Intelligent Reconfiguration Using Knowledge Based Agent System

Munir Merdan; Ivanka Terzic; Alois Zoitl & Bernard Favre-Bulle
Automation and Control Institute
Faculty of Electrical Engineering and Information Technology
Vienna University of Technology, Gusshausstrasse 27-29/E 376, 1040 Vienna, Austria
[merdan, terzic, zoitl, favre@acin.tuwien.ac.at]

Abstract

The new reconfigurable manufacturing paradigm requires development of systematic methods, and software tools, for the rapid design and build up of production systems that change the capacity and functionality in response to the market demand. The effectiveness of reconfiguration will depend upon the development of such tools.

In this work, the new concept consisting of ontology knowledge based systems in connection with software agents will be applied for the solution of the reconfiguration problem. Reconfigurability will be achieved by the employment of the most modern technologies such as ontology driven solution.

1.Introduction

Future manufacturing companies need to effectively react to sudden changes in customer demands, constant evolution of technology, and unpredictable events such as failures and disruptions. Production systems must show more flexibility, more robustness and a higher degree of reusability for hard and software modules in order that these basic conditions can be fulfilled. Agent-oriented techniques offer a new control approach that, using concepts of autonomy and cooperation, leads to more flexible and robust production systems. The reconfiguration ability is one of the fundamental problems in modern production systems. The purpose of reconfiguration is to allow a manufacturing system to change rapidly and cost-effectively from its current configuration to another configuration without being taken off-line maintaining system effectiveness when product or production changes or breakdowns occur [7], [19], [20].

Our aim is to achieve the reconfigurability by means of autonomous agents.

This work studies the transformation of a particular ontology from manufacturing domain into the form suitable for communication with software agents. Such ontology based system can be used e.g. as translators between two differently working machines. A key feature of the agent model is that agents must have access to various kinds of expertise. Agents advertise their services and process requests either by making inferences based on local knowledge, by routing the request to a more appropriate agent, or by decomposing the request into a collection of sub-requests and then routing these requests to the appropriate agents and integrating the results.

From a multi-agent systems perspective agents that have access to an ontology-backed semantic description of the behaviors and services from which they are composed, should be better able to proactively coordinate themselves [21].

The integration of ontologies used in the area of multi agent system for manufacturing with those developed for the semantic web is a key strategic issue for successful development of future production systems [8].

This paper researchs how software agents and ontologies can together provide breakthrough technologies for industrial applications.

Multi-agent systems have been used in several projects and models for improving knowledge management. Examples are InfoSleuth [1], Transaction-Oriented Computation Models [2], AALAADIN [4], DESIRE [18] and A.O.R [17].

The paper is structured as follows. The ontology and the knowledge base are described in the second section. The third section gives an overview of the agents and agents communication. The forth section describes system architecture implementation. Finally, section five gives the conclusion and future work.

2.Ontology and knowledge base

2.1 Ontology

Ontologies play significant role in knowledge sharing, use, and communication. Ontologies are used to specify the infrastructure in which the various agents operate, i.e., the information manipulated by the various agents and the relationships between them. This enables decisions on which agents to route the various requests to. That means the concepts and relationships that can exist for an agent or a group of agents are descripted by ontology [9]. Only what can be represented using ontologies can be represented in agent's knowledge bases [8]. For our purposes, it is a specification of the software agent and relationships in their area of interest, i.e. the agent's world. It consists of a structured set of concepts related by typed relations such as classification

0-7803-9402-X/05/$20.00 © 2005 IEEE

(is_a), composition (composed_of), instantiation (instance_of): O = (C , R) where C is a set of concepts and R a set of relations [22].

Ontologies give a uniform and declarative description of semantic information. To be semantic effective, it must be employed consistently and predictably across an agent system, with respect to how individual agents react when a task fails or is canceled, and how the agents in the system interact when problems develop with a delegated task.

Ontologies are developed and used because they enable among others [8]:

- to share knowledge → by sharing the understanding of the structure of information shared among software agents and people
- to reuse knowledge → ontology can be reused for other systems operating in a similar domain
- to make assumption about a domain explicit → e.g. for easier communication

The semantic interoperability (a possibility to understand shared data, information, and knowledge) is one of the main reasons why ontologies are being used. There are two distinct problems/tasks for an ontology based model: one is the extraction of semantic concepts from the keywords and the other is the actual construction of the ontology for a multi agent system architecture. With regard to the first problem, the key issue is to identify appropriate concepts for the problem domain. For the second problem, we would like to generate ontology, in order to make our approach scalable, as shown in other works.

2.2 Knowledge base

The knowledge base is a "large collection of facts, rules, and heuristics that capture knowledge about a specific domain of applications" [15]. Knowledge base is typical examples of structured information sources [14].

To build up an applicable knowledge base, the following steps are required:

- modeling the agent knowledge domain,
- designing and employing a persistent knowledge base,
- bridging that domain model to the knowledge base with an extensible domain interchange grammar,
- develop a process (methodology) for structuring a knowledge base containing the domain knowledge model.

3. Agent communication

According with Wooldridge and Jennings [12], each agent consists of five basic blocks: local data, processing unit, control unit, a buffer and an interface. The local data is used as a memory to acquire the knowledge for software agent to act as seller and buyer, using a bilateral communication. The processing unit allows successful achievement or accomplishment of agent's particular goals or performing of particular tasks.

Due to control unit agents interact by making reciprocal commitments and carry out particular tasks. A buffer is necessary to handle processing data. An interface characterizes every agent for the management of communication protocol [5].

The main task of the agents is to gather as much as possible knowledge about the system. Once the knowledge is gathered, it is shared at the controlling agent-broker and the broker takes decisions about the system after clustering and analyzing the data.

The agents share the same base structure so they inherit common communication and knowledge management methods. Agents advertise their skills during the cooperation protocol so there is no static representation of agent capabilities in our system.

In general, communication can be synchronous or asynchronous, and the communication mode can be point-to-point (between two agents), broadcast (one to all agents), or multicast (to a selected group of agents) [13]. Our environment takes KQML (The Knowledge Query and Manipulation Language) as communication language for the interaction among agents. The KQML language will allow the controlling agent-broker to send unsolved subgoals to other agents. We adopt broker architecture to allow an agent to locate other agents with the desired expertise. The Knowledge Query and Manipulation Language (KQML) is a language and a set of protocols that support software agents in identifying, connecting with and exchanging information with other agents.

Our system depicts client server architecture, and communication is via TCP/IP sockets. Implementation of the system is on a java platform using java socket programming for communication.

4. System architecture

We propose an agent architecture that enables cooperation among a set of autonomous and heterogeneous units. In the multi agent system each agent is an autonomous semantic entity responsible for the commitment of the local data described in its knowledge base. Our agent architecture is presented as a group of agents with different intent, background knowledge that work together. A sentinel is an agent that may interact with any other agent. Its purpose is to guard certain functionality and prevent the system from entering undesired states.

Our Testbed architecture consists of an automatic storage system with a handling unit for the extraction of the parts, a pallet transfer system with redundant paths, as well as a portal robot for the final assembly.

The composits parts of the pallet transfer system are:

- 6 switch units

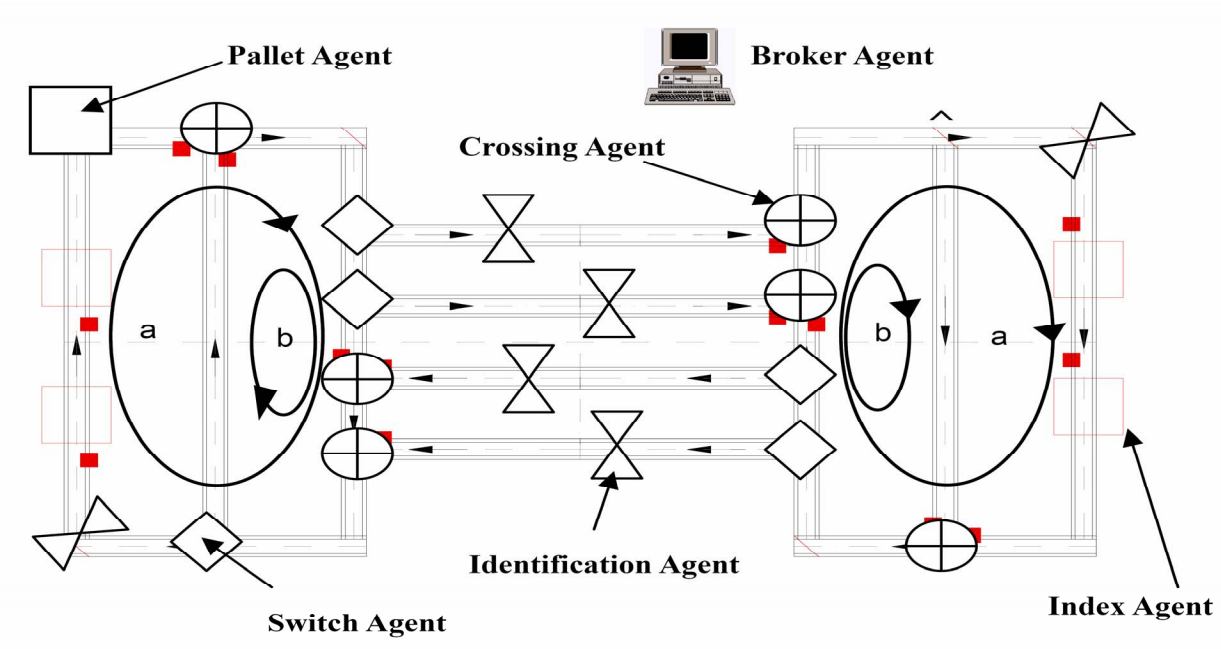


Figure 1. Structure of the transfer system.

- 6 crossing units
- 4 index units, that fixes the pallet in defined position for handling unit (loading/unloading).
- 6 identification units for identification of passing pallet units

In our work each of these units is regarded as a software agent [3]. The structure of the transfer system is represented in figure 1.

These architecture applies the "broker agent concept", where the specialized broker agents semantically match information needs (specifed in terms of the ontology) with currently available resources, so retrieval and update requests can be routed only to the relevant resources.

Software agents in a distributed architecture share knowledge describing how they can be reached (via network address, port, and protocol). They also know the area of expertise of each individual agent. They inform the broker about the knowledge, which they possess. The agents send also periodically summarized information about their current area of interest to the broker. The broker analyzes this information in order to produce better results. If an intelligent agent needs information, which he does not have, he sends a query to the broker. The broker, who administers a register of the fields of knowledge (domain) of the individual agents, refers then the question to the agent, who could have sufficient expertise for answering the question. The answer is commonly simply a partial result. The agent includes the result into its own proof tree just as if it had done the work itself, and continues its task.

An example application for the transport system would be the computation/negotiation of the best way routes of the transportation pallets.

Each agent in the system gives appropriate information to the broker agent. A pallet would like to arrive from an assembly station to another. It wants to use the shortest distance and has the largest priority. Such a task required from the pallet specialized knowledge that it does not have. The pallet communicates to the broker agent its question. The broker passes it on among other things to the next crossing and to the next switch. The switch has the necessary specialized knowledge over the direction steering and distance to the next switch. It knows, which pallet has to be sent in which direction. The crossing has the specialized knowledge of priorities, i.e. which pallet with which priority have to be transferred in the output channel. Due to the answers the pallet will arrive at its destination (fig. 2).

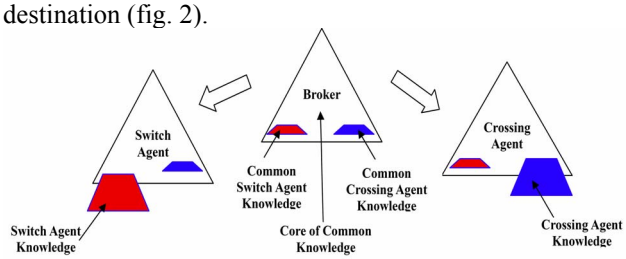


Figure 2. Distributed artificial inteligence arcitecture

5. Conclusion and future work

We have described an approach to increase robustness in a multi-agent system. In order to realize reconfigurability of our system architecture we present an ontology driven solution. The agents that have

sufficient expertise, the information resources, and the system itself cooperate to address the information processing requirements of the users, allowing for easy, dynamic reconfiguration of the system.

With the planned experimental architecture the following characteristics of the knowledge-based agent systems can be tested:

- Reconfiguration: Agents should have the ability of adaptation on current system conditions i.e. the percipience of environment and the reaction on occured changes. e.g.: failure of transport system components
- Cooperation: Agents should be able to interact with their environment. The interaction can take place with other agents or with a user. A goal of the interaction is to help the agent to solve the problem or to support other agents by the solution of their problem.
- Autonomy: An agent should be able to act, without direct interaction of the user, or another agent. Further an agent has to possess a control of its own actions and internal status.

The advantage of our concept is that in this knowledge-based system it does not have to be said, how a problem has to be solved (i.e. which details have to be done), but the problem and the goal have to be described. The system decides on its own how to achieve the goal.

Our approach can be used as a basis for further research the self-organization and the self-learning of the individual autonomous components in a transport system.

Reference

[1] BAYARDO R.J. & AL, "InfoSleuth : Agent-based Semantic Integration of information in Open and Dynamic Environments", in Proceedings of the 26th SIGMOD Int'l Conference on Management of Data, Tucson, Arizona, May (1997).

[2] Unruh, A., Bailey, J., Ramamohanarao, K., "A framework for goal-based semantic compensation in agent systems"1st International Workshop on Safety and Security in Multi-Agent Systems, AAMAS '04.

[3] Dutzler C., Glanzer K., Zeichen G, „Hochflexible Steuerungsarchitektur für Montageprozesse auf Basis verteilter holonischer Systeme", Tagungsband zur SPS/IPC/Drives 2002, Seite 203-211, Hüthig Verlag, Heidelberg, ISBN 3-7785-2863-7

[4] [Ferber and Gutknecht, 1998] Ferber J., Gutknecht O., "A meta-model for the analysis and design of organizations in multi-agent systems",IEEE Computer Society, Proc. 3rd ICMAS, p. 128-135, 1998.

[5] V. Di Lecce, C. Pasquale & V. Piuri "A Basic Ontology for Multi Agent System Communication in an Environmental", C1MSA 2004 - International Conference on Computational Intelligence for Measurement Systems and Applications Boston. MA, USA, 14-16 JULY 2004

[6] L. K. Wickramasinghe, L.D. Alahakoon," Discovery and Sharing of Knowledge with Self-Organized Agents", Proceedings of the IEEE/WIC International Conference on Intelligent Agent Technology (IAT'03), 2003 IEEE

[7] Kordic V.; Lazinica A. & Merdan M. (2005). "Future of Manufacturing: Concepts of Autonomy and Selforganisation", *International Journal of Advanced Robotic Systems*, ISSN 1729-8806, Volume 2, Number 1, March 2005, pp.12

[8] Marek Obitko, Vladimır Marık, "Ontologies for Multi-Agent Systems in Manufacturing Domain", Proceedings of the 13th International Workshop on Database and Expert Systems Applications (DEXA'02), 2002 IEEE

[9] T. R. Gruber, "A Translation Approach to Portable Ontology Specifications", Knowledge Acquisition,

[10] Marc J. Raphael & Scott A. DeLoach, "A Knowledge Base For Knowledge-Based Multiagent SystemConstruction", National Aerospace and Electronics Conference (NAECON) Dayton, OH, October 10-12, 2000.

[11] http://www.cs.utexas.edu/users/mfkb/related.html

[12] M. Wooldridge and N. R. Jennings " Pitfalls of agent-oriented development," Proceeding of the second international conference an Autonomous agents, pages 385-391,1998.

[13] Liu Honga, Cui Weib, "Supporting Design in An Agent Based Collaborative Environment", The 8th International conference on Computer Supported Cooperative Work in Design Proceedings, 2003 IEEE.

[14] Marc J Raphael, Captain, "Knowledge Base Support for Design and Synthesis of Multi-Agent Systems", Thesis, Department of the Air Force, Air University, Air Force Institute of Technology, Wright-Patterson Air Force Base, Ohio

[15] J. W. a. C. T. Schmidt, "Preface" Foundations of Knowledge Base Management, Springer-Verlag, Harrisburg, VA, VII-IX, 1989

[16] J. Heaton, "Programming Spiders, Bots, and Aggregators in Java", Richard Mills, Marina Village Parkway, Alameda, 2002.

[17] Wagner G., Agent-Oriented Analysis and Design of Organizational Information Systems, Proceedings of Fourth IEEE International Baltic Workshop on Databases and Information Systems, May 2000, Vilnius (Lithuania).

[18] Brazier, F.M.T., Dunin Keplicz, B., Jennings, N., and Treur, J., DESIRE: Modelling Multi-Agent Systems in a Compositional Formal Framework. International Journal of Cooperative Information Systems, vol. 6, Special Issue on Formal Methods in Cooperative Information Systems: Multi-Agent Systems, (M. Huhns and M. Singh, eds.), 1997, pp. 67-94..

[19] Mehrabi, M.G., Ulsoy, A.G., and Y. Koren, 1998-b, "Reconfigurable Manufacturing Systems: Key to Future Manufacturing," Proceedings of the 1998 Japan-USA Symposium on Flexible Automation, pp. 677-682, Otsu, Japan.

[20] Lee, G.H., 1997, "Reconfigurability Consideration Design of Components and Manufacturing Systems," Int. Journal of Advanced Manufacturing Technology, Vol. 13, No. 5, pp. 376-386.

[21] http://computing.breinestorm.net/software+paradigms+agent+development+engineering/

[22] http://iaks-www.ira.uka.de/iaks-calmet/papers/VKC_Maret_Calmet.pdf