Agents and Objects: Towards Component Integration for Manufacturing Systems Re-engineering

L. B. Sheremetov^{1,2}, A. V. Smirnov²

¹*Computer Science Research Center, National Technical University, Mexico (CIC IPN)*

²*St.-Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences (SPIIRAS)*

¹ *Unidad Profesional Adolfo López Mateos, México, D.F.*

07738, México tel.: (915) 7296000 ext. 56576,

fax: (915)5862936, e-mail: sher@pollux.cic.ipn.mx

² *39, 14th Line, St.-Petersburg, 199178, Russia,*

tel.: (812)218-8071, fax: (812)218-0685

e-mail: smir@mail.iias.spb.su

Abstract

This article reports on an experiment in which the pre-existing engineering systems for the small and medium size FMS re-engineering were transformed into an open framework of cooperating agents - DESO. To reflect the standards of communication in a distributed multi-agent system, the multi-level model and the agent template, reflecting this model are proposed. Agents coordinate their activities using a communication and control language (currently, restricted version of KQML) and DCOM/ActiveX interaction mechanisms. The results of experiments and future work on the system development are considered.

Keywords

Multi-Agent Systems, Distributed Object Computing, Communication, Manufacturing Systems, Re-engineering

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: [10.1007/978-0-387-35390-6_58](https://doi.org/10.1007/978-0-387-35390-6_58)

L. M. Camarinha-Matos et al. (eds.), *Intelligent Systems for Manufacturing*
© IFIP International Federation for Information Processing 1998

1 INTRODUCTION

Complex technical systems such as FMS consist of numerous components with complex interrelations between them. Computer environments used for the design and re-engineering of such systems are usually composed of different software components. So, the problem how to incorporate pre-existing and newly developed systems into the common environment has become an important issue. Distributed problem solving and concurrency become a fundamental requirement, providing cooperation between engineering systems (Darr and Birmingham, 1994; Frost and Cutkosky, 1996; Osório and Camarinha-Matos, 1996).

Multi-agent problem solving, where agents cooperate to achieve common objective, is a rapidly emerging and promising technology (Singh, 1994; Müller et al., 1997). Usually the problem partitioning gives tasks that cannot be solved in isolation or even if they can, the environment conditions are constantly changing, and it is impossible to synthesise a global solution because of the incompatibility or inconsistency of the partial ones. As a result, there arise the problems of communication between heterogeneous components, coordination, and conflict resolution, as well as the need for communication reduction during the problem solving process. That is why there are not many successful applications of DAI technology in real world manufacturing domain. Probably the most famous are PACT (a testbed for building large-scale concurrent engineering systems), DesignWorld, ARCHON (Architecture for Cooperative Heterogeneous ON-line systems), KAoS (Knowledgeable Agent-oriented System), to mention some of them (Cutkosky et al., 1993; Huyhn et al., 1993, Jennings, 1994, Bradshaw 1996).

This article reports on an experiment in which the pre-existing engineering systems for the small and medium size FMS re-engineering were transformed into an open framework of cooperating agents, called DESO (DESIGN of Structured Objects). The approaches, forming the basis of the proposed work are developed within the frameworks of multi-agent systems and distributed object computing, and their application in Concurrent Engineering (CE) and Distributed Enterprise Computing Environments (DECE).

The aim of this article is to discuss the problems of component integration within the DESO framework and also the ways of DESO incorporation into the general object-oriented and/or multi-agent CAD frameworks. We have verified the proposed architecture for a number of FMS shop-floor level re-engineering tasks that indicate the feasibility of the use of the proposed models and tools for real-size industrial applications.

2 DESO: A GENERAL FRAMEWORK FOR COMMUNICATION IN COOPERATIVE RE-ENGINEERING

In the late 80th in the laboratory of computer integrated systems of the SPIIRAS we began the development of engineering systems for FMS design and simulation:

- requirements specification analysis,
- configuration design,
- layout design,
- discrete-event simulation and performance analysis, and

- cost/benefit analysis.

First, these systems were developed and implemented as stand-alone components (but with the certain degree of concurrency within the subsystems), using different platforms and communication mechanisms, such as distributed objects, actors, and blackboards. In the 1994 we have started the development of the DESO framework for configuration design and component integration according to the CE concept and agents technology. At this stage, we have analysed 2 different technological aspects of the development of multi-component distributed CAD systems:

1. the integration of the existing components, developed without the concern of their future integration, and
2. the development of the components, having all the necessary means for the future integration in the general CAD systems based in the agent and/or object technologies.

The first stage was associated with the development of interfaces, Knowledge Query and Manipulation Language (KQML) based protocols, architecture, and knowledge sharing models between the components, based on federation architecture (Smirnov et al., 1995, 1996). At the current stage of our experiments, an open framework for component integration has been developed. This work is carried out in cooperation with the Agents laboratory of the CIC IPN.

In order to reduce the impact of complexity of software systems we need to define an architecture which outlines distributed application components and allows easy integration of any application system into a DECE. We think that currently there is no consensus on the issues if the use of "objects/components" or "agents" is a necessary and sufficient enabler of software component integration and reusability in the distributed environments. In multi-agent systems, KQML and ARCOL specifications are proposed as standards for agents communication (Finin, 1995; FIPA, 1997), for example:

- Agent-K and LALO. The both languages are based on the concept of agent-oriented programming (Shoham, 1993) and use KQML and sockets for agent communication (Davies and Edwards, 1994; Gauvin, 1995).
- The Java Agent Template (JAT). Agents are stand-alone or transient processes (applets), using KQML, sockets and Java RMI for communication (Frost and Cutkosky, 1996).

Unfortunately the use of KQML does not solve the problem of component interoperability in complex distributed environments, composed of a number of applications, not all of which support the agent framework, such as CAD systems. Distributed object computing paradigm also deals with this problem. It proposes an architecture, which treats each element of the system as a reusable software component. These objects are encapsulated using a single, well-defined, abstract interface, which insulates the user from the languages and tools used to implement it. This concept however also has well known limits being applied to the knowledge-based tasks or for the applications that do not support it.

In DESO we have proposed the following communication architecture that can be considered as another alternative to component integration based on KQML and

DCOM (DCOM, 1996), taking advantages of the both paradigms (Figure 1). The ideas of KQML-CORBA integration were also discussed in (Benech et al., 1997).

Figure 1. Multi-level communication architecture

The first level defines the high-level strategy and the negotiation structure between agents. For the dynamic and cooperative context of manufacturing re-engineering system we define it as a combination of request/answer and knowledge exchange strategies. In the current implementation in DESO, at the first level of communication architecture the message semantics is connected with the agent actions by means of commitment rules. The second level defines message semantics, request type and content and is based on KQML.

The third level defines the standard mechanism of logical component interactions (Chappell, 1996). At this level, the OLEnterprise mechanisms take KQML expressions and pass them as semantic events. This level is used to map data and operation by converting events and messages to the set of KQML performatives. It also maps events data structure to the data structure accepted by Object Linking and Embedding (OLE) servers. ActiveX/OLE provides interoperability functions using Object Data Base Connectivity (ODBC), Object Request Broker (ORB), Internet Inter-ORB Protocol (IIOP), and Object Remote Procedure Calls (ORPC) that have been extended to distributed implementation.

Finally, OLEnterprise mechanisms use Transmission Control Protocol/Internet Protocol (TCP/IP) protocol stack to implement low level communications. In the case other mechanisms than OLE are used, KQML messages can be transmitted directly by communication Application Programming Interfaces (API).

We have been developed the DESO agent template, taking this model into account (Figure 2). The template is based in 3-layer architecture. At the layer of mental states the class IntelligentAgent supports the control mechanisms for commitment rules, capabilities, and beliefs. At the layer of agent behaviour, the ReactiveAgent class supports the commitment rule execution and solution search based in constraints propagation facilities. At the communication layer, CommunicationAgent class supports KQML- and ActiveX based communication with other system components, according to the schemata, described above.

Figure 2. DESO agent template

3. CCDA MULTI-AGENT ARCHITECTURE

The proposed DESO framework have been used in the development of Concurrent Configuration Design Advisor (CCDA) - the distributed multi-agent expert system for configuration design of FMS (Smirnov and Sheremetov, 1998). For the purposes of this article we shall briefly discuss the key ideas of the approach.

Configuring is the construction of a technical system according to the requirements of a specification by selecting, parameterizing and positioning instances of suitable component types from a given catalogue. Manufacturing system under design is decomposed into fragments. A fragment is a structured model of a system to be configured.

In CCDA an agent is considered as a computational process, capable of solving a limited portion of a configuration design problem, and communicating with other agents. So, only communication and behaviour levels of agent template have been used. An agent is associated with each task of fragment configuring. We shall call this agent a 'design agent' or D-agent. Each D-agent is oriented to solve a fragment-configuring problem on its level of decomposition organising a solution search. Initially, a single agent, responsible for the entire project configuring, is created for the root of the tree (D-agent server). It tries to find a solution for the whole system in the database, and if it does not exist, decomposes the system into fragments and creates a new agent for each fragment (D-agents clients). This process continues iteratively for each level of decomposition. A channel created for

each new task is used to return to its parent all the solutions located in its sub-tree. Hence, new agents and channels are created in a wavefront as the search progresses down the search tree. If no template is found for a fragment in the knowledge base (KB), D-agent generates an agent of another type - a 'project assistant' or A-agent that assists a domain expert to generate a new template.

Each D-agent client when finding fragment solutions sends them to the D-agent server in the form of a proposal. Server evaluates these concurrent proposals. If these proposals satisfy interval constraints then it sends the confirmation message and writes these proposals in the design KB. When the deadlock is reached, i.e. an agent can't decompose its fragment or find appropriate solution, it transmits a message to the upper level, asking for relaxation of constraints. These steps are repeated until a satisfactory decision is achieved or it is recognised that conflict can not be solved due to the over-constraintness.

CCDA communication architecture within the DESO framework is depicted in Figure 3. Agents can use peer-to peer communication structure; in this case the IP address is directly defined in the *receiver* field of the KQML message. On the other hand, indefinite agent or method invocation mechanism is also provided. This type of communications is supported by a special Service Control Manager (SCM) in DCOM and Agent Name Server (ANS).

Figure 3. DESO high-level communication architecture

To support the message content between different subsystems, we have used a special class of information agents (facilitators or mediators). Facilitators are processes (agents) which situate themselves between provider-processes and consumer-processes and perform services such as providing standardized interfaces; integrating information from several sources; translating queries or replies. Facilitators are commonly proposed as an effective method for integrating new information systems with inflexible legacy systems (Wiederhold 1992, Cutkosky et al., 1993). For example, to access shared knowledge and data, an agent generates SQL queries for an information agent, responsible for component database maintenance, in order to be supplied with the relevant information about components. Then these data are converted into agent's internal representation.

4 EXPERIMENTAL RESULTS AND FUTURE WORK

This section presents some experimental results on the DESO evaluation and discussion of the future work. We have provided a number of experiments on the FMS re-engineering on the shop-floor level. The goals of these experiments were:

- to evaluate distributed multi-agent algorithm for configuration design with CCDA, and
- to establish a reliable communication within a team of cooperating agents and users from different DESO subsystems to evaluate the proposed communication model.

To accomplish with the first objective, a sequence of experiments were performed, where D-agents were incrementally added and assigned to the fragment configuration tasks as the FMS was decomposed into fragments. We analysed performance in terms of time required to solve the problems in uni-agent and multi-agent fashions. To make the comparisons fair we have distributed physically the D-agents for the level of FMS cells between the Pentium PCs with Windows95, though at the level of equipment configuring they were executing in parallel. Figure 4 shows visualised results of a typical configuring task for 6 FMS cells. For the illustrative purposes, only 3 D-agents are presented: D-agent project server AgentFMS1, D-agent client 11b164 (FMS shop-floor fragment, served by a separate AGV), D-agent client la_155f3 - level of the FMS cell.

An experimental FMS consisted of a number of cells (usually 6-8), including laths, milling and drilling machines, different types of stores and transport system. Each cell typically consisted of 1-2 machines, a robot, and a clamping/resetting station. Each type of the equipment had from 20 to 200 instances to be analysed.

An A-agent supports the general system interface. This provides the designer with one entry point to the system facilities, such as master function tools for template definition, and makes him an active member of the agent community. Each D-agent has also editing facilities, which makes it possible to a designer to correct the results, proposed by agents. The results of any intervention are passed to all the interesting agents and are subject to the overall constraint satisfaction.

We were giving our agents tasks that were too small. In this case the system was supposed to have an optimum in the agent population, because of the negative impact of communication on the overall performance. In our experiments it was

Figure 4. Example of FMS configuration results with CCDA

the case. When comparing with the uni-agent problem solving time, the algorithm shows the decrease of total computational time with respect to the increase of number of agents. We can explain it that by using fragment templates as the model for agents generation, CCDA defines all necessary communication paths a priori and minimise the amount of information communicated.

At the second stage of the experiments the following subsystems that are integrated in the DESO framework have been considered:

- 1 the PQE system for requirement specification, implemented in Visual FoxPro with ActiveX interface, Windows95 platform,
- 2 the CCDA, which are currently enhancing two main versions: one written in C++ to take advantage of Microsoft's ActiveX and DCOM technologies, using OLEEnterprise and DCOM 95 as the experimental testbeds and a KQML library, and the second written in LALO to take advantages of agent-oriented programming paradigm (both use Windows95 as the platform),
- 3 the layout design system, implemented in CLP(R) for Unix, and
- 4 the discrete-event simulation system, implemented in C++ for Unix.

At the moment only for the first 2 Windows based systems the communication protocols were tested. The use of OLE Gateway enables Windows development tools to integrate applications from Unix environments. Exploring this advantage of the architecture we are currently working on the incorporation of the Unix based components into the proposed framework.

5 CONCLUSIONS

Our experience with the current DESO architecture has shown it to be a powerful and flexible basis for system integration for the FMS re-engineering tasks. The strength of the architecture derives from several sources:

- it is built on a foundation of distributed object and agent technologies;
- it supports the work with component integration architectures such as DCOM/ActiveX, CORBA, and Java;
- it supports inter-agent communication on the knowledge level, using KQML specification that facilitates cooperation with the other agent-based systems;
- it provides facilities for agents names and services registry as the system functional specification;
- it supports simple agents to be implemented with embedded constraint satisfaction mechanism as the basic agent functionality, while providing facilities to develop more complex ones;

ACKNOWLEDGMENTS

The Ministry of Science and Technology of Russia initially supported the work described under the grant N236/132. Support for this work has been also provided by the CONACYT, Mexico, under the grant N 3923P-A9607 and REDI Program.

REFERENCES

- Benech D., Desprats T., Raynaud Y. (1997) A KQML-CORBA Based Architecture for the Communication of Intelligent Agents in Cooperative Service and Network Management. *Proceedings of the 1st IFIP/IEEE International Conference on Management of Multimedia Networks and Services*, Montréal, Canada.
- Bradshaw J.M. (1996) KAoS: An open agent architecture supporting reuse, interoperability, and extensibility. *Proceedings of the 10th Banff Knowledge Acquisition for Knowledge-Based Systems Workshop*, Banff, Alberta, Canada, 48:1 - 48:20.
- Chappell D. (1996) Understanding ActiveX and OLE. Microsoft Press.
- Cutkosky M.R. et al. (1993) PACT: An Experiment in Integrating Concurrent Engineering Systems, *IEEE Computer*, **26**, 1, 28 - 37.
- Darr T.P. and Birmingham W.P. (1994) Automated Design for Concurrent Engineering, *IEEE Expert*.
- Davies W.H.E. and Edwards P. (1994) Agent-K: An Integration of AOP and KQML. AUCS/TR9406, Univ. of Aberdeen, Scotland, UK, AB92UE.
- Distributed Component Object Model protocol DCOM/1.0. (1996) Network Working Group, Internet-Draft.
- Finin T., Labrou Ya., and Mayfield J. (1995) KQML as an agent communication language, in *Software Agents* (ed. Jeff Bradshaw), MIT Press.
- FIPA7412, (1997) Draft 1.0, document of the Foundation for Intelligent Physical Agents (FIPA), April 18th.
- Frost H.R. and Cutkosky M.R. (1996) Design for Manufacturability via Agent Interaction. In *1996 ASME Design for Manufacturing Conference*, Irvine, CA, Aug., 18 - 22.

- Gauvin D. (1995) Un environnement de programmation orienté agent. Dans *Troisièmes journées francophones sur l'intelligence artificielle distribuée et les systèmes multiagents*, St-Baldoph, Savoie, France, 15-17 mars.
- Huyhn P.N., Genesereth M.R., and Letsinger R. (1993) Automated Concurrent Engineering in Designworld, *IEEE Computer*, **26**, 1, 74 - 76.
- Jennings N.R., (1994) The ARCHONTM System and its Applications, *II Int. Working Conference on Cooperating KBS (CKBS-94)*, Keele, UK, pp. 13-29.
- Intelligent Agents III (Eds. Müller J., Wooldridge M.J. and Jennings, N.R) (1997) *Proceedings of the 3rd International Workshop on Agent Theories, Architectures, and Languages. Lecture Notes in Artificial Intelligence*, **1193**, Springer-Verlag.
- Osório A.L., and Camarinha-Matos L.M. (1996) A Federated Multi-Agent Infrastructure for Concurrent Engineering (CIM-FACE), *Studies in Informatics and Control*, **5**, 2, 143 - 156.
- Singh M.P. (1994) Multiagent Systems, A Theoretical Framework for Intentions, Know-How, and Communications. *Lecture Notes in Artificial Intelligence*, **799**, Springer-Verlag.
- Shoham Y. (1993) Agent-oriented programming. *Artificial Intelligence*, **60**, 1, 51 - 93.
- Smirnov A.V., Sheremetov L.B., Romanov G.V., Turbin P.A. (1995) Multi-Paradigm Approach to Cooperative Decision Making, *Proceedings of the II International Conference on Concurrent Engineering: Research and Applications*, McLean, Virginia, Concurrent Technology Corporation, 215 - 222.
- Smirnov, A.V., Sheremetov, L.B., Turbin, P.A. (1996) Information Support of FMS Configuration Design, in *Balanced Automation Systems II - Implementation challenges for anthropocentric manufacturing* (eds. Camarinha-Matos L.M. and Afsarmanesh H.), Chapman & Hall, 357 - 364.
- Smirnov A.V. and Sheremetov L.B. (1998) Complex Systems Configuring Based on Multi-agent Technology. *Automatic Control and Computer Sciences*, Allerton Press, N.Y. (to appear).
- Wiederhold G. (1992) Mediators in the Architecture of Future Information Systems , *IEEE Computer*, **5**, 3, 38 - 49.

7 BIOGRAPHY

Leonid Borisovitch Sheremetov received his Ph.D from SPIIRAS. His research interests include Multi-Agent Systems, Decision Support Systems, Expert Systems, and Intelligent Tutoring. Dr. Leonid Sheremetov is a professor of the Agents Laboratory in CIC-IPN, México and a senior research fellow of Computer Integrated Systems Laboratory of SPIIRAS, Russia.

Alexander Victorovitch Smirnov received his Ph.D from St. Petersburg University of Electric Engineering and D.Sc. from SPIIRAS. His research interests include knowledge engineering, multi-agent systems, decision support systems, design theory, concurrent engineering, and virtual enterprises. Prof. Smirnov is a Deputy-Director and a head of Computer Integrated Systems Laboratory of SPIIRAS, Russia.