Agent-Oriented Material Flow Control System Based on DCOM

Dr. Ronald Schoop
Ralf Neubert
Schneider Automation GmbH
Development Central
Steinheimer-Stasse 117, 63500 Seligenstadt, Germany
Email: ronald.schoop@modicon.com, ralf.neubert@modicon.com

Abstract

This paper reports on the design and realization of an agent-oriented control system dedicated for material flow systems. The implementation is based on DCOM (Distributed Component Object Model). Regarding the increasing availability, performance and acceptance of distributed object-oriented systems as well as the growing number of agent-oriented software it is natural to adapt and use the capabilities of both object-oriented and agent-oriented software as a platform for distributed industrial applications.

1. Introduction

Today's transferlines are producing workpieces (e.g. cylinderheads, engines) at a high production rate, but with enormous limititations:

- the availability of the whole system is poor because of the fixed in-line arrangements of the machines,
- the design of a line is dedicated for a specific type of workpiece only and
- extensions and modifications are very expensive.

The basic idea is to use general machines with redundant manufacturing profile and to supply these machines with workpieces via a flexible material flow system. Any workpiece has now multiple choices for manufacturing. If a machine is failed, another can take the job over.

A conventional material flow pre-planning wouldn't be able to manage all the possibilities at production time.

What is needed is a control system, which

- is strictly object-oriented to handle the complexity of the system (thousands of workpieces and dozens of machines)
- uses the capability of autonomous and robust behavior in order to handle failures of a sub-system without further impact to the overall system.

The aim of this paper is to describe the basic architecture of the designed control system, to focus on some relevant implementation aspects and to draw finally conclusions based on experiences on real shop floor manufacturing.

2. Requirements

The agent-oriented production system requirements are:

- robust behavior (of communication and algorithm) of software components based on Windows NT 4.0 platform with DCOM,
- flexible transportation system (easy to reconfigure and to extent),
- higher availability of production system compared to conventional transferlines,
- handle failures of a sub-system and guarantee a production rate and
- manufacture different types of workpieces simultaneously.

3. Architecture

In co-operation with DaimlerChrysler AG an agent-oriented production system was designed. It's based on agent system specification by DaimlerChrysler AG Research and Technology, Berlin – Germany [1].

3.1. Software design

During design phase the auction-based, agent-oriented system passed through incremental steps of simplification.

Starting point was a mobile agent platform based on JAVA . To simplify the architecture of common mobile agent systems and to consider the fixed arrangement of the production system as well, the system was re-designed with static, non-mobile agent components.

A limited subset of interactions was defined during the next step of design. Interactions are often described with languages like KQML (Knowledge Query and Manipulation Language) [2] and usually processed by an event handler [3].

In fact the use of a mobile agent platform (JAVA / CORBA – Common Object Request Broker Architecture) for unknown areas (e.g.: internet search) was not absolutely essential caused by the good knowledge of the production system design and the strong relation of the components to the fixed shop floor environment and topology. Without the need for a JAVA-based mobile agent platform and a JVM (JAVA Virtual Machine) an object model based on DCOM [4] fits well. Fig. 1 shows the derived agent architecture.

Finally, the subset of interactions were described as set of methods provided by various agent objects.

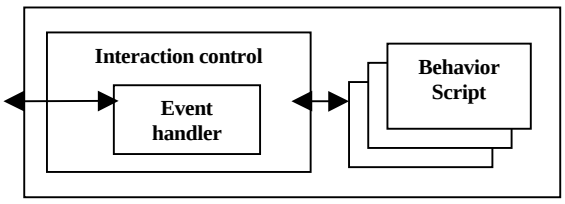

Figure 1 Simplified Agent Architecture

3.2. Component oriented design

The agent-oriented system consists of several PC- and PLC- (Programmable Logic Controller) based agents (see figure 2).

Agents are assigned to system components like machines or shift tables. This document is aimed to the PC-based components mainly.

Every agent controls its corresponding mechanical component (e.g.: shift table) autonomously. For any agent type a specific COM module (DLL) encapsulates the algorithm (behavior). This behavior module is linked at runtime to a DCOM-based communication framework (see figure 3), which provides a remote Agent Method Interface (AMI) for interactions.

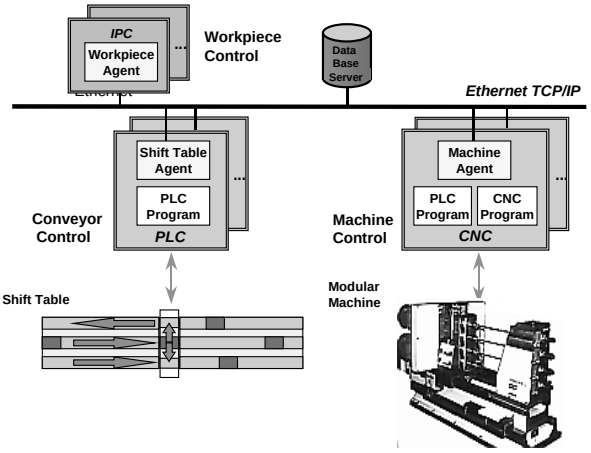

Figure 2 Agent System View

The idea of separation the algorithm and communication allows the use of an unique framework for different agent types (DLLs), to build a preprocessed algorithm- and communication-related fault-detection (agent-type independent) and to keep small and reliable DLLs for different algorithm as well as a low software maintenance effort.

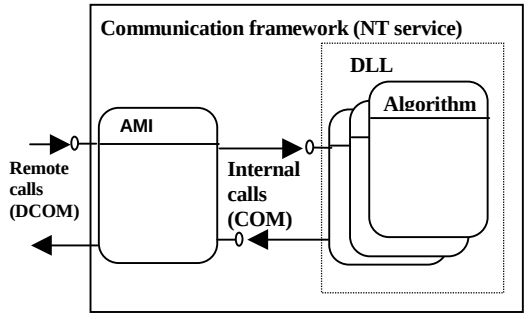

Figure 3 software architecture at component level

The AMI provides invocation and confirmation methods. An interaction (remotely or locally) between agent objects is formed by calling an invocation method and its corresponding confirmation method for the asynchronous result of the interaction (e.g.: invocation method “Request” with confirmation method “RequestConfirm”).

In case the invocation method would be handled synchronously at runtime the calling object has to wait until the method returns the result. If a remote call fails, the result will be received after a timeout, which is handled by DCOM.

In order to create a more robust asynchronous behavior regarding the desired level of fault-tolerance, an interaction is separated into invocation and confirmation method.

A remote call will be handled by the AMI. If a self-defined timeout occurs, the algorithm will be informed by the framework. There's a distinct improvement if a caller object has the possibility to go on in case of an error and to switch interactions before a DCOM-handled timeout occurs.

The algorithm object is build by some threads representing each type of agent. Therefore an agent can proceed several interactions for various workpieces simultaneously at runtime.

The flexible material flow system requires several interactions like 'requesting offers', 'ordering jobs', 'creating transportation requests' etc. described as state charts. AMI-methods are linked to these interactions to comply with this requirements. Some steps could be critical or non-critical. Critical steps are necessary for algorithm. Therefore transitions can be defined. In order to enable such a transition in the state chart, the corresponding confirmation has to be received. Otherwise an alternative interaction or state chart will be selected. For non-critical steps or interactions transitions are enabled always.

4. Results and related works

The system has shown a robust and redundant behavior during common shop floor manufacturing as well as performance tests regarding an increased availability.

Different types of workpieces where processed simultaneously (chaotic manufacturing).

The benefit is obvious regarding the handling of sub-system failures.

The availability of conventional transferlines collapses in case of an error caused by the fixed sequential arrangement of the machines. Machines after the defective equipment running empty. On the opposite site of the failed machine the manufacturing system goes overfilled and stops.

Figure 4 illustrates why the agent-oriented manufacturing system overcomes failures of sub-systems: Machine #2 fails (e.g.: broken tool) in mid day. Running interactions of the responsible machine agent are switched by a workpiece agent to other machine agents (machine #3 and later machine #1). These are taking over and sharing the current jobs subsequently if they are providing the right handling scope like drilling or milling. A lower production rate remains guaranteed compared to a conventional transferline. For that reason the availability of the overall system is higher.

The Specifications and simulations done by DaimlerChrysler AG are proved now under real shop floor conditions. A complete manufacturing system controlled by auction-based agent-oriented software components [5] is running without any problems since September 1999 – as far as we know the worldwide first agent-controlled transfer line. The results of the prototype installation are corresponding to the simulations and expectations.

5. Conclusions

A static, non-mobile agent-oriented software based on Windows NT and DCOM can be a possible solution for industrial manufacturing applications. In order to build autonomous sub-components the object-oriented composition of a production system has to be dedicated to its mechanical components as well as the desired behavior of the specific component. Agent-based technology fits best to this.

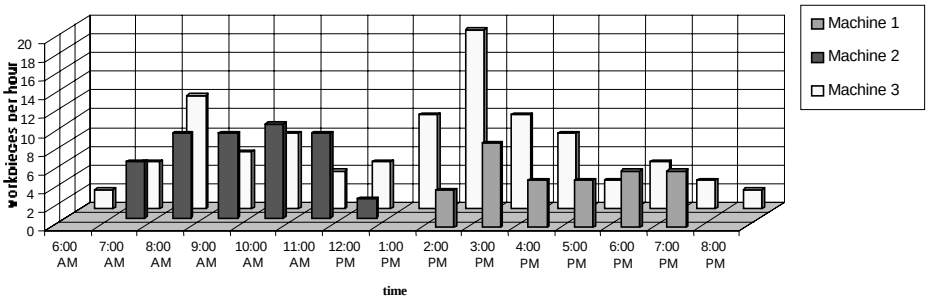

Figure 4 redundant behavior

The usage of an unique communication framework (NT service) and encapsulation of algorithm into type-specific modules (DLL) allows to extend and to migrate the agent software more easier.

The framework extends the DCOM platform and provides features like a user-defined level of fault-detection at application layer.

Regarding the limited set of interactions within the fixed arrangement of the manufacturing system and the restricted degree of freedom of its components as well, a modification of the behavior can be realized by an pre-compiled and exchangeable set of instructions.

Based on the results and experiences different fields of applications are conceivable (e.g.: assembly lines) for agent-oriented software systems in future.

For industrial purposes a description of the components behavior as state charts or function blocks could be done by using standards like IEC (International Electro-technical Commission) 61499-1 [6].

Further, agent-based software gives the opportunity to realize holonic systems [7] in general.

6. References

[1] Sven Brückner, Stefan Bussmann, Klaus Schild, and Harald Windisch, Grobspezifikation des Agentensystems im Prototypen. Technical report, DaimlerChrysler AG, Berlin/Germany, 1998.

[2] J. Weber, T.Finin, et al. Specififcation of the kqml agent communication language (draft). Technical report, The DARPA Knowledge Sharing Initiative, 1993.

[3] A.T.M. Aerts, M. Dalmeijer, and D.K. Hammer, Mobile agent architectures: What are the design issues? In *Proceedings of the Engineering of Computer Based Systems (ECBS)*, 1998.

[4] Eddon, Guy, and H. Eddon. *Inside Distributed COM*. Redmond, WA: Microsoft Press, 1998.

[5] S. Bussmann, and K. Schild, "Self-Organizing Manufacturing Control: An Industrial Application of Agent Technology", to appear in *Proceedings of The Fourth International Conference on Multi-Agent Systems (ICMAS '2000)*, Boston, 2000.

[6] IEC CDV 61499-1, Function Blocks for Industrial-Process Measurement and Control Systems Part 1 - Architecture, 1999.

[7] S. Bussmann, and D. McFarlane, Rationales for holonic manufacturing control. In *Proceedings of the 2nd Int. Workshop on Intelligent Manufacturing Systems*, 1999.