MULTIAGENT SYSTEMS FOR SHOP FLOOR ARHITECTURE MANAGEMENT

Dan FLOROIAN
Transilvania University of Brasov, Romania
Faculty of Electrical Engineering and Computer Science
d.floroian@ieee.org

Abstract: *The paper presents the problem of shop floor agility. In order to cope with the disturbances and uncertainties that characterise the current business scenarios faced by manufacturing companies, the capability of their shop floors needs to be improved quickly, such that these shop floors may be adapted, changed or become easily modifiable (shop floor reengineering). One of the critical elements in any shop floor reengineering process is the way the control/supervision architecture is changed or modified to accommodate for the new process and equipment. This paper, therefore, proposes an multi-agent architecture to support the fast adaptation or changes in the control/supervision architecture.*

Keywords: multi-agent system, shop floor agility, control/supervision architecture, virtual organisation.

JEL Classification: M_{15}, O_{32}.

1. Introduction

Shop floor agility is a central problem in current manufacturing companies. Internal and external constraints, such as growing number of product variants and volatile markets, are changing the way these companies operate by requiring continuous adaptations or reconfigurations of their shop floors. This need for continuous shop floor changes is so important that finding a solution to this problem would offer a competitive advantage to contemporary manufacturing companies (Barata, J., Camarinha-Matos, L. M., 2002; Floroian, D., 2008).

The central issue is, therefore, which techniques, methods, and tools are appropriate to address shop floors whose life cycles are no more static but show high level of dynamics. In other words, how to make the process of changing and adapting the shop floor fast, cost effective, and easy. The long history of industrial systems automation shows that the problem of developing and maintaining agile shop floors cannot be solved without an integrated view, which accommodate the different perspectives and actors involved in the various phases of the life cycle of these systems. Moreover, supporting methods and tools should be designed and developed to accommodate the continuous evolution of the manufacturing systems along their life cycle phases (Camarinha-Matos, L. M., 2002; http://protege.stanford. edu).

Agility is a fundamental requirement for modern manufacturing companies in order to face challenges provoked by the globalisation, changes on environment and working conditions regulations, improved standards for quality, fast technological mutation, and changes of the production paradigms. The turbulent and continuous market changes have impacts at different levels, from company management to shop floor. Only companies that exhibit highly adaptable structures and processes can cope with such harsh environments. Furthermore, the capability to rapidly change the shop floor infrastructure is a fundamental condition to allow participation of manufacturing enterprises in dynamic cooperative networks. Networked enterprise associations, such as virtual enterprises, advanced supply chains, etc. are examples of cooperative structures created to cope with the mentioned aspects. Manufacturing companies wishing to join these networked structures need to be highly adaptable in order to cope with the requirements imposed by very dynamic and unpredictable changes. On the other hand, agility corresponds to operating efficiently but in a competitive environment dominated by change and uncertainty, which means adaptation to conditions that are not determined or foreseen a-priori. The participation in dynamic (and temporary) organisations requires agile adaptation of the enterprise to each new business scenario, namely in terms of its manufacturing capabilities, processes, capacities, etc. (Barata, J., Camarinha-Matos, L. M., 2002; Camarinha-Matos, L. M., 2002; Floroian, D., 2009).

Addressing this need, an multi-agent architecture is proposed to support the fast adaptation of agile shop floor control systems. According to this approach manufacturing systems are no more than compositions of modularised manufacturing components whose interactions, when aggregated, are governed by contractual mechanisms that favour configuration over reprogramming.

2. Supporting Concepts

Multi-agents systems represent a relatively new area in computer science, which started to be developed in the 1980s but that only in the mid 1990s gained widespread interest. Multi-agent systems are compositions of computing elements that possess autonomous action, and which are able to interact among themselves, not only for exchanging messages but also for a more elaborated kind of communication that resembles social activity: cooperation, coordination, negotiation (Camarinha-Matos, L. M., 2002; Floroian, D., 2009).

The multi-agent area can be organised into two different sub areas. The first is concerned with the individual agent (micro level), while the second deals with the way these agents form societies of interacting agents (macro or group level). Using an analogy with humans beings, the first one is concerned with the individual human being, while the second one deals with the society in which humans live (Floroian, D., 2008).

An agent is a computer-based program that is situated in some environment and that is able to take autonomous actions in this environment, based on its goals and perceptions from that environment, in order to meet its design objectives.

These programs must thus be able to interact with the environment (external world) through some actions and must be able to make decisions based on the perceptions they obtain from that world. Agents should never assume they have complete control or knowledge over the world (the world is not deterministic), and perceptions must be considered as giving only partial knowledge. Consequently, agents have to be prepared for the possibility of failure. Agents know about the world when they receive messages and influence the world by sending messages.

The agent and object oriented paradigms are so often confused, when in fact they are different, that certain points need to be highlighted. People confuse both paradigms because an object is an encapsulation of some abstract concept, which can also be the case for an agent (http://sharon.cselt.it/projects/ jade/ ; http://protege.stanford. edu).

To clarify better how the subsumption architecture operates, an example of a mobile robot that goes from a start position to a destination point with some obstacles in its way is considered (Figure 1). The robot knows the direction of the destination place because a sensor in the robot can detect the emission of a beacon. The control logic of this sensor gives, when requested, the azimuth of the beacon in relation to the longitudinal line of the robot. Since the destination point is marked by a dark area, which can be sensed by another sensor of the robot, it is possible to know when the destination point is reached. The robot also has sensors to detect obstacles (Moldoveanu, F., Comnac, V., Floroian, D., Boldişor, C., 2005).

This robot is composed of three hierarchical behaviours that run concurrently and asynchronously. The higher-level task is GoalOriented, which guides the robot to its destination. This task is continuously running and, for each interaction, the azimuth sensor is read and a function to align the robot with the read azimuth is called. Simultaneously, the behaviour AvoidObstacles is also running as well as the lowest level one Stop. If an obstacle is found while the robot is moving, a racing condition happens because GoalOriented tries to keep the alignment with the beacon while AvoidObstacles try to avoid it and the actions might be in conflict. Under this condition the subsumption architecture determines that the low level behaviour has precedence over the higher level one. This is why the behaviour AvoidObstacles inhibits the higher level one. After the obstacle avoidance, the higher-level behaviour is free to run again (instruction run (GoalOriented)). When the robot reaches the destination point, the race is between the three behaviours. In this case the lower level behaviour (Stop) takes precedence and the robot is stopped.

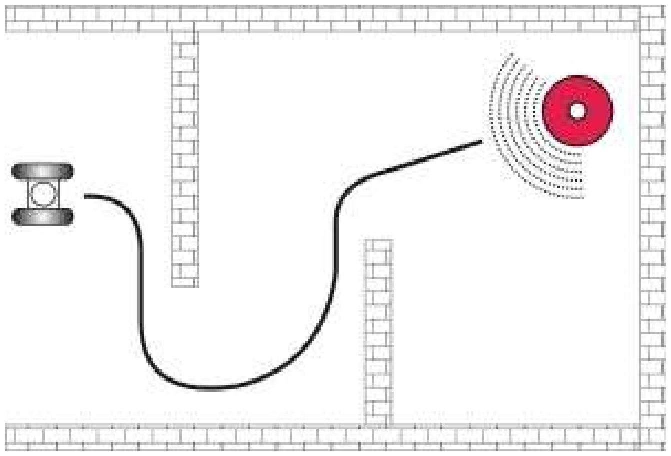


Fig. 1. Mobile robot

2.1. Agent Communication Languages

The study of the animal world (humans included) has shown that the most complex, long-lived communities are those that have developed complex communication mechanisms that allow the establishment of complex interactions. Human language is, in fact, one of the most complex communication mechanisms and, unsurprisingly, humans were able to develop the most complex societies in the animal world. Considering these aspects, it seems unquestionable that complex agent societies can only be created with the aid of Agent Communication Languages that, through interaction, allow the creation of agent communities that tackle problems an individual agent could never handle. ACLs should be public to permit the conversation between a large number of agents.

Agents engaged in negotiations need ACLs that support more than just single message exchanges. These negotiations are supported by agent engagement in conversations that are task-oriented, shared sequences of messages that they follow. These conversations are also known as pre-arranged coordination protocols. Well-defined and sharable conversation protocols can be used to coordinate agents that attempt to accomplish specific tasks.

2.2. Coalitions

Coalitions are discussed in the context of societies of agents because they are an important method for achieving cooperation in multi-agent systems, even with self-interested agents, which can increase their ability to satisfy their goals and maximise their own personal payoffs. Coalitions and negotiation are related because it is not possible to create alliances or coalitions without involving the potential partners in a negotiation process. Only after the partners reach a common understanding is it possible to claim that a coalition or alliance exists (Floroian, D., 2008; Floroian, D., 2009).

2.3. Ontologies

The work on ontologies was mainly motivated by the need for sharable and reusable knowledge bases. An ontology produces a common language for sharing and reusing knowledge about phenomena in a particular domain.

An ontology is a branch of philosophy dealing with the order and structure of reality. In terms of computer science it is a simplified view of a particular subject area using a formal abstract model composed of the concepts and their interrelationships, which characterise that area. A few years ago researchers, if asked about the meaning of ontologies would reply that it was an esoteric field in philosophy that studies beings or what type of things exist. However, nowadays a simple Internet search for this term will give thousands of hits and, among the first hits, it is possible to find web pages containing terms such as "web semantics", "enterprise ontology", "virtual enterprise", which are more practical.

2.4. Contracts

Contracts are used in human societies to shape behaviour, secure rights, and protect liberties. They can be used in the same way in artificial agents societies (Floroian, D., 2008).

A contract might be terminated either under normal, or abnormal conditions. Contracts that terminate under normal conditions are known as terminated by discharge, which may occur under the following situations:

By performance – This is what happens when a contract reaches its validity. The contract terminates naturally by being fully performed by the involved parties.

By agreement – This is what happens when the contract was not fully performed but all the involved parties agree to terminate it.

By operation of law – A contract is frustrated where, after the contract was concluded, events occur which make performance of the contract impossible, illegal or something radically different from that which was in the contemplation of the parties at the time they entered into the contract.

2.5. Electronic Contracts – state of the art

The driving force behind electronic contracts is business, and in particular the area of electronic commerce. Therefore, almost all the work on electronic contracts comes from researchers in this field.

2.6. Collaborative networked organisation

A collaborative networked organisation is any group of autonomous entities, which may be organisations, people, or artificial agents that have together formed a cooperative dynamic network to reach individual or group benefits.

A cluster represents a group or pool of enterprises and related supporting institutions that have both the potential and the will to cooperate with each other through the establishment of a long-term cooperation agreement.

2.7. Shop-floor management

Initially the automation of production lines was achieved using only purely mechanical solutions. As would be expected, mechanical solutions have some limitations in terms of being able to control complex situations, and even in the cases where this is possible, such solutions require much maintenance, are complex to build, subject to wear, and any required change implies complete rebuilding of the system. Mechanical solutions are very inflexible whenever changes are required. Pneumatic control, which is still very popular for certain applications, uses compressed air, valves and switches to construct simple control logic. Despite exhibiting slow response times, it is easier to build than mechanical control because it is possible to construct logic functions using standardised components. However, reprogramming is difficult and time-consuming since it requires rewiring air ducts.

Before the advent of computers the creation of electromechanical control solutions became possible. An electromechanical control uses switches, relays, timers, counters, etc, to build control logic. Since electrical signals move faster than air, this type of control was faster than pneumatic control. Although far from the flexibility of today's computerised solutions, it is much more flexible than mechanical and pneumatic control because it is possible to build more complex logic and it is also easier to rewire electrical cables than air ducts.

Even if mechanical and electromechanical control solutions could only achieve poor flexibility, and required fairly high skills, this was not very problematic in the era of mass production because the goal was to rapidly produce as many pieces as possible, without changing the product. This signified that few control alterations were needed. However, this situation changed when the demand for more product variety started, and then these rigid (fixed) control solutions were no longer effective. Despite this need, it was only possible to create more flexible control solutions with the advent of computers because they are inherently flexible. With them it is possible to create a control sequence and, if not appropriate, that sequence may be completely altered just by software (programming). This was a revolution in terms of production systems design, and it was the most important enabler to build production systems able to cope with diverse products, i.e. flexible production systems.

The main problems with the control and supervision architecture is how to integrate the various heterogeneous controllers that can be found on the shop floor, how to supervise and synchronise the various tasks, and how to develop a strategy to facilitate the plug and unplug of equipment. Much research was done in the 1980s and early 1990s but none resulted in commercially adaptable solutions. The problem is not with the flexibility of the individual machines, namely CNCs and robots that are reprogrammable, but with the system considered as a whole (integration). The complexity of the shop floor and the diversity of applications and operational requirements just increase the difficulty of integrating them because the system becomes difficult to understand, generalise, and standardise.

Being able to develop a control and supervision architecture whose programs could be independent of the process has been the dream of shop floor engineers ever since. In fact, this dependence is a source of inflexibility because whenever a machine (resource) is changed, the product is changed, or the process plan is changed, it is immediately necessary to change the programs of the individual components.

3. System Architecture to Support Shop Floor Reengineering

In the manufacturing shop floor the manufacturing components, which are controlled by a diversity of controllers and correspond to companies in the Virtual Enterprise world, are the basic set from which everything is built up. A shop floor can be seen as a micro-society, made up of manufacturing components. The components have basic core capabilities or core competencies (skills) and, through cooperation, can build new capabilities. A robot, for instance, is capable of moving its tool centre point and setting different values for speed and acceleration. Its core competencies are represented in Figure 2. A gripper tool, on the other hand, has as basic skills the capability to close or open its jaws. These two components when acting alone can only perform their core skills. However, when they cooperate, it is possible to have a pick-and-place operation that is a composition of the move with the open and close skills. The greater the diversity and complexity of individual capabilities, the greater are the chances of building more complex capabilities (Floroian, D., 2008).

A manufacturing component is a physical piece of equipment that can perform a set of specific functions or basic production actions on the shop floor such as moving, trans-forming, fixing or grabbing.

An agentified manufacturing component is composed of a manufacturing component and the agent that represents it. The agent's skills are those offered by the manufacturing component, which is connected to the agent through middleware.

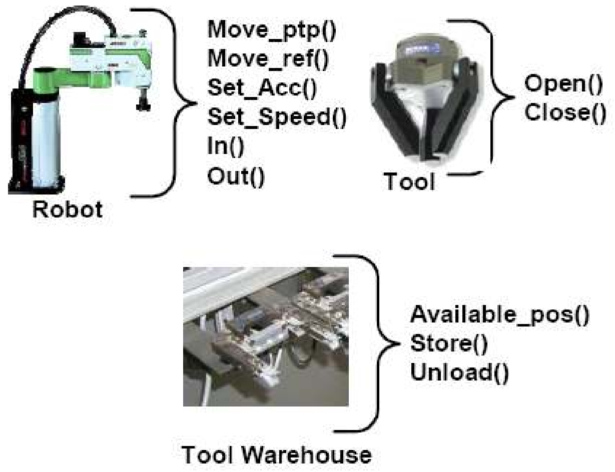


Fig. 2. Basic manufacturing components

A coalition/consortium is an aggregated group of agent manufacturing components, whose cooperation is regulated by a coalition contract, interacting in order to generate aggregated functionalities that, in some cases, are more complex than the simple addition of their individual capabilities.

A shop floor cluster is a group of agent manufacturing components which can participate in coalitions and share some relationships, like belonging to the same manufacturing structure and possessing some form of technological compatibility.

A community of agents belonging to the same physical structure – a manufacturing cell, thus forms a cluster, and when a business opportunity (i.e. a task to be executed by the shop-floor) arises, those agents with the required capabilities (skills and capacities) and compatibility are chosen to participate in a coalition. The limitation for an agent manufacturing component to be accepted in a shop floor cluster is that it must be compatible with the others physically installed in the cell. For instance, an agent robot installed far from a cell is not a good candidate to join the cluster that represents that cell, because it can never participate in any coalition. Since all the manufacturing components installed in a cell answer the requirements for compatibility a shop floor cluster is associated with a physical cell.

Figure 3 shows how manufacturing agents, cluster, and coalition interrelate. Agentified components in the same "geographical" area of the shop-floor join the same cluster. The different coalitions that can be created out of a cluster represent the different ways of exploiting/operating a manufacturing sys-tem. Adding or removing a component from the physical manufacturing system also implies that the corresponding agent must be removed from the cluster, which can also have an impact on the established coalitions. A broker is used to help the formation of coalitions to reduce the complexity of the individual agents in terms of coalition formation. By delegating this responsibility to the broker, the individual agents can be simpler because all they have to do is negotiate the terms of their participation with the broker rather than carrying out all complex details of coalition formation such as deciding which members are better indicated to answer the requirements of a coalition being formed.

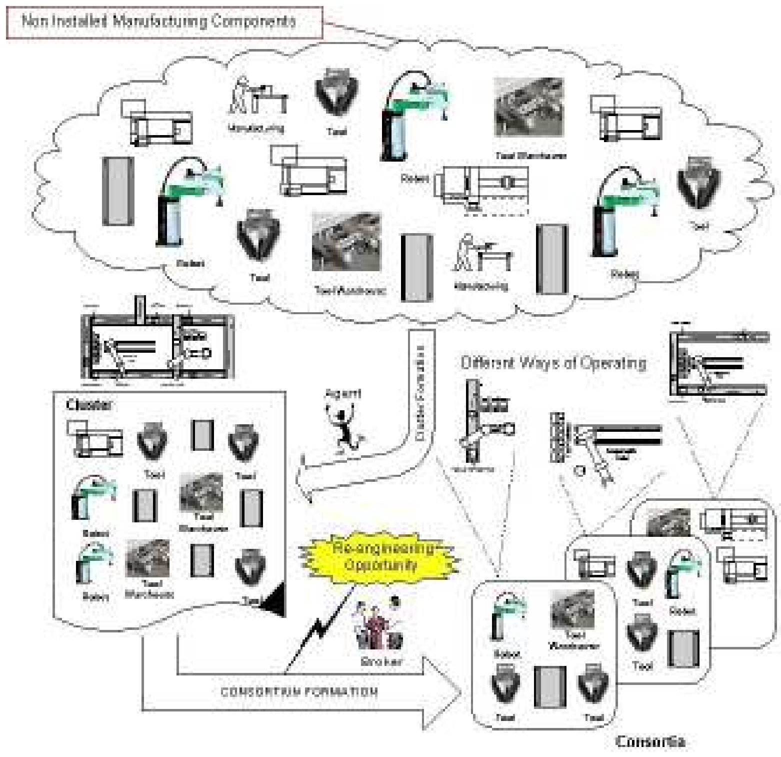

Non Installed Manufacturing Components
Tool
Robot
Tool Warehouse
Different Ways of Operating
Agent
Cluster Formation
Cluster
Re-engineering Opportunity
Broker
CONSORTIUM FORMATION
Consortia

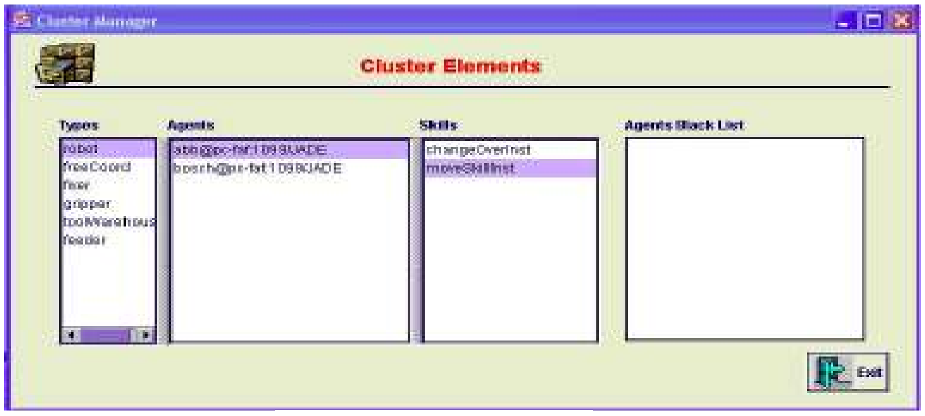
Fig. 3. Consortia formation

The interactions between the cluster and its members are regulated by a contract. This contract establishes the terms under which the cooperation is established. It includes terms such as the ontologies that must be used by the candidate, the duration, the consideration (a law term that describes what the candidate should give in exchange for joining the cluster, usually the skills that the candidate is bringing to the cluster). The behaviour of a coalition is regulated by another contract that is "signed" by all its members. The important terms of this type of contract, other than the usual ones like duration, names of the members, penalties, etc., are the consideration and the individual skills that each member brings to the coalition. The importance of contracts as a mechanism to create/change flexible and agile control structures (consortia) lays in the fact that the generic behaviours presented by generic agents are constrained by the contracts that each agent has signed. This calls forth the idea that different coalition behaviours can be achieved by just changing the terms of the coalition contract, namely the skills brought to the coalition.

The expectation at this point is that coalitions of agentified manufacturing components, if regulated by contracts, that are declarative and configurable information structures, may lead to significantly more agile manufacturing systems. It is expected that the different ways of exploiting a system depend only on how coalitions are organised and managed. This approach solves the problem of how to create dynamic (agile) structures, but not the problem of how to integrate heterogeneous manufacturing components' local controllers. In order to overcome this difficulty, the process used to transform a manufacturing component into an agent (agentification) follows a methodology to allow their integration.

4. Implementation

The cluster user interface is shown in Figure 4. The interface shows all the agent types that are registered. When an agent type is selected the addresses of all the agents that belong to that type are shown. In the same way, selecting an address implies that the corresponding skills of that agent are shown. The cluster interface also shows the agents in the black list. The figure 5 illustrates the window corresponding to the membership contract.


Cluster Manager

Cluster Elements

Types	Agents	Skills	Agents Black List
robot	abb@pc-fat1099JADE	changeOverlist	
freeCoord	bosch@pc-fat1099JADE	moveSkillist	
fixer			
gripper			
toolWarehous			
feeder			

Fig. 4. User interface

The user interface depicted in Figure 6 is used for coalition contracts alterations. As it can be observed, all the coalition leader agents are listed. By selecting one leader the respective coalition contract appears in the right hand side of the window.

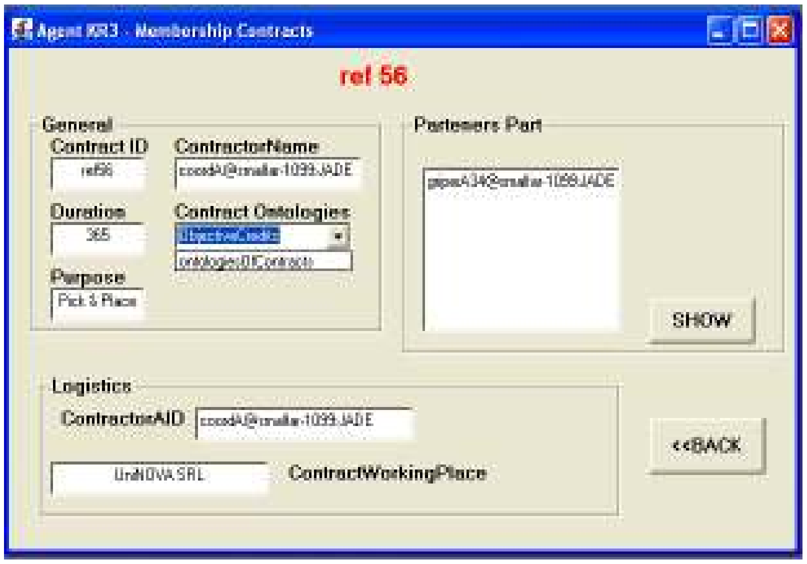


Fig. 5. Membership contract

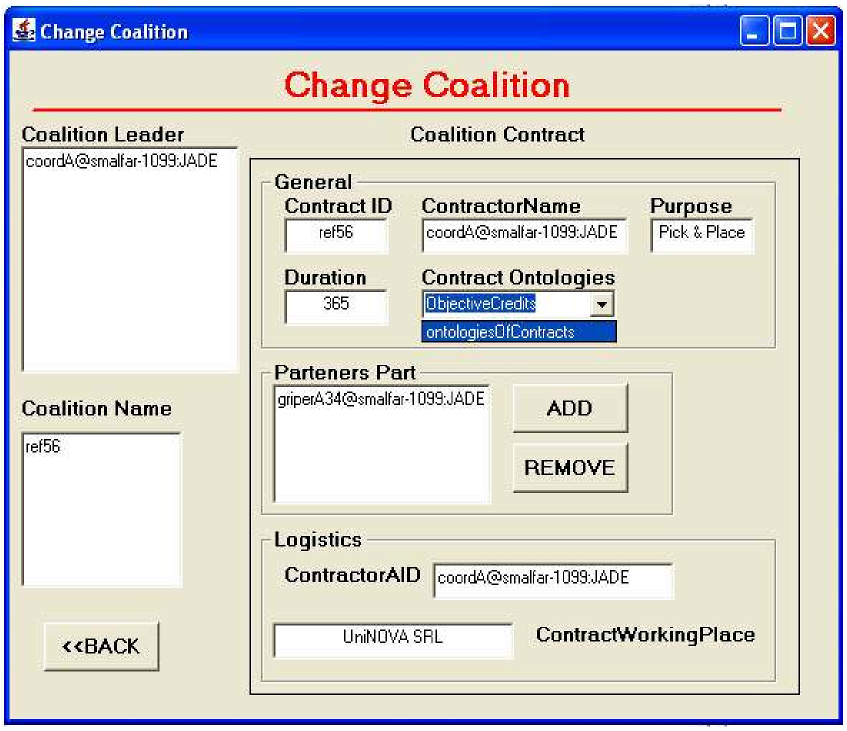


Fig. 6. Change coalition window

The flexible manufacturing system (Novaflex) resembles a real manufacturing system composed of industrial components. A simplified diagram of part of the Novaflex is illustrated in Figure 7, while Figure 8 shows a partial picture.

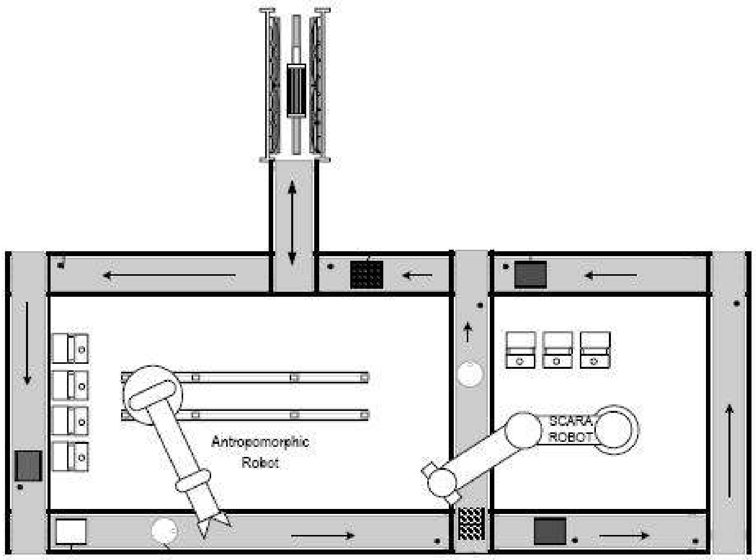


Fig. 7. Nova Flex

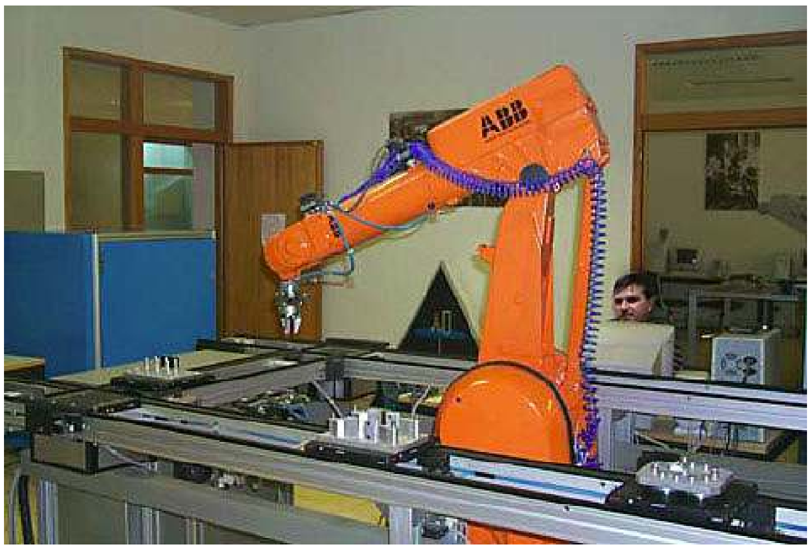


Fig. 8. Production line

5. Remarks

The design and development of such a shop floor reengineering architecture, targetting real shop floor applications, required the study and analysis of a wide range of concepts and supporting technologies, which were detailed to enable the reader to better understand the concepts and technologies. The proposed architecture, featuring coalitions of agentified manufacturing components, whose members are chosen from an cluster that contains all the manufacturing components of a given cell, using a broker agent as an intermediary, proved to be an adequate solution for the agile shop floor problem.

Contracts inspired on legal principles have been used to govern the relationships between autonomous agents. By using this approach, the interactions are constrained by what the contract says and not by the individual program of each interacting agent. As long as the agents are able to read and understand the role of contracts they can be generically involved in any relationship constrained by contracts.

References

1. Barata, J., Camarinha-Matos, L. M., (2002), *Shop Floor Re-engineering Using Agents*, In: Proc. of ISR2002 3rd Int. Symp. on Robotics, Stockholm
2. Camarinha-Matos, L. M., (2002), *Virtual Organisations in Manufacturing*, In: Proc. of FAIM, 12th Int. Conf. on Flexible Automation and Intelligent Manufacturing, Munich
3. Floroian, D., (2009), *Sisteme multiagent*, Editura Albastră, Cluj-Napoca
4. Floroian, D., (2008), *Contributions on re-enginering of control and supervision manufacturing systems using a multiagent based architecture*, PhD. Thesis, *Transilvania* University of Brasov
5. Moldoveanu, F., Comnac, V., Floroian, D., Boldişor, C., (2005), *Trajectory Tracking Control of a Two-link Robot Manipulator Using Variable Structure System Theory*, In: "Control Engineering and Applied Informatics (CEAI) Journal", Published by the Romanian Society of Control Engineering and Technical Informatics, Sept., Vol. 7, No. 3
6. Protégé-2000: http://protege.stanford. edu [web site]. Retrieved Jan 2006 from the World Wide Web, 2006
7. FIPA: *FIPA Communicative Act Library Specification (XC00037J)*. Geneve: FIPA - Foundation For Intelligent Physical Agents, 2006
8. JADE: *http://sharon.cselt.it/projects/ jade/* [web site]. Retrieved Jan 2008, from the World Wide Web