An Autonomous Multi-Agent Approach to Supply Chain Event Management

Lorena Bearzotti
Facultad de Ingeniería
Universidad Nacional Andrés Bello, Sede Reñaca
Viña del Mar, Chile
lo.bearzotti@uandresbello.edu

Enrique Salomone[1], Omar Chiotti[1,2]
INGAR[1], CIDISI[2]
CONICET[1], UTN FRSF[2]
Santa Fe, Argentina
{salomone, chiotti}@santafe-conicet.gov.ar

***Abstract*— Event management is a problem in the supply chain context that requires a solution with the goal of mitigate the event effect during the execution plan. We present an autonomous agent-based approach to support a system for this problem. Our proposal introduces two novel aspects: we conceive the system as a collaborative inter-organizational information system and we aim to provide autonomous mechanisms for the system to perform proactive control actions. We develop an example illustrating the principal concept and how this decomposition and collaborative negotiations allow finding a solution to an exception.**

Keywords* – *supply chain, event management, multi-agent system, auonomous behaviour.

I. INTRODUCTION

A Supply Chain (SC) can be defined as a network of autonomous business entities, collectively responsible for procurement, production and distribution activities associated with one or more families of related products [14]. This definition, as well as others ones [3] [12] [4], states the need of integrating every member into both information and material flow with the aim of meeting clients′ requirements.

Current planning and execution systems are not flexible enough and lack adaptive capabilities to respond to changes caused by disruptive events in the SC. For this reason the SC does not respond effectively to unexpected events throughout the global chain [8]. There is an open gap between SC planning and execution systems, responsible for the SC not being able to adequately respond to unplanned events and hence real-time SC disruptions occur.

A new generation of information systems known as *SC Event Management Systems* (SCEM Systems) [8] [18] emphasize the necessity of exception-based management, supporting short term logistic decisions, avoiding complex cycles of re-planning.

In this context, *event management* is an essential process whose goal is to mitigate the effects of exceptions to the plan under execution. If the effect cannot be mitigated, the plan becomes obsolete due to it cannot adapt to the new circumstances, then a re-planning is necessary.

New information technology is key to perform effective SCEM [2] [9]. In this way, SCEM is defined as the business process where significant events are timely recognized, reactive actions are quickly triggered, the material and information flows are adjusted and the notification of key employees is immediate.

In this article, we present an autonomous agent-based approach to support a system for the SCEM problem. Our proposal introduces two novel aspects with regards to previous works: first, we conceive the system as a collaborative inter-organizational information system and second, we aim to provide autonomous mechanisms for the system to perform proactive control actions. The first aspect is based on the observation that the current approaches for SCEM are extensions of traditional ERP systems, and they are not well suited to support the inter-organizational collaboration. The second point is included in response to a requirement not yet covered by the existing proposals, which are mainly focused in addressing the monitoring, the capture and the communication of disruptive events. The ability to exert corrective control actions has been identified as an area barely explored [18]. The occurrence of unexpected events is a fact well known to the planning task, then SC planner develops plans with slacks. In our approach the control actions use the slack of the plans.

In the following section the related works are presented. Our approach is developed in Section 3 whereas in Section 4 an example is shown. The conclusions and future work are presented in the Section 5.

II. RELATED WORK

The SCEM implies to assess, monitor and evaluate events within and across companies, and to initiate consequent actions. The focus is on inter-company visibility of critical SC objects [8].

A report from AMR Research [7] specifies five components a software should possess to fall in to the SCEM category: monitor (measuring relevant events in real time), notify (alerts decision makers in real-time), simulate (find alternatives as a response to unforeseen events), control (analyzes and documents the effect for the subsequent SC processes, and allows the decision maker to proactively change previously established conditions) and measures (assessing, analyzing and evaluating historical data).

An approach for the SCEM problem solution can present different automation levels, and then it can be classified in the following groups:

- Monitoring: the system is planned as an extension of traditional Tracking and Tracing Systems.
- Alarm: in this group, the system can detect deviations in the plan and notify adequately.
- Autonomous Corrective: if a deviation is detected the system looks for a solution.

PROVE [15] and DIALOG [6] are monitoring systems. ECTL-Monitor [5], PAMAS [18] and CoS.MA [16] are alarm systems with different proactivity grades in the data recollection and notification. The third group is constituted by autonomous systems able to detect an event, make a decision about the exception and implement a solution if one exists. Our approach can be classified in the last level.

PROVE is a prototype whose role is to monitor the products in Virtual Enterprises. It only handles client orders.

DIALOG is based on software agents that share data to facilitate the orders tracking, offering information on its state. It does not consider reacting to an event.

ECTL-Monitor is an agent-based system. It will be embedded in the company´s Internet portal in order to provide the customers the possibility to track and trace their orders. This system provides proactive elements regarding notification capabilities but proactive data gathering is not provided, the process is initiated by user request. It does not present a dynamic adaptive behaviour to the change.

PAMAS is a SCEM system based on the monitoring of the orders when they move in the SC, detecting when an event affects an order, using adaptive order profile and integrating several data sources from the SC members. They proposed an algorithm that is mapped in to a multi-agent architecture. This proposal is an alarm system where the control actions are not part of the solution.

CoS.MA is another alarm system. Its peer-to-peer based architecture is aimed to integrate data from single members, so that all members have a visualization of pertinent data. With support from Auto-ID and mobile technologies tracking and tracing of products in the SC will be possible.

Another system proposed is SChEMA [11], it models the SCEM process, but is based on previous knowledge of which event can happen and how they affect the normal operation of SC. To previously know every event that can occur in the SC is a hard task. To avoid this problem, our approach proposes to identify the possible events sources and to define the control points on them. In this way, the identification of events and how they affect the plan is a dynamic process.

Error recovery is a problem in manufacturing systems and requires flexible and distributed solutions when unexpected events occur. There are many research efforts in this area [2] [17] and they are important because this problem is similar to SCEM problem, but they are limited to the intra-enterprise context.

Actually, Tracking and Tracing Systems are the status quo in most enterprises, but they cannot satisfy the requirements for a SCEM solution. The proposals were an evolution of these systems, but the next step is an autonomous SCEM system in the exceptions control.

III. Multi-Agent Based SCEM System

An automated event management system should be able to monitor events to detect exceptions and to support the process for analyzing the plan slacks to define control actions to mitigate the effect of these exceptions. When these control actions cannot be automatically generated, the event management system will notify the exception to the planning level for re-planning decisions.

In the context of SC, the organizations manage their planning processes, but the best of plans can be interrupted by unexpected events, then the SCEM goal is to allow that the SC can respond to unexpected events minimizing the impact of them. In this context an event is defined as a change of state. Examples of unplanned events are [8] [7]: A supplier advances or retards an order. A supplier cancels a planned and confirmed order. A customer changes or cancels an order. A customer asks for a new unplanned order. Unexpected change on the availability of the resource affected to the execution of an operation (any equipment breakdown, breakage of materials).

Any practical application can involve thousands of state variables, usually interrelated and these relationships may not be explicit in a global perspective. This makes the SCEM a complex control problem.

We have tackled the problem developing a *conceptual model* in which the main concepts of the problem are identified as well as how they are interrelated. Based in this model we propose a *multi-agent architecture* for the solution of the SCEM problem.

A. A Conceptual Model of SCEM

The SCEM problem is a complex control problem where it is possible to identify three types of variables: *observed variable* (observed during the execution process with the purpose of detecting the occurrence of an unexpected event), *controlled or state variable* (defines a control point, it has a plan with slacks defined by the planing system) and *decision variable* (independent variable whose value can be adjusted to mitigate the effects of an exception with the purpose of bringing the system back to the specified objectives).

To approach the complexity the global problem is decomposing into a set of simpler interwoven sub-problems. Each state variable is represented by a subsystem and for each of them, a control point and their observed and decision variables are defined.

The state variables are related with the possible source of events. The definition of a "plan" in this context provides the necessary information to identify the possible event sources. A plan is defined as the materials allocation to different places, where the allocation is represented by materials transformation and transference orders, stating the affected resources in each order. The execution of these orders is subject to an unexpected variability; the object of SCEM being to react to that variability by minimizing diversion of what has been planned. The conclusion is that the sources of events are of two types: Orders and Resources. Events related to an order imply a change in the amount specified in the order and/or in the time in which the order must be fulfilled, it represents the variation of some inventory with regards to the planned. On the other hand, the events related to the resources represent an unexpected change in the availability of the resources affected for the execution of an operation, for example today a truck breaks down and it will be unavailable for a week. Based on this identification it is possible to define the conceptual model.

Each state variable is represented as a control point, then a SCEM *Model* (SCEMMO) is defined as a network of inventory control points linked among them by supply processes. The supply processes use resources also modelled as resource control point. This approach is the main difference with regards to the other works in the area because in their models the main concept is the order. In our model the main elements are the resources. Perhaps taking the order as the main element of the model is useful when a monitoring and alarm activity is performed, but when the objective is to present autonomous behaviour this decomposition of the problem is not adequate.

Each control point or subsystem has to monitor its observed variables to detect the occurrence of an event and to analyze it to detect whether it produces an exception. When an exception is detected, the subsystem has to support the analysis process to define control actions with the goal of mitigating the effect of this exception. The subsystem can negotiate with its related subsystems to reach a collaborative control action to mitigate the effect of this exception. If appropriate joint control actions cannot be defined, the subsystem has to notify the exception to the planning level for re-planning decisions.

A control point, which is called RKU (*Resource Keeping Unit*), is defined for each required resource in the SC. The function of the RKU is to manage those resources that can alter the execution of the plan assigned to the resource it represents. In order to carry out this function, the attributes defining an RKU determine its availability. Some examples of possible types of RKUs are: processing units, transport equipment. A special kind of resources is: materials, raw material, products being processed and finished products. These resources are included in the term "inventories". Due to their particular characteristics, they are modeled as a specialized control point called MKU (*Material Keeping Unit*). The function of the MKU is to manage the events that occur in the inventory it represents. An MKU is defined by three basic attributes: material, container and place. Their combination allows their univocal identification. Conceptually, an MKU is a specialization of the RKU.

These control points are linked by the SP (*Supply Process*) that represents the transition of one or more MKUs to another or other MKUs. For this purpose, different RKUs are required: from this perspective, SP may be conceived as balance point among the different control points it relates. Then, each SP represents an activity that determines the way in which SKUs are related and which RKUs are required for this relation to be effective.

In this model, events are detected by RKUs and their specialized entities, the MKUs, who are in charge of evaluating if that event leads to an exception. If this is so, the RKU starts a search process for implementing a solution to the occurring exception. In case the solution is not found, the RKU is in charge of notifying that the plan has become obsolete. Then, the RKUs interact among one another only by means of the SPs.

B. The Multi-Agent Architecture for the SCEM

The agent technology is chosen because the multi-agent system is the best way to characterize and design distributed information system [10]. An agent-based SCEM system can be an effective solution by the existence of multiple facilities, inter-enterprise relations and a global environment. This system is developed to solve the SCEM problem by means of agent cooperation. The agents interact to obtain a common goal, but the agents have goals and interest then there are possibilities of conflicts. In this work we propose a coordination process as a negotiation process set among the agents.

Based on our conceptual model SCEMMO above describe, we propose a *SC Event Management Multi-Agent Architecture*) (SCEMMA). The approach is an automated system that manages events in the SC domain. It can define automated control actions to mitigate the deviation that an event could produce. If these control actions can not be inferred, then SCEMMA notifies the Planning System that an exception has occurred.

SCEMMA is composed by two types of principal agents: RKU Agent and SP Agent, plus a specialized agent called MKU Agent. Also, there are three service agents: EVA (*EVent Agent*), PAGE (*Planning AGEnt*) and IRA (*InteRaction Agent*) (Fig. 1). In each member of the SC there will be an EVA and a PAGE, and an IRA for each member related with. This approach preserves the member autonomy.

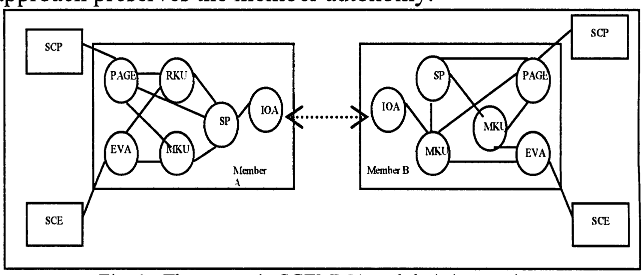

Fig. 1. The agents in SCEMMA and their interactions

RKU Agent represents a control point for each relevant resource in the SC. It has a Usage Agenda, from which it is possible to create a load profile that is the state plan for the resource. It indicates the required capacity of the resource and for how long. Data of Usage Agenda are: Order Number, Starting Date, Duration, Type (start, middle, end, used to indicate when the state change occurs during the period) and Name SP (SP related with the order).

MKU Agent represents a control point where the management of events related to inventory is done. Its basic attributes are: material, package and location. An MKU has an Input/Output List that defines to which SP Agents it is related, the information is the same that Usage Agenda. The relevant events are detected by comparing the current value with the planned value. The monitored variable is the current inventory value, and controlled variables are parameters of the Input/Output List.

SP Agent represents an action for the transition from one or more source MKUs to one or more final MKUs. There are three basic transitions: material change, packing change and location change. Composite transitions are also possible yielding seven types of transitions. In order to support its function, the SP agent maintains an Activity Plan that tracks the resources assigned for the execution. An Activity Plan is a 4-tuple indicating: Order Number, Start Date, Duration and a list indicating the resources involved in the activity and detailing when and how much is required, this list extends the BOM concept because considers all type resource not only materials.

PAGE is a service agent responsible for being the interface between the Planning System and the other components of SCEMMA. It receives the general plan from Planning System, then it determines the control points to represent the plan dynamics, after that, it creates the different agents, with their plans and slacks. When a RKU or MKU cannot solve an exception sends to PAGE a message informing the situation and PAGE sends to Planning System a notification.

EVA is a service agent that receives from the execution systems the events that occur and distribute them to RKUs or SKUs. Information on how to distribute events is contained in its knowledge base on which it is detailed the variable monitored by each RKU. It acts like awakening of agents.

IRA is another service agents, its goal is to be an interface with other related member, thus maintaining the autonomy of each member of the chain. One side the SP acts as a RKU but with the other side it acts as a SP.

C. *The Event Management Process*

The goal of SCEMMA is to minimize the disturbance of a plan that can be caused by unexpected events using the slack associated to this plan. To this aim, we identified the event management process as the main process. It has two associated sub-processes: the event monitoring process and exception control process. In SCEMMA the agents need to cooperate among them with the goal of finding a solution to an exception. The coordination is decentralized due to there is no unique coordinating agent. The proposal is that the role of coordinating agent is assumed by the agent that initiates the negotiation process when it detects an exception. It stops being coordinator when the negotiation process finishes.

An important aspect is determining what information the agent interchange with the goal of cooperating in the search of a solution. As a first approach we use a computational market economy to define the information to exchange; we are now analysing to use cooperative negotiation.

In this architecture when an event occurs in the external systems it is captured by the EVA. The EVA sends to the affected agent a message notifying what has happened, EVA knows who RKU is affected for this event using its knowledge base, there are a relation between monitored variable and events. When this happens, the RKU that receives the message evaluates the situation with this change in the value of the observed variable. If the change produces an exception, then the agent becomes a Coordinator Agent of the negotiation process, then the Coordinator sends messages to other agents SPs starting a negotiation. The proposed interaction protocol behind the negotiating process is based on the contract net protocol [13], in the next section, an example details how the coordination is obtained. The negotiation finishes when a solution is found or the deadline is reached. If there is a solution then the Coordinator Agent notifies the solution to other affected agents. When there is not a solution, the Coordinator Agent sends a message to the PAGE informing the problem, and this inform to Planning System that the plan is obsolete.

IV. An Example of SCEMMA Instantion

The example is a product distribution problem, involving a SC integrated by a supplier, a wholesaler with a central and a branch, and two retailers. Two products are distributed: P1, found in two packages (P11 and P12), and P2 found just in one package. All members in the SC present restrictions as regards their storage capacity as well as minimum levels of inventory required for these products.

The example presents a chain of storage and transport. Each member is in charge of a process for storing these products and there is a transport process among them. Considerations: events have their origin in the inventories (deadlines, breakdown, etc.) and resources (unavailable), the planning horizon is one week, the Retailer1 doesn't manage P11 and Retailer2 doesn't manage P12. The example is instantiated with the proposed model, SCEMMA, and its behavior when unexpected events occur is evaluated using the prototype SCEMMAS.

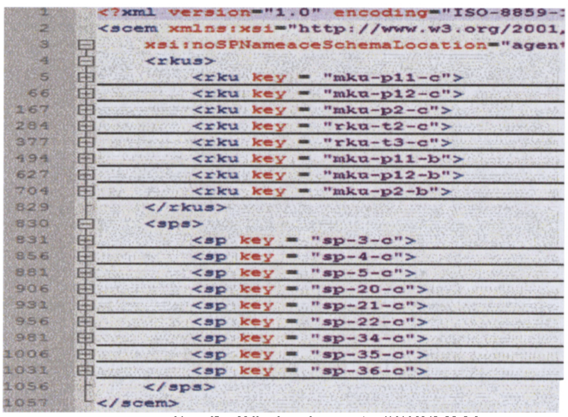

Fig. 2. Wholesaler part of SCEMMA

TABLE I. WHOLESALER PLAN

Order	Date	Origin	Destination	Product	Quantity	Resource
1	Day1	Central	Retailer1	P12	30	Truck3
2	Day1	Central	Retailer1	P2	40	Truck3
...						
35	Day5	Central	Branch	P12	20	Truck2
...						

The SC members collaboratively elaborate their supply plans that are the base for the agency generation. Table I shows part of Wholesaler Plan. When the agent PAGE, in the Wholesaler, receives this plan generates the different agents and builds the part of agency that correspond to supplier. In Fig. 2 part of agency is detailed. The information for each agent is elaborated. For example, the order 35 in this plan (Table I) is represented as order 5 in the MKU-P12-B, the order 7 in the MKU-P12-C and the order 8 in the RKU-T2-C. This can be seen in the Input/Output List for MKU-P12-B and MKU-P12-C, in the Usage Agenda for RKU-T2-C and in the Activity List for SP-35-C, in Figure 3, 4, 5, and 6. This process is carried out for all SC members.

Once the agency is instantiated starts its monitoring function of the plan in execution. For example, EVA receives a message informing that the P12 initial inventory in the Branch is 10 units instead of 5 units planned. Then, EVA sends a message to MKU-P12-B with this change. When MKU-P12-B analyzes this event finds that an exception has occurred. MKU-P12-B assumes the role of coordinator in the process to find a solution and generates solution proposals.

This process evolves in three stages: (1) Seeking to solve the exception working only with the period order; (2) Considering the orders of the periods immediately before and after the period where the exception occurs; (3) Considering all orders from actual time are. At each stage the process includes three steps: (a) Modifying only a parameter of one order; (b) Changing both parameters of one order; (c) Combining variations in more than one order.


```
<rku key = "mku-p11-c">
<rku key = "mku-p12-c">
<rku key = "mku-p2-c">
<rku key = "rku-t2-c">
<rku key = "rku-t3-c">
<rku key = "mku-p11-b">
<rku key = "mku-p12-b">
    <name>mku-p12-b</name>
    <ae>ae</ae>
    <configuration>
    <orders>
        <order key = "#0000">
        <order key = "#0001">
        <order key = "#0002">
        <order key = "#0003">
        <order key = "#0004">
        <order key = "#0005">
            <orderNumber>5</orderNumber>
            <start>5</start>
            <duration>1</duration>
            <quantity>20</quantity>
            <periodMode>1</periodMode>
            <SPName>sp-35-c</SPName>
        </order>
        <order key = "#0006">
        <order key = "#0006">
    </orders>
    <strategy>superCompuesta</strategy>
    <utilityFunction>default</utilityFunction>
</rku>
<rku key = "mku-p2-b">
```


Fig. 3. Input/Output List– MKU-P12-B


```
<rku key = "mku-p11-c">
<rku key = "mku-p12-c">
    <name>mku-p12-c</name>
    <ae>ae</ae>
    <configuration>
    <orders>
        <order key = "#0000">
        <order key = "#0001">
        <order key = "#0002">
        <order key = "#0003">
        <order key = "#0004">
        <order key = "#0005">
        <order key = "#0006">
        <order key = "#0007">
            <orderNumber>7</orderNumber>
            <start>5</start>
            <duration>1</duration>
            <quantity>-20</quantity>
            <periodMode>1</periodMode>
            <SPName>sp-35-c</SPName>
        </order>
        <order key = "#0008">
        <order key = "#0009">
        <order key = "#0010">
    </orders>
    <strategy>superCompuesta</strategy>
    <utilityFunction>default</utilityFunction>
</rku>
<rku key = "mku-p2-c">
<rku key = "rku-t2-c">
<rku key = "rku-t3-c">
<rku key = "mku-p11-b">
<rku key = "mku-p12-b">
```


Fig. 4. Input/Output List – MKU-P12-C

Then, MKU-P12-B begins the proposal generation process where the orders set is defined by the period order (order 5 and 6, but the order 6 is no negotiable, then only can work with the order 5) (Fig. 3). In this way MKU-P12-B finds two proposals: decrease in five units the order 5 or delay one day the order 5.

The order 5 is associated with SP-35-C, then with this information MKU-P12-B begins a negotiation protocol called double contract net, in this protocol MKU-P12-B is the initiator, then send to SP-35 a message with the proposals. When SP-35-C receives this message acts as a responder and


```
<rku key = "mku-p11-c">
<rku key = "mku-p12-c">
<rku key = "mku-p2-c">
<rku key = "rku-t2-c">
    <name>rku-t2-c</name>
    <ae>ae</ae>
    <configuration>
    <orders>
        <order key = "#0000">
        <order key = "#0001">
        <order key = "#0002">
        <order key = "#0003">
        <order key = "#0004">
        <order key = "#0005">
        <order key = "#0006">
        <order key = "#0007">
        <order key = "#0008">
            <orderNumber>8</orderNumber>
            <start>5</start>
            <duration>3</duration>
            <quantity>20</quantity>
            <periodMode>1</periodMode>
            <SPName>sp-35-c</SPName>
        </order>
        <order key = "#0009">
    </orders>
    <strategy>superCompuesta</strategy>
    <utilityFunction>default</utilityFunction>
</rku>
<rku key = "rku-t3-c">
<rku key = "mku-p11-b">
<rku key = "mku-p12-b">
<rku key = "mku-p2-b">
```


Fig. 5. Usage Agenda – RKU-T2-C

using its Activity List generates the proposals to the affected agents (RKU-T2-C and MKU-P12-C). With these agents the SP-35-C acts like an initiator; it is in a SP where the double role happens because for the same negotiation it is responder and initiator. To RKU-T2-C the proposals are: (1) decrease in five units the order 8, (2) delay one day the order 8. The proposals to MKU-P12-B are: (1) reduce the order 7 to 15 units, (2) delay one day the order. With a message SP-35-C sends these proposals to each agent and waits the answer. MKU-P12-C and RKU-T2-C in this process are responders, they receive the proposals and simulate changes proposed to determine whether they accept or reject requests for changes.

For MKU-P12-C both proposals are valid because any of them generate an infeasible state, then responses with an accept proposals 1 and 2 to SP-35-C. In the other hands RKU-T2-C only can accept the first proposal then send an accept proposal 1 and reject proposal 2. When the SP-35-C receives responses from responders must unify them in order to generate an answer to its initiator. Thus the SP-35-C sends to MKU-P12-B a message accepting the proposal 1 and rejecting the proposal 2. Upon receiving the response of SP-35-C, MKU-P12-B implements as a solution the proposal 1 (reducing 5 units in the order 5) sending a message to the SP-35-C, and it to MKU-P12-C and RKU-T2-C to confirms the acceptation of proposal 1, so that everyone update their data. Thus an exception that had become infeasible a plan is settled through cooperation of various agents using their slacks that can absorb the exception that had been generated.

V. Conclusion and Future Works

The agent-based approach for the SCEM problem presented offers a solution to the existing gap between SC planning and execution, allowing the reaction to unexpected events in a collaborative way using plans slacks. In this way the SC visibility is improved, as well as its agility and ability to respond to unexpected events. We first obtained a conceptual model that faces the complexity of the problem. This model is implemented by an agent based architecture called SCEMMA, which was prototyped using JADE [1] obtaining SCEMMAS. Our approach SCEMMA allows reducing the negative effects of disruptive events searching for appropriate solution. Its proactive and autonomous behavior in the solutions generation differentiates it from other approaches since they are based on detecting anomalies and proactively notify them. Another important feature of SCEMMA is that the trading partner′s autonomy is preserved. To this, SCEMMAS was designed as collaborative inter-organizational information where the overall behavior emerges as a result of interactions between its components.

SCEMMAS is still in development, its goal is to serve as a test bed to conduct experiments in a laboratory environment. Future works contemplate the implementation of the PAGE and EVA total functionalities, the analysis of methods involved in the coordination mechanism, specifically in the generation of the solution space; and the incorporation of decision and control point in different levels and granularity. The last allows introducing new control criteria, due to new aspects are modeled with a hierarchy, but always maintaining the distributed and autonomous character.

References

[1] Bellifemine Fabio, Agostino Poggi, Giovanni Rimassa, CSELT internal technical report. Part of this report has been also published in Proceedings of PAAM'99, London, April 1999, pp.97-108.

[2] Bussmann, S.; Jennings, N. R. y Wooldridge, M. Multiagent Systems for Manufacturing Control: A Design Methodology. Springer Series on Agent Technology. T. Ishida, N.R. Jennings, K. Sycara (Series Eds.) Springer-Verlag. Berlin Heidelberg 2004. Alemania.

[3] Chopra, Sunil, and Meindl, *Supply Chain Management: Strategy, Planning, and Operations*, Upper Saddle River, NJ: Prentice-Hall, 2001.

[4] Christopher, Martin (2005). Logistics and Supply Chain Management. Creating Value-Adding Networks. Prentice Hall

[5] Hoffmann,O.; Deschner, D.; Reinheimer, S. and Bodendorf, F. Agent-Supported Information Retrieval in the Logistic Chain. In proceeding of the 32nd Hawaii International Conference on System Sciences, Maui, 1999.

[6] Kärkkäinen, M., Främling, K. And Ala-RiMKU. Integrating material and information flows using a distributed peer-to-peer information system. In Collaborative System Production Management (Jaddev H.S., Wortmann, J.C., Pets H.J. Eds), pp 305-319, Kuwer Academic Publishers, Boston, 2003.

[7] Knickle, Kimberly and Kemmeler Jennifer. *Supply Chain Event Management in the Field Success with Visibility*, AMR Research, Boston. Febrero 2002

[8] Masing, N. Supply Chain Event Management as Strategic Perspective – Market Study: SCEM Software Performance in the European Market. Master Thesis. Universitié du Québec en Outaouasis.

[9] Monostori L, E. Szelke, B. Kadar. Management of changes and disturbances in manufacturing systems. *Annual Reviews in Control 22 (1998) (85-97).*

[10] Pathak, S., Nordstrom G., Kurokawa S. Modeling the Supply Chain: a multiagent approach. *In proc. Of 2000 IEEE International Conference on System, Man and Cybernetic, vol 3* 2000, pp 2051-2056.

[11] Schoenthaler, R and Alvarenga Carlos. A New Take on Supply Chain Event Management. In Supply Chain Management Review. March/April 2003.

[12] Shapiro, J. *Modelling the Supply Chain*. Duxbury Press. 2000

[13] Smith, R. G. The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. *IEEE Trnasactions on Computers, C-29 (12), pages 1104-1113, 1980.*

[14] Swaminathan, J., Smith, S and Sadeh N. *Modeling Supply Chain Dynamics: A Multiagent Approach* Decision Sciences. Volume 29 Number 3 Summer 1998. Printed in the USA

[15] Szirbik, N.B., Wortmann, J.C., Hammer, D.K., Goosenaerts, J.B. M. and Aerts, A.T.M.. Mediating Negotations in a Virtual Enterprise via Mobile Agents. In Proceedings of the Academia/Industry Working Conference on Research Challenges (IEEE Computer Society Eds.), pp. 237-242, Buffalo, NY, 2000.

[16] Teuteberg F, Schreber D (2005) Mobile Computing and Auto-ID Technologies in Supply Chain Event Management – An Agent-Based Approach In Proceedings of the Thirteenth European Conference on Information Systems (Bartmann D, Rajola F, Kallinikos J, Avison D, Winter R, Ein-Dor P, Becker J, Bodendorf F, Weinhardt C eds.), Regensburg, Germany. (ISBN 3-937195-09-2)

[17] Wyns, J.. Reference Architecture for Holonic Manufacturing Systems. PhD- Thesis. Katholieke Universiteit Leuven. Belgium, 1999.

[18] Zimmermann, R. Agent-based Supply Network Event Management. Whitestein Series in Software Agent Techonologies, Eds: Marius Walliser, Stefan Brantschen, Monique Calisti y Thomas Hempfling, 2006.