.e. information is already available at one department, but has not been transmitted to others.

## 2.3 Common coordination requirements

The tasks of clinical theatre management and simultaneous engineering share a number of common attributes:

- *Distributed*: Tasks, concerns, organization and know-how are distributed among various human agents and departments. For example, each of the clinical units has to deal with only certain aspects of the patient, so that data and information are also distributed. Patient data as well as design and requirements documents from engineering processes are distributed across various types of media, such as fax messages, data sheets, oral communication, or (incompatible) computerized information systems.
- *Parallel*: The clinical tasks as well as the engineering tasks run concurrently. Furthermore, these tasks operate on a shared subject, i.e. the patient or the engineering artifacts, respectively.
- *Non-deterministic*: Since patients are complex and little-understood biological systems, the prediction of the outcome of surgical procedures is difficult. Unexpected emergency cases happen, as do incidents in anesthesia. Similarly, the outcome of engineering and design processes is hard to predict.
- *Self-organizing*: Responding to the uncertainty, the clinical and engineering staffs have to be able to react in a flexible manner. Although there is a strict hierarchy in terms of commands, for example between attending and other anesthetists and between group leaders and assistants, the personnel acts to a certain extent self

dependent. Many decisions in theatre management are negotiable, since resources are limited.

- *Communication-intensive*: The properties mentioned above demand for sophisticated and efficient communication paths among the personnel. Commands, requests, intentions, schedules, patient information, emergency calls, new design documents and constraints have to be delivered to the right person at the right time.

Most of these aspects deal with information flow and processing between (human) agents in a distributed environment. In the following section, we will show that the multi-agent paradigm provides a very suitable platform to support this information flow.

### 3 Agent-based information logistics

In most of the various definitions of agenthood that can be found in the literature, the following attributes are assigned to software agents (cf. [3]):

- *Situatedness*: The agent receives sensory input from its environment and can perform actions which change the environment.
- *Autonomy*: The agent is able to act without direct intervention of humans.
- *Flexibility*: The agent is able to respond in a timely fashion (responsive), to take the initiative (pro-active) and to interact with other human or artificial agents (social).

These properties correspond to the needs of information management identified in section 2. The software agents we propose are situated in the *distributed* clinical and engineering networks. Each of the various *parallel* processes taking place in this environment can be supported by one or more autonomous agents. These agents are to a certain extent *self-organizing* and therefore require fluent communication pathways to interact with each other and with the human agents they act on behalf of. They need to respond to the *non-deterministic* processes in a flexible and timely fashion.

The following subsections present tasks that can be supported by means of agents.

#### 3.1 Agents in clinical resource and patient management

The strength of computers is to quickly process large amounts of data. For legal and other reasons, clinical decision-making must remain with human agents. Thus, the main task of software agents supporting human decision making by relieving from the burdens of data overload and those tasks that require little intellectual background. We envision small hand-held devices carried by each of the anesthesiologists, nurses, surgeons and other mobile personnel. Depending on the context, the configurable interface agents running on these devices supply the respective user with information such as patient data and the current theatre management plan. Comparable to the pager devices already present in modern hospitals, the devices might proactively notify its wearers about important incoming information via acoustic signals. In particular, we have identified the following tasks for the agents in our clinical setting:

- For the attending anesthetist: Managing the operation schedule (rooms, time, surgical personnel), coping with high-priority cases, such as emergency patients. Proposing suitable anesthesia personnel for the scheduled operations, assisting the attending anesthetist

in resource management. Supervising the current surgical procedures by condensing information from the current operations, so that the attending anesthetist is able to quickly assess the future development (e.g. the remaining operation time).

- For the anesthetists: Pro-actively notifying each anesthetist on plan changes and the relevant patient data. Collecting and analyzing the patient data measured by the clinical devices, providing an overview of the patient's history, state and location.
- For the nurses: Requesting that anesthetic equipment and drugs are made available for each surgery and ensuring that the operating room is in the desired state (cleaned, heated, etc.).
- For the storage: Processing hardware requests and re-ordering hardware if the stock is running low.
- For the ward: Processing calls for delivering the patient from the ward to the operating room or back.
- For the laboratory: Receiving and processing incoming requests for blood analysis etc. Depending on the current task queue, agents can give an assessment of when the results will be available and negotiate with other agents that represent the requirements of the units that placed the orders.

#### 3.2 Agents in distributed engineering

In the context of distributed, simultaneous engineering, agents appear to be instrumental to support the subsequent tasks:

- Pro-actively notifying about updated documents to distribute information.
- Delivering documents among organizationally distributed partners crossing system boundaries while obeying organizational privacy.
- Transforming representations and abstracting information from several sources, e.g. for weekly audits of the development state.
- Negotiating deliverables and dates for coordination meetings. The coordination overhead for handling change request typically takes up to 50 percent of the workload. Agents for coordinating working tasks, i.e. due dates, arranging meetings, and notifying partners about changes, will be crucial for reducing workloads.
- Process auditing, e.g. monitoring timelines and delivering documents properly. More sophisticated agents can be built upon knowledge of the process in order to assess the current state (cf. [9]).
- Supporting coordination procedures. For example, most of the engineering tasks are performed in cooperation with suppliers yielding to the need to obey typical patterns, such as an engineering cycle with five reviews. An agent will typically schedule the review meetings biweekly and check the sequence of reviews with regards to gateways of the entire development process.

### 4 Conclusion and future work

The main contribution of this paper is the identification and description of two potential application scenarios for multi-agent systems. The software agents described in this document, have the potential to optimize clinical processes both in terms of costs and the quality of patient care. By pro-actively providing the clinical personnel with the right information at the right time, it can support medical reasoning and planning with a reliable information background. Agents can autonomously perform routine tasks or even analyze incoming data to detect potentially critical situations in advance. At the same time, the context-sensitive provision of information can reduce staff workload and time and shorten communication paths. Other potential advantages of agents in clinical health care have already been

pointed out in [6] and [2]. In the context of simultaneous engineering, multi-agent systems can be employed to overcome the lack of task-specific information by automating the distribution of document updates. Furthermore, they can assist in monitoring and coordinating timelines.

In future work, our main task will be to tackle technological barriers involved in the development of multi-agent systems. The lack of a clean-room software engineering methodology for single agents as well as their emerging behavior in a multi-agent setting is an especially important issue here [3]. Since the main task of the agents described above is the distribution and sharing of information, such a methodology should provide efficient and flexible means of dealing with and translating information structures, i.e. ontologies. Furthermore, since the complexity of the clinical and engineering processes demands for frequent tests of the executing system, support for rapid prototyping and respective development tools is needed. We are currently upgrading the software engineering method for "single-agent" knowledge-based systems described in [4] for the development of multi-agent systems. The main idea here is to represent ontologies in the object-oriented modeling language UML and to map these models to structure-preserving Java classes, in support of round-trip engineering.

## ACKNOWLEDGEMENTS

The authors would like to thank Dr. Bernd Claßen, Dr. Bernhard Schwilk and Martin Sedlmayr for providing medical background knowledge.

## REFERENCES

- [1] A. Dix, 'Computer Supported Cooperative Work: A Framework', in *Design Issues in CSCW*, ed., D. Rosenberg, 9–26, Springer Verlag, London, (1994).
- [2] J. Huang, N. Jennings, and J. Fox, 'An Agent-based Approach to Health Care Management', *Applied Artificial Intelligence: An International Journal*, **9**(4), 401–420, (1995).
- [3] N. Jennings, K. Sycara, and M. Wooldridge, 'A Roadmap of Agent Research and Development', *Int. Journal of Autonomous Agents and Multi-Agent Systems*, **1**(1), 7–38, (1998).
- [4] H. Knublauch and T. Rose, 'Round-Trip Engineering of Ontologies for Knowledge-Based Systems', in *Proc. of the Twelfth International Conference on Software Engineering and Knowledge Engineering (SEKE)*, Chicago, IL, (2000).
- [5] H. Knublauch, T. Rose, and M. Sedlmayr, 'Towards a Multi-Agent System for Pro-active Information Management in Anesthesia', in *Proc. of the Agents-2000 Workshop on Autonomous Agents in Health Care*, Barcelona, Spain, (2000).
- [6] G. Lanzola, L. Gatti, S. Falasconi, and M. Stefanelli, 'A Framework for Building Cooperative Software Agents in Medical Applications', *Artificial Intelligence in Medicine*, **16**(3), 223–249, (1999).
- [7] H. Luczak, D. Herbst, C. Schlick, J. Springer, and J. Stahl, 'Kooperative Konstruktion und Entwicklung', in *Kreative Unternehmen*, eds., R. Reichwald and H. Wildemann, Schäffer-Poeschel, Stuttgart, (1995).
- [8] J. Pennel and R. Winner, 'Concurrent Engineering: Practices and Prospects', in *Proc. of the GLOBECOM'89 IEEE Global Telecommunications Conference and Exhibition for the 1990s and Beyond*, pp. 647–655, Dallas, TX, (1989).
- [9] C. Rupprecht, M. Fünffinger, H. Knublauch, and T. Rose, 'Capture and Dissemination of Experience about the Construction of Engineering Processes', in *Proc. of the 12th Conference on Advanced Information Systems Engineering (CAISE)*, Stockholm, Sweden, (2000).

# Perspectives on Plan-based Multiagent Systems for Distributed Supply Chain Management in the Steel Industry<sup>1</sup>

Michael Beetz, Jürgen Schumacher, Armin B. Cremers, Bernd Hellingrath\*, and Christian Mazzocco\*  
Department of Computer Science III, University of Bonn,  
Roemerstr. 164, D-53117, Bonn, Germany.

\* Fraunhofer IML, Joseph-von-Fraunhofer-Str. 2-4 D-44227 Dortmund, Germany.

**Abstract.** Proper management of supply chains has been seen to be of manifest importance. This paper discusses how the application of new results in Artificial Intelligence and multiagent system research can serve to improve supply chain management by making supply chains self-adapting on the basis of past experience, able to better cope with incomplete information and more robust. Although the example of the steel industry is used, the principles discussed have general applicability.

## 1 Introduction

The dynamics, globalization, and frequent variations of customer demands that are typical for many of today's markets require companies to form *supply chains*, flexible and cooperative business partnerships that enable them to stay more competitive in their markets [17, 14, 12]. Supply chains are networks of autonomous business entities that collectively procure, manufacture, and distribute certain products. The objective of supply chains is to respond more accurately and more quickly to customer demand and to keep the size of the inventory necessary to respond to changing customer demands to a minimum. To do so, the supply chain management must transform available production resources and incomplete, inaccurate, and unreliable information about the market into *coordinated* and *optimized* plans for the procurement, production, replenishment, and distribution of goods. However, it is not sufficient for the supply chains to be merely properly planned. They must also be flexibly and robustly executed because the operation of supply chains is often jeopardized by many external influencing factors such as bank rate changes, delays in the deliveries of ordered semi-finished goods, and failures of production and transportation facilities.

In our research we investigate computational models for more effective control of supply chains in the steel industry. These supply chains present several challenging problems for the application of AI techniques and multi-agent systems. Firstly, the supply chains of the steel producers are very deep. Starting from raw materials such as iron ore and coal, they produce complete modules – for example for the automotive industry – which necessitate a multitude of produc-

tion steps. The large number of different end-products demanded by consumers requires provident and flexible planning and smooth coordination of the production process. The production steps within the supply chains must satisfy complex and diverse technological constraints. Satisfying these constraints requires powerful and sophisticated software tools for production planning. Finally, most steel producing companies operate world-wide, having supplier relationships between their plants. This requires that the logistics and production planning operations of the different plants be properly synchronized and closely coupled.

We have identified three key capabilities required by the next generation of supply chain management systems that have received surprisingly little attention in existing systems. Firstly, SCM systems must be capable of coping with incomplete and unreliable information. Secondly, to make supply chain processes more reliable, planning systems have to plan for accomplishing robustness, even when facing significant disturbances. Thirdly, the systems should learn from experience to become better over time.

In the project MAP-SCM (Multiagent, plan-based Supply Chain Management) we develop a novel and comprehensive model for the plan-based control of supply chains. The model is developed on the basis of the *process chain model* [13], which is extended to accommodate an agent-oriented view of supply chain management. The model forms the groundwork for a new class of supply chain management systems that

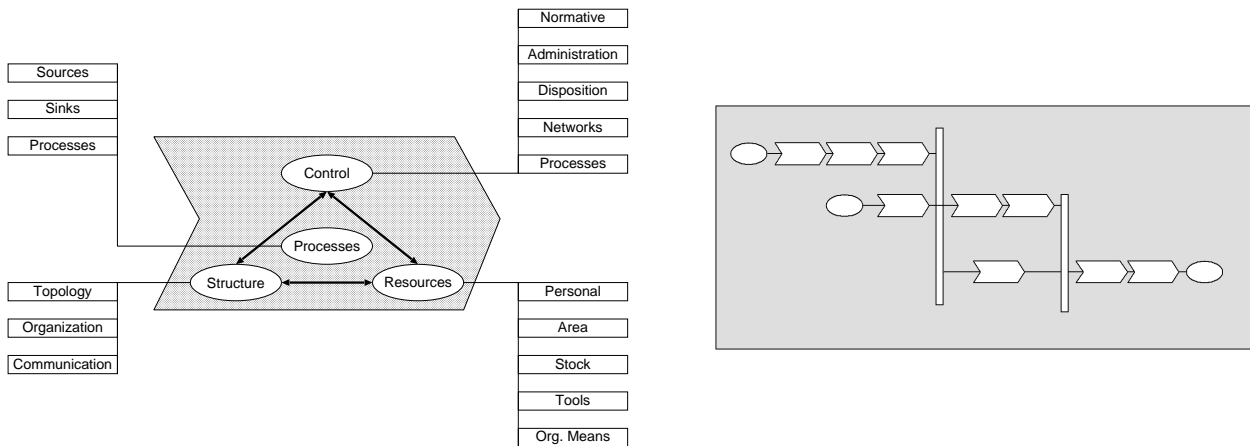
- support autonomy, secrecy, and information sharing among the business entities of a supply chain;
- can learn better control of supply chain processes from experience; and
- are capable of constructing reliable and flexible supply chains.

## 2 Design Principles

In the remainder of the paper we briefly sketch the principles underlying the design of our supply chain management system, which we consider to be the following.

- The process chain is the basic modeling tool for supply chain processes.

<sup>1</sup> The research described in this paper is partly funded by the Deutsche Forschungsgemeinschaft (DFG) (Contract No. BE 2200 5-1). The project is part of the German special research initiative "Software Agents in Business Applications"



**Figure 1.** Element of the process chain model (left) and process chain (right).

- The computational model of supply chain management distinguishes between strategical and tactical supply chain management.
- The information flow within the supply chain is mediated through predefined classes of speech acts.
- The supply chain management system is implemented as a *structured reactive controller* in order to allow for the flexible and robust execution of supply chain processes.
- The supply chain management system is designed to be adaptive: it is capable of improving its performance through long-term operation.
- The SCM system applies expectation-oriented planning of SC processes in order to form robust and flexible supply chains.
- Multi-agent software tools support the co-operation of supply chain partners by providing tools for sharing information that take the autonomy and non-disclosure policies of business entities into account.

## 2.1 The Process Chain Model for Supply Chains

The project is built on the foundation of the *process chain model* [13], a model of all processes occurring in the production and logistics of the production union. The elements of this process chain model have been developed at the Fraunhofer IML and were successfully applied in numerous industrial projects [5]. Major objectives for the realisation of this modelling approach have been: the visualisation of logistic chains in order to achieve a transparency across the whole order fulfilment process; offering a systematic collection of influence factors for the design of supply chain processes as well as their effect on the enterprise objectives; and giving sustainable support for the effective reorganisation of logistic process chains.

The basic element of this modelling approach is the process chain which incorporates different views of the system to be designed or analyzed. This modelling approach will be extended in order to cover the negotiation and co-ordination processes necessary to decide on order placement and the reaction to disturbances. We use multiagent system technology [24] for the realization of these extensions.

## 2.2 Supply Chain Management as a Planning/Control Process

Supply chains are flexible, cooperative, and longterm business partnerships of autonomous business entities that collectively procure, manufacture, and distribute certain products. The business partners provide means for production, storage, and transportation of intermediate and final goods. Means have capacities, setup times, cost, etc. In a supply chain contracts and business agreements are placed to control availability and cost of intermediate/final goods and plant utilization. The objective of supply chains is to meet customer demands at minimal production cost.

In our approach supply chain management implies two different kinds of planning/control problems:

1. **Strategical supply chain management.** Strategical SCM is concerned with the planning of production means and storage capacities in order to maximize the future expected utility of the supply chain. The means for controlling the capacities of production means are the placement of contracts about the availability and cost of goods and plant utilization. The contracts and business agreements formed in the strategical SCM constitute the constraints for the tactical SCM.
2. **Tactical supply chain management.** Tactical SCM is concerned with the effective execution of specific orders under the constraints by production resources and customer orders. This task comprises the planning and scheduling of production steps, the monitoring of the scheduled activity, and — if required — the rescheduling of activities if the situation has changed. Unlike the strategical SCM, that aims at longterm optimization of supply chains, tactical SCM considers a supply chain process as an isolated problem solving episode.

The key technical challenges of controlling/planning supply chain processes are implied by the lack of complete information about the internal operations of many business entities and the dynamical change of the contexts in which the supply chain processes are to be executed.

### 2.3 Speechact-based Information Flow

The information exchange within supply chains are viewed as *speech acts* [1, 8, 9], that is the exchange of information has effects and is performed to achieve some ends. MAP-SCM provides speech acts for plan generation and plan execution. The speech acts for supply chain plan generation comprise requesting goods, making offers, negotiation, and proposals for obligations. The speech acts for execution comprises the activation of production steps, signalling the progress of their execution, acknowledging the receipt of goods, assessing the quality of received goods, and so on. Thus, MAP-SCM interacts with supply chain managers, “perceives” the state of supply chain processes, and controls production steps via speech acts.

The speech acts themselves are specified in KQML (Knowledge Query and Manipulation Language) [10] and their content in KIF (Knowledge Interchange Format) [16]. Figure 2 shows an example of a MAP-SCM speechact.

#### (REQUEST

```

:SENDER Supply Chain Manager Plant A
:RECEIVER Supply Chain Manager Plant B
:TIME May 27, 2000
:REPLY-WITH “Re: Request for car axes A-2000-59”
:CONTENT REQUEST
    Good: car axes A-2000-59
    Amount: 500
    Delivery date: August 3rd, 2000

```

Figure 2. A MAP-SCM speechact for making the obligation to deliver 500 car axes by August 3rd, 2000.

MAP-SCM uses speech acts as data structures. Externally, supply chain managers interact with graphical interfaces that represent the state of the supply chain and provide the managers with means to communicate, negotiate, and coordinate with business partners in the supply chains. Thus the MAP-SCM speechact interpreter interprets incoming speech acts and updates the graphical interface of a manager accordingly and transforms user input into speechacts and sends the speechacts to their intended receivers.

Besides constituting an effective means for the control of user interaction, the speech acts are also used for updating the states of supply chain processes. Thus, upon receiving the commitment to deliver 500 car axes by August 3rd, 2000, the MAP-SCM speechact interpreter creates an instance of a process chain element for the delivery of 500 car axes and inserts it into the supply chain process and links it using consumer and producer links to the prior and subsequent process chain elements in the supply chain process. Then the parameters of the process chain element such as the “requested amount” and “requested completion date” are filled according to the content of the speech act. Later when the business partner who has requested the axes acknowledges their receipt, other parameters of the process chain element such as the “actual delivery time” and the “assessed product” quality are updated according to the acknowledging receipt.

### 2.4 Flexible and Robust Execution of Supply Chains

We will use a plan-based controller for supply chains – which we call a Structured Reactive Controller (SRC) [6] – that specifies a de-

fault execution strategy for a supply chain and can adapt itself to non-standard situations. SRCs execute three kinds of control processes: routine activities that handle process control in standard situations, monitoring processes that detect non-standard situations and process failures (see below), and planning processes that are responsible for the execution time adaptation of supply chain processes to non-standard situations.

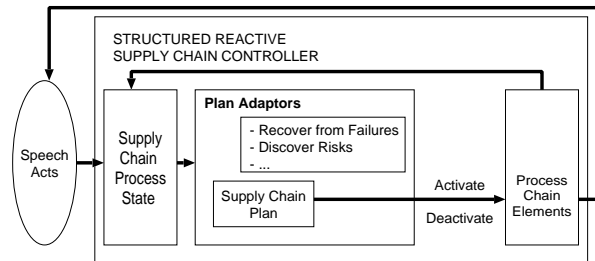


Figure 3. Components of a structured reactive controller.

The main components of a structured reactive controller for supply chain processes (see Figure 3) are the *process modules*, the *supply chain process state*, and the *structured reactive supply chain plan*. The elementary program units in the structured reactive controller are process modules, which correspond to the execution of process chain elements. The supply chain process state, which is updated continuously according to the information contained in the speech acts that are sent within the supply chain, provide information about the current state of the process chain elements and the supply chain process as a whole. The structured reactive plan specifies how the supply chain is to respond to the speech acts it receives in order to satisfy customer demand fast and efficiently. A structured reactive supply chain plan is a collection of concurrent control routines which specifies process execution in standard situations and that can adapt itself to non-standard situations by executing planned responses [7].

Structured reactive controllers operate as follows. When given a set of customer demands, the structured reactive controller computes default supply chain for the fulfillment of each individual demand and then executes them concurrently.

In order to generate such default plans the business partners provide rough estimates of their projected production capabilities and stock levels that can be used by the supply chain to generate offers for the fulfillment of customer demands quickly. Upon the customer's ordering, the production steps for the order are integrated into the supply chain process based on the projected production capabilities. The supply chain steps for achieving the new orders are then passed to the business partners that are to perform them. These steps form the basis for negotiating their detailed execution. The proposed production steps give the partners the opportunity of negotiating their contracts to maximize their own profits without jeopardizing the success of the supply chain process. In particular, this method gives business partners that are directly connected in the supplier-customer relation the opportunity to dynamically form coalitions in which the constraints of the proposed schedule can be relaxed as long as constraints with non-group members remain satisfied.

The elementary components of these schedules are the process chain elements. In order to enable supply chain plans to handle ranges of standard situations, the process chain elements are asso-

ciated with time windows that specify the earliest and latest start time for a process chain element and its earliest and latest completion time.

These default supply chains are general and flexible: they work for standard situations and also when executed concurrently with other routine activities. Default supply chains can cope well with partly unknown and changing situations and interactively co-operate with human expert planners. For standard situations, the execution of these default supply chain plans cause an appropriate performance. While the SC management system executes the chain it monitors its operation for non-standard situations. If it encounters a non-standard situation, it will try to anticipate problems by predicting how the chain might work in these non-standard situations. If necessary, it revises its routines to make them robust for this kind of situation. Finally, it integrates the proposed revisions smoothly into the supply chain.

A key constituent of MAP-SCM's robustness and flexibility is its capability to locally recover from failures in a supply chain process. For example, a process failure is signalled by the MAP-SCM system if the deadline of an obligation passes without the obligation being satisfied. In this case a failure description is computed that comprises the diagnosed causes of the failure (missing resources, non-operational machines, missing transportation means, etc.) and the set of process chain elements whose successful execution is jeopardized by the unachieved obligation. Our approach to recovering from such execution failures is to determine small subnetworks of the overall supply chain process that are crucially affected by the failure but can probably be rescheduled in a way to cause minimal disturbances for the remaining part of the supply chain process.

## 2.5 Adaptive Supply Chain Management Systems

The key idea for making the supply chain management system adaptive is to monitor the information flow and shared information bases within the chains in order to generate probabilistic models of the parameters of the *process chain model*. The models of the frequency of parameter values that we intend to learn from experience are (1) the time resources needed for carrying out subprocess modelled by a process chain element; (2) the likelihood of predetermined categories of problems; (3) the quality of the goods produced by the process chain element; and (4) expectations about the degree to which production obligations are satisfied.

Thus, the MAP-SCM speechact interpreter does not update the state of the supply chain process and the interfaces of the supply chain managers but also records the data contained in the speechacts in order to enable MAP-SCM to learn from experience. Therefore, MAP-SCM stores for each process chain element of the supply chain process a data record that contains all the element's management relevant parameter values. The purpose of this data collection is the formation of predictive models for classes of process chain elements. These predictive models should allow for the anticipation of probable supply chain process failures. The predictive models are essentially mappings from context information, obligations, availability of raw material into probability distributions for the delivery times and quality level of the material produced by a process chain element.

In order to build these models we intend to apply data mining techniques and techniques for learning Bayesian networks. We will also evaluate the approaches to adaptation and learning in multiagent systems with respect to their applicability for learning predictive models of process chain elements [22, 23].

## 2.6 Prediction-based Formation and Revision of Supply Chains

In our view, another key problem with existing supply chain management systems is that they are limited to optimize the supply chain for a given situation [11]. Even though, these optimization techniques allow for the analysis of *what if* scenarios they are incapable of constructing flexible and robust supply chains.

In our research we complement existing optimization planning techniques with techniques from decision theoretic transformational planning [15, 7]. In general, decision theoretic planning techniques aim at the construction of supply chains with the highest expected utility for given probability distributions over the expected state of the market and the facilities available in the supply chain, expected disturbances and possible market changes, and the causal models of process chain elements. Of course, the determination of supply chains with the highest expected utility is infeasible.

To avoid the problems of infeasibility, we use a standard optimization planning tool to compute a default supply chain for a default situation. This default supply chain will then serve as an initial candidate supply chain for the computation of a more flexible and robust supply chain that has a higher expected utility. To compute better supply chains the SCM system applies a transformational planning technique, which iteratively performs the following steps [15, 7].

It first probabilistically samples possible execution scenarios using the predictive models of process chain elements that we have already described in Section 2.5 and by probabilistically sampling situations from the probability distribution over the current system state. Based on the sampled supply chain scenarios, the system tries to estimate the severities of predicted problems. The supply chain is then transformed to forestall the predicted problems in order to increase its expected utility.

A method that has already been successfully used in supply chain management, although not for their automated control, is the design of supply chains such that decisions based on unreliable information can be postponed. The basic idea is to redesign the supply chain such that the supply chain steps that require decisions based on uncertain information are moved to the end of supply chains. This way, the supply chain can react more readily to unpredictable market changes.

## 2.7 Multiagent-based Tools for Supply Chain Management

The use of multiagent systems has been proposed by a number of researchers in order to improve SCM systems [19, 18, 20, 21]. Indeed, the tools for implementing multiagent systems have to offer several methods for the improved realization of SCM systems. Key methods are the ones for controlling the communication, co-operation, and negotiation between supply chain partners. In our research project we use the COOL language for realizing the communication between supply chain partners [4, 3, 2]. COOL is a powerful language for the specification of speech acts that represent coordination protocols between supply chain partners.

Another key area, where multiagent systems mechanisms provide valuable means for making SCM systems more acceptable is the support of the autonomy and non-disclosure policies of the supply chain partners that the multiagent systems can provide.

## 3 Conclusions

In this paper we described and discussed the basic design of a new category of supply chain management systems. The distinguishing

feature of the design is that it better reflects the dynamics and uncertainty of the markets that the supply chain is embedded in. This is accomplished by using the *process chain model* as the comprehensive model for the representation of supply chain processes. Based on this model, novel learning, planning, and execution techniques for supply chain control are realized. These techniques work with explicit representations of the uncertainty that can be expected in supply chains and which can, at least partially, be learned from experience. These techniques, which to the best of our knowledge, have not yet been applied to the problem of supply chain management, have already proven to be successful in the control of autonomous robot control.

## REFERENCES

- [1] J. L. Austin. *How To Do Things with Words*. Harvard University Press, Cambridge, Massachusetts, 1962.
- [2] M. Barbuceanu and M. S. Fox. The architecture of an agent building shell. In M. Wooldridge, J. P. Müller, and M. Tambe, editors, *Intelligent Agents II (LNAI 1037)*, pages 235–251. Springer-Verlag: Heidelberg, Germany, 1996.
- [3] Mihai Barbuceanu. A negotiation shell. In *Proceedings of the Third International Conference on Autonomous Agents (Agents'99)*, pages 348–349, Seattle, WA, USA, 1999. ACM Press.
- [4] Mihai Barbuceanu and Mark S. Fox. Integrating communicative action, conversations and decision theory to coordinate agents. In W. Lewis Johnson and Barbara Hayes-Roth, editors, *Proceedings of the First International Conference on Autonomous Agents (Agents'97)*, pages 49–58, Marina del Rey, CA, USA, 1997. ACM Press.
- [5] H. Beckmann and A. Schmidt. Prozeßkettenmanagement. In *Materialfluß und Logistik Sonderpublikation "Logistik Hochburg Dortmund"*, pages 18–22. Henrich Publikationen GmbH, Frankfurt, 1996.
- [6] M. Beetz. Structured reactive controllers — a computational model of everyday activity. In O. Etzioni, J. Müller, and J. Bradshaw, editors, *Proceedings of the Third International Conference on Autonomous Agents*, pages 228–235. ACM Press, 1999.
- [7] M. Beetz and D. McDermott. Improving robot plans during their execution. In Kris Hammond, editor, *Second International Conference on AI Planning Systems*, pages 3–12, Morgan Kaufmann, 1994.
- [8] P. Cohen and H. Levesque. Communicative actions for artificial agents. In *Proceedings of the International Conference on Multi-Agent Systems*, Cambridge, Ma, 1995. AAAI Press.
- [9] P. Cohen and C. Perrault. Elements of a plan-based theory of speech acts. *Cognitive Science*, 3(3):177–212, 1979.
- [10] T. Finin, R. Fritzon, D. McKay, and R. McEntire. KQML – A language and protocol for knowledge and information exchange. In *Proceedings of the 13th International Workshop on Distributed Artificial Intelligence*, pages 126–136, Seattle, WA, July 1994.
- [11] F. Gehr, J. Braun, and B. Hellingrath. Marktstudie supply chain management software. Technical report, Fraunhofer IML, Fraunhofer IPA, Dortmund, 1999.
- [12] B. Hellingrath. Unternehmensübergreifende optimierung der logistikkette. *hk*, Okt 1999.
- [13] A. Kuhn. *Prozeßketten in der Logistik: Entwicklungstrends und Umsetzungsstrategien*. Praxiswissen, Dortmund, 1995.
- [14] A. Kuhn, B. Hellingrath, and M. Kloth. Anforderungen an das supply-chain-management der zukunft. *Information Management and Consulting*, 3:22–27, 1998.
- [15] D. McDermott. Transformational planning of reactive behavior. Research Report YALEU/DCS/RR-941, Yale University, 1992.
- [16] R. Neches, R. Fikes, T. Finin, T. Gruber, R. Patil, T. Senator, and W. Swarton. Enabling technology for knowledge sharing. *AI Magazine*, pages 37–51, March 1991.
- [17] Hyacinth S. Nwana, Jeff Rosenschein, Tuomas Sandholm, Carles Sierra, Pattie Maes, and Rob Guttman. Agent-mediated electronic commerce: Issues, challenges and some viewpoints. In Katia P. Sycara and Michael Wooldridge, editors, *Proceedings of the 2nd International Conference on Autonomous Agents (AGENTS-98)*, pages 189–196, New York, May 9–13 1998. ACM Press.
- [18] Norman M. Sadeh, Tom Laliberty, Stephen F. Smith, and Robert Bryant. Development of an integrated process planning/production scheduling shell for agile manufacturing. Technical Report CMU-RI-TR-95-40, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, 1995.
- [19] J. M. Swaminathan, N. M. Sadeh, and S. F. Smith. Information exchange in the supply chain. Technical Report CMU-RI-TR-95-36, Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213-3890, 1995.
- [20] Katia Sycara, Stephen Roth, Norman Sadeh, and Mark S. Fox. Distributing production control. In *Proceedings of the Fourth International Conference on Expert Systems in Production and Operations Management (ESPOM-90)*, Hilton Head Island, SC, USA, 1990.
- [21] Katia P. Sycara, Steven P. Roth, Norman Sadeh, and Mark S. Fox. Resource allocation in distributed factory scheduling. *IEEE Expert*, 6(1):29–40, February 1991.
- [22] G. Weiß and S. Sen, editors. *Adaptation and Learning in Multi-Agent Systems*. Lecture Notes in Artificial Intelligence. Springer Verlag, Berlin, 1996.
- [23] Gerhard Weiß. Adaptation and learning in multi-agent systems: Some remarks and a bibliography. In Gerhard Weiß and Sandip Sen, editors, *Adaptation and Learning in Multi-Agent Systems*, Lecture Notes in Artificial Intelligence, pages 1–21. Springer Verlag, Berlin, 1996.
- [24] Gerhard Weiss, editor. *MULTIAGENT SYSTEMS: A Modern Approach to Distributed Artificial Intelligence*. MIT Press, 1999.



## Workshop Organization Committee

**Nadia Abchiche**, Université Paris 8, Département Informatique, Laboratoire d'Intelligence Artificielle,  
2, rue de la Liberté, 93526 Saint Denis Cedex 02, France.  
Email: abchiche@ai.univ-paris8.fr

**Paul Davidsson**, University of Karlskrona/Ronneby, Department of Computer Science,  
37225 Ronneby, Sweden.  
Email: Paul.Davidsson@ipd.hk-r.se

**Yves Demazeau**, Laboratoire Leibniz, Institut IMAG,  
46, avenue Félix Viallet, 38031 Grenoble Cédex, France.  
Email: Yves.Demazeau@imag.fr

**Francisco. J. Garijo**, Telefonica Investigacion Y Desarrollo,  
C/ Emilio Vargas 6, 28043 Madrid, Spain.  
Email: fgarijo@tid.es

**Otthein Herzog**, Universität Bremen, Technologie-Zentrum Informatik (TZI),  
P.O.Box 33 04 40, 28334 Bremen, Germany.  
Email: herzog@informatik.uni-bremen.de

**Stefan Kirn**, Technische Universität Ilmenau, Institut für Wirtschaftsinformatik,  
P.O.Box 05 65, 98684 Ilmenau, Germany.  
Email: wi2-office@wirtschaft.tu-ilmenau.de

**Peter Knirsch**, Universität Bremen, Forschungsverbund Logistik,  
P.O.Box 33 04 40, 28334 Bremen, Germany.  
Email: knirsch@informatik.uni-bremen.de

**Heinz-Jürgen Müller**, T-Nova Deutsche Telekom Innovationsgesellschaft mbH  
Technologiezentrum Darmstadt FE14k, 64307 Darmstadt, Germany.  
Email: Heinz-Juergen.Mueller@telekom.de

**Charles Petrie**, Stanford University, Department of Mechanical Engineering,  
Center for Design Research (CDR),  
Packard Bldg. #232, Stanford, California 94305-9510, USA.  
Email: petrie@stanford.edu

**Mathias Petsch**, Technische Universität Ilmenau, Institut für Wirtschaftsinformatik,  
P.O.Box 05 65, 98684 Ilmenau, Germany.  
Email: Mathias.Petsch@wirtschaft.tu-ilmenau.de

**Catherine Tessier**, ONERA-CERT, Département Commande des Systèmes et Dynamique du vol,  
Unité de recherche Conduite et Décision,  
2 avenue Edouard-Belin, 31055 Toulouse Cedex 4, France.  
Email: cath@cert.fr

**Ingo J. Timm**, Universität Bremen, Forschungsverbund Logistik and Technologie-Zentrum Informatik,  
P.O. Box 33 04 40, 28334 Bremen, Germany.  
Visiting Researcher at: Indiana University-Purdue University Indianapolis (IUPUI),  
Department of Computer and Information Science,  
723 W. Michigan St., Indianapolis, Indiana 46202-5132, USA.  
Email: i.timm@tzi.uni-bremen.de

## Author Index

- Beetz, Michael**, Department of Computer Science III, University of Bonn,  
Roemerstr. 164, D-53117 Bonn, Germany.  
Email: beetz@cs.uni-bonn.de
- Breckle, Hans**, Forschungszentrum Informatik, Dept. Mobility Management and Robotics,  
Haid-und-Neu-Str. 10-14, D-76131 Karlsruhe, Germany.  
Email: breckle@fzi.de
- Brun, Alessandro**, Politecnico di Milano,  
Piazza L.da Vinci, 32 – 20133 – Milan – Italy.  
Email: Alessandro.Brun@polimi.it
- Coelho, Helder**, Faculdade de Ciências,  
Campo Grande, 1700 Lisboa, Portugal.  
Email: hcoelho@di.fc.ul.pt
- Cremers, Armin B.**, Department of Computer Science III, University of Bonn,  
Roemerstr. 164, D-53117 Bonn, Germany.  
Email: abc@cs.uni-bonn.de
- Fernando Lopes**, INETI,  
Estrada do Paço do Lumiar, 1699 Lisboa Codex, Portugal.  
Email: flopes@dms.ineti.pt
- Gerber, Andreas**, German Research Center for Artificial Intelligence (DFKI GmbH),  
Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany.  
Email: agerber@dfki.de
- Haag, Zsolt**, Department of Computing, Glasgow Caledonian University,  
Glasgow, G4 0BA, UK.  
Email: z.haag@gcal.ac.uk
- Hellingrath, Bernd**, Fraunhofer IML,  
Joseph-von-Fraunhofer-Str. 2-4, D-44227 Dortmund, Germany.  
Email: helling@iml.fhg.de
- Henoch, Jens**, Institute for Operations Research, Swiss Federal Institute of Technology, Zürich,  
Clausiusstrasse 47, 8092 Zürich, Switzerland.  
Email: henocho@ifor.math.ethz.ch
- Hollmann, Oliver**, TZI, Center for Computing Technologies, University of Bremen,  
Universitätsallee 21-23, D-28359 Bremen, Germany.  
Email: oho@tzi.de
- Knublauch, Holger**, Research Institute for Applied Knowledge Processing (FAW),  
Helmholtzstr. 16, 89081 Ulm, Germany.  
Email: Holger.Knublauch@faw.uni-ulm.de

- Mair, Quentin**, Department of Computing, Glasgow Caledonian University,  
Glasgow, G4 0BA, UK.  
Email: qma@gcal.ac.uk
- Mamede, Nuno**, IST,  
Avenida Rovisco Pais, 1049-001 Lisboa, Portugal.  
Email: Nuno.Mamede@acm.org
- Mazzocco, Christian**, Fraunhofer IML,  
Joseph-von-Fraunhofer-Str. 2-4, D-44227 Dortmund, Germany.  
Email: mazzocco@iml.fhg.de
- Novais, A. Q.**, INETI,  
Estrada do Paço do Lumiar, 1699 Lisboa Codex, Portugal.  
Email: anovais@dms.ineti.pt
- Portioli-Staudacher, Alberto**, Politecnico di Milano,  
Piazza L.da Vinci, 32 – 20133 – Milan – Italy.  
Email: Alberto.Portioli@polimi.it
- Rose, Thomas**, Research Institute for Applied Knowledge Processing (FAW),  
Helmholtzstr. 16, 89081 Ulm, Germany.  
Email: Thomas.Rose@faw.uni-ulm.de
- Schillo, Michael**, Multi-Agent Systems Group, Saarland University,  
Im Stadtwald, 66123 Saarbrücken, Germany.  
Email: schillo@ags.uni-sb.de
- Schumacher, Jürgen**, Department of Computer Science III, University of Bonn,  
Roemerstr. 164, D-53117 Bonn, Germany.  
Email: schumac1@cs.uni-bonn.de
- Thum, Thomas**, Dr. Thum Process-Engineering and Project-Management Consultancy,  
Neue Strasse 49, D-21073 Hamburg, Germany.  
Email: IngBueroDrThum@t-online.de
- Ulrich, Heinz**, Institute for Operations Research, Swiss Federal Institute of Technology, Zürich,  
Clausiusstrasse 47, 8092 Zürich, Switzerland.  
Email: ulrich@ifor.math.ethz.ch
- Vierke, Gero**, German Research Center for Artificial Intelligence (DFKI GmbH),  
Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany.  
Email: vierke@dfki.de
- Zinnikus, Ingo**, German Research Center for Artificial Intelligence (DFKI GmbH),  
Stuhlsatzenhausweg 3, D-66123 Saarbrücken, Germany.  
Email: zinnikus@dfki.de

## Keyword Index

Artificial Market Systems 55  
Chemical Factory 45  
Clinical Information Systems 85  
Conflict of Interests 61  
Contract-Net Protocol 29  
Distributed Artificial Intelligence 35  
Distributed Engineering Teams 85  
Electronic Commerce 21, 55  
Electronic Marketplace 21  
Engineering Works 45  
Information Logistics 85  
INTERRAP 29  
Logistics Networks 21  
Management of Complexity 45  
Management Systems 11  
Market Mechanism 11, 21  
Matchmaking 21  
Mobility 17  
Modeling 29, 45, 75  
Multi-Agent Communication 55  
Multi-Agent Coordination 55  
Multi-Agent Systems 11, 17, 45, 61, 85, 89  
Multi-Criterion Decision Making 35  
Multi-Agent Collaboration 55  
Negotiation 61  
Optimization 45  
Organizational Cybernetics 11  
Parallel Processing 45  
Planning 45, 61  
Rationality 35  
Self-Adaptation 89  
Services 17  
Simulation 17, 45  
Sociology 35  
Software Configuration management 75  
Speech-act 29, 89  
Supply Chain Management 61, 89  
Systems Engineering 45  
Transportation Logistics 29, 35, 45  
Virtual Software Cooperations 75