

A Multi-Agent Computer Aided Assembly Process Planning System

Zhang Shijie, Jing Shuang

Shenyang Institute of Automation, Chinese Academy of Sciences, P. R. China

Abstract

This paper presents a multi-agent computer aided assembly process planning system (MCAAPP) including system framework, global facilitator, the macro agent structure, agent communication language, agent oriented programming language, knowledge representation and reasoning strategy. The system can produce the technological file and technological quota, which satisfies the production needs of a factory.

Keywords: multi-agent, intelligent-agent, computer aided assembly process planning, knowledge representation, KQML

1 Introduction

The usage of multi-agent and intelligent-agent technologies to develop large software system began since 1990. With the development of concurrency and distribution, multi-agent system becomes a studying trend. Some researchers developed agent-based application systems applied in traffic control, office information processing, distributed sensor, planning and scheduling, etc. in manufacturing system [1]. The results obtained were satisfying. Recently, multi-agent technology was used in computer aided process planning (CAPP) and gained much attention. Several papers were published in this domain. The development of CAPP systems follows the changes in the manufacturing industrial environment and relates closely to the development of information, computer and artificial intelligence technologies. In particular, today, the need for flexible manufacturing systems increases due to economic and ecological constraints. Therefore, the new CAPP system must have an open and flexible structure to meet the needs of current and future manufacturing industries.

The activity of ship hull assembly is very important in shipbuilding. In general, assembly work used for ship hull assembly accounts for 45 % of the whole shipbuilding activity [2]. Therefore, the improvement of assembly productivity can yield great economic

efficiency and thus the requirements of assembly process optimization and automation are urgent. Because ship hull assembly technology is very complicated, it is necessary to develop an efficient computer aided assembly process planning (CAAPP) system for shipbuilding enterprises by multi-agent technology to meet the needs of market competition.

2 The Framework of MCAAPP

The framework of an agent-based system describes mainly how to construct computer system according to the agent theory. The choice of framework affects the degree of asynchronous, concurrent, or persistent activity that occur; the way information is stored and shared and how agents communicate. In general, there are three kinds of multi-agent frameworks: (1) Agent network. In this type of agent framework, the communication among agents is direct. Each agent must know where and when messages should be sent, what other agents are available and what capabilities other agents have. (2) Federations. In this type of agent framework, connectivity and message routing are handled by facilitators. (3) Agent-based blackboards. In this type of framework, interactions among agents are managed by grouping. The local information on the blackboard can be shared by agents [3].

In this system, the federations agent framework was adopted. There is a global facilitator in the server, while each agent has its facilitator in the respective computer node. The framework of MCAAPP is shown in Figure 1.

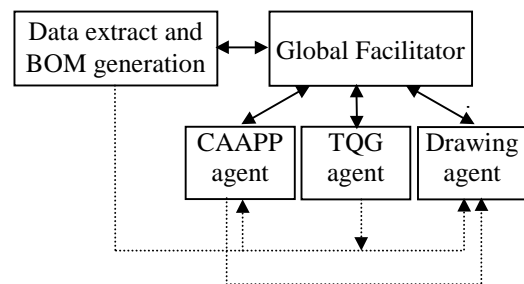


Figure 1. The framework of MCAAPP. The single arrowhead denotes data stream, while double arrowhead denotes message stream. The key

of the CAPP system decomposition lies in the grain size of the agent. The MCAAPP has four macro agents: data extract and Bill of Materials (BOM) generation agent, computer aided assembly process planning agent, technological quota generation (TQG) agent and technological graph drawing agent. We also divide the computer assembly process planning macro agent into several micro agents, such as hull type distinguish agent, assembly process planning agent, technological file print agent, etc.

2.1 Global facilitator

Global facilitator is an agent, which coordinates the interactions among agents and offers communication services. Global facilitator acts as repositories for knowledge about what information and services are available. When an agent needs a service, it communicates this to its facilitator, which matches the request to advertise capabilities of other agents (either locally or system-wide). Global facilitator provides useful network services. It maintains a registry of service names; transmits messages according to the content of KQML (Knowledge Query and Manipulation Language), offers matching and disposing for the message sender. The structure of Global Facilitator is shown in Figure 2.

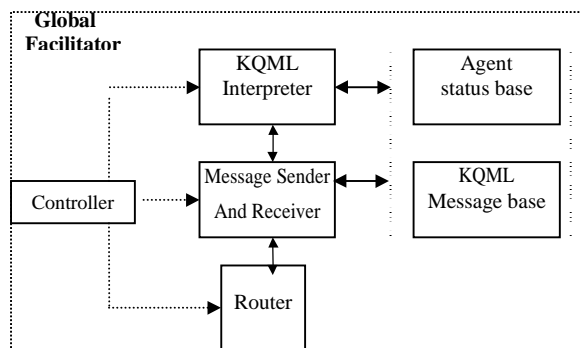


Figure 2. Structure of the Global Facilitator.

The router takes charge of the communication between this agent and other agents. Message sender and receiver of the facilitator sends and receives KQML messages. The interpreter of global facilitator takes charge of analyzing, managing and dealing with the KQML messages. Controller manages the execution of router, messages sender and receiver and KQML Interpreter.

2.2 The structure of macro agent

There are four macro agents. Each macro agent can be divided into several micro agents [4]. The structure of a macro agent is shown in Figure 3.

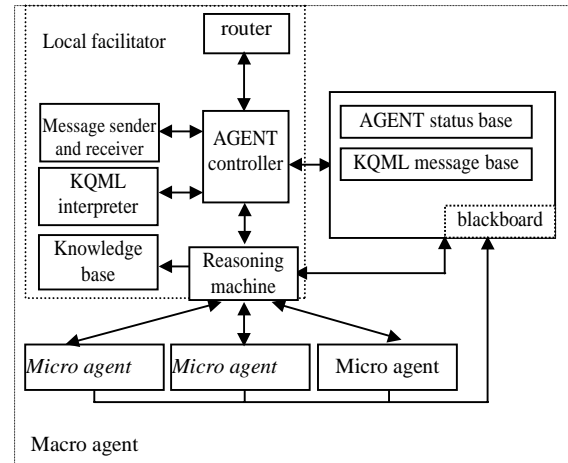
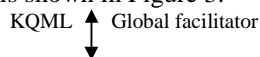


Figure 3. The structure of a macro agent.

The local facilitator takes charge of the correspondence among the micro agents.

- (1) Router: orient and seek global facilitator.
- (2) Agent controller: initializes and gives overall control of the running agents, sends register/unregister message to global facilitator, supervises local facilitator and triggers global facilitator.
- (3) KQML message manager: interprets and disposes the received messages.
- (4) Reasoning machine: processes logic reasoning by means of the content of knowledge base and blackboard.
- (5) Micro agent: triggered by the reasoning machine, does its domain work and writes the information on the blackboard.

3 Agent Communication Language

In this system, KQML is used as an agent communication language. KQML is both a message format and a message-handling protocol supporting run-time knowledge sharing among agents. KQML includes lots of extensible performative, which defines manipulation to the knowledge and goal [5]. Through the facilitator, KQML provides the basic mechanism of knowledge sharing to correspond to other agents' cooperation. In this system, the subset of KQML was selected as the communication performative, such as Ask-if, Ask-one, tell, untell, insert, advertise, register, unregister, etc. The content message can be expressed in any representation language and written in either ASCII strings or in one of the many binary notations. The attributes are expressed by keywords, which include sender, receiver, from, to, content, language, ontology and in-reply-to/with. In order to match the system requirement, we define TASKSTATUS, TASKNAME, RUNNING and DBPATH as keywords

of content. For example, a KQML message representing a query about the TASKSTATUS of task1 might be encoded as:

```
(Ask-one
 : content (TASKSTATUS task1 ?taskstatus)
 : sender agent1
 : receiver facilitator
 : language PowerBuilder
 : ontology MCAAPP)
```

4 Communication Mechanism Between Facilitator and Other Agents

PowerBuilder is a powerful application system development environment. By using PowerBuilder, a user can easily access the data from server databases. To allow the user to work with persistent, shared data in a distributed application, PowerBuilder provides support for shared objects. Shared objects are user objects that can be shared by multiple client connections. The concrete scenario is as follows:

- (1) Mutual client /server structure is adopted.
- (2) The facilitator application has the user object, which includes write-lock symbol and disposal function. Other agent application has the proxy of the user object. When a macro agent is going to send a statement of KQML to the facilitator, it sets up the connection first and queries write-lock symbol, then inserts a record to the KQML message base and triggers disposal function.
- (3) When the facilitator is going to send a statement of KQML to macro agent, the steps involve resemble to that mentioned above.

5 Agent Oriented Programming Language

In order to realize an agent-based application system, it is necessary to design an agent oriented programming language. It is very convenient to encapsulate application system using agent oriented programming language. This kind of language and facilitator was used to construct a multi-agent system.

5.1 Knowledge representation

The syntax of the language is as follows:

Rule-set :: = rule1 | rule2 | ... |ru1en

Rulei :: = [item node_name func_name subitem-1 ... subitem-n]

item :: = identifier

node_name :: = identifier

func_name :: = F_AND|F_OR|ASK_STATUS|

DO_GET|F_EXCUTE|F_REGISTER

subitem-I :: = identifier

where, item is the name of task or subtask.

Node_name is the node name of a running task.

Func_name is the function name. The left of func_name is the subitems of item.

5.2 Function description

The function description is as follows:

- (1) F_AND: when the values of arguments are all true, the value of item is true, or false.
- (2) F_OR: when the values of any arguments is true, the value of item is true, or false.
- (3) ASK_STATUS(D): gets the information according to argument of the function.
- (4) DO_GET(D): gets the contents of argument D by means of list name and fields.
- (5) F_EXCUTE (item): runs a task named item.
- (6) F_REGISTER (item): write the object name of item into local knowledge base and transmit the status of the task into global facilitator.

Up to now, the system mainly provided for 6 functions. A symbol "F" was used before some functions to avoid the confusion between the functions in MCAAPP and that used in the PowerBuilder language.

5.3 Reasoning strategy of the interpreter

The reasoning strategy of the interpreter is as follows:

- (1) Task decomposition. A task or subtask of process planning can be decomposed into several subtasks according to some logical relations. Each subtask can also be decomposed into some subtasks. Until the leaf items are found, the execution module can be executed. It records the results on a global database (blackboard). The direction of the decomposition is from top to bottom.
- (2) Task synthesis. Task synthesis is the opposite of task decomposition. It synthesizes the results from knowledge sources and judges whether the task is solved or not. The direction is from bottom to top.
- (3) Task evaluation. Task evaluation is made by means of evaluation methods and the knowledge about task and computer node to establish the node-name. Then system distributes the task on other computer node to run concurrently.

The whole decomposition, synthesis and evaluation process are executed in the blackboard.

5.4 The encapsulation of agents

The encapsulation of agents is as follows:

(1) Data extract and BOM generation agent

After data extract and BOM generation agent receives a planning task, this agent extracts the design information from TRIBON software, forms the BOM and CAAPP needed and informs it to global facilitator.

(2) Computer aided assembly process planning agent

Computer aided assembly process planning agent adopts case-base reasoning method to produce technological file. This agent can run alone or run in CIMS environment. In order to adapt the multi-agent environment, this agent is encapsulated according to following style:

```
[Assembly_task F_AND a b c T_register]
[a F_OR f1 f2]
[f1 ASK_STATUS bom ]
[f2 F_keyboard_input bom]
[b F_AND get_b do_cap caapp_re]
[get_b DO_GET bom]
[do_cap F_EXCUTE caapp ]
[caapp_re F_REGISTER caapp ]
[c F_AND draw draw_re]
[draw F_EXCUTE auto_cad ]
[draw_re F_REGISTER auto_cad ]
[T_register F_REGISTER Assembly_task ]
```

The meaning of the rules mentioned above are as follows. The accomplishment of Assembly_task depends on right side conditions of F_AND function. That is, the execution status of subtask a, b ,c and T_register are all true. Then, the execution status of Assembly_task is true, and the global facilitator is informed. In order to judge the status of subtask bom, the local virtual knowledge base need to be queried. If subtask bom is not finished, this function (ASK_STATUS) sends a message to the global facilitator to get the status of subtask bom. The format of message adopts KQML criterion. When global facilitator gets the message, the interpreter analyse and process the KQML statements. If subtask bom is finished, the global facilitator sends back the accomplished status of subtask bom. Otherwise, the global facilitator informs the data extract and BOM generation agent to do this work as soon as possible. The accomplishment of subtask b depends on subtask get_d, do_a, caapp_re. That is, the status of these three subtasks are all true. By means of DO_GET, F_EXCUTE and F_REGISTER functions, the three subtasks can be done one by one. DO_GET can get the design data and BOM from Data extract and BOM generation agent. F_EXCUTE executes the caapp system. F_REGISTER informs the accomplished

status of the caapp to the global facilitator. Symbol c stands for technological graph drawing. It runs the Autocad software and then tells the status of technology drawing to the global facilitator. T_register tells the accomplished status of Assembly_task to the global facilitator.

The CAAPP system was written in PowerBuilder and SQL language. When the CAAPP system receives a process design task from the MIS system, the process feature extract module of the CAAPP system is triggered. It analyses the block and BOM information of the ship hull and then the case-based reasoning module of the CAAPP starts to work. Through case retrieval and case adaptation, the system produces a process file which satisfies the production needs of the factory.

(3) Technological quota generation agent

Technological quota generation agent outputs technological quota and material quota. This agent is encapsulated according to following style:

```
[gyjh_task F_AND a b cp2 ]
[a F_OR f1 f2]
[f1 ASK_STATUS bom ]
[f2 F_keyboard_input bom]
[b F_OR f3 f4]
[f3 ASK_STATUS h_info ]
[f4 F_keyboard_input h_info]
[cp2 F_AND get_bh do_cp2, gyjh_register]
[get_bh F_AND f4 f5]
[f4 DO_GET bom]
[f5 DO_GET h_info]
[do_cp2 F_EXCUTE gyjh ]
[gyjh_register F_REGISTER gyjh ]
```

where, gyjh is a micro agent which generates technological quota and material quota. h_info stands for welding information. In ship hull assembly process, technological quota consists of welding hours and rivet connection hours. Welding hours are divided into automation welding hours and hand welding hours. Because the system can directly get weld size from the CAD system, it can calculate the welding hours accurately and form an operation item list. In addition, the gyjh micro agent can output material consumption quota list.

(4) Technological graph drawing agent

Technological graph drawing agent draws the technological graph by means of the number of block and geometry information of the ship's hull. This agent is encapsulated according to following style:

```
[draw_task F_AND a , cp3 , T_register]
[a ASK_STATUS caapp]
[cp3 F_EXCUTE auto_cad ]
```

[T_register F_REGISTER draw_task]

After triggering the Autocad software, the technician can draw the technological graph according to the number of block and geometry information of the ship's hull in the drawing environment. Finally, the system can print the technological file and technological graph together.

6 Conclusion

The system developed has been running on personal computers located in BOHAI shipyard. Features of the MCAAPP system are as follows:

- (1) The design information (bill of material and welding information) can be obtained directly from the TRIBON system by MCAAPP.
- (2) The multi-agent methods were adopted to produce assembly process planning files and technological quota, which meet the requirements of the production.
- (3) The technological process information was directly sent into the Informix database of BAAN IV (an ERP software). The integration between TRIBON and BAAN software is realized.
- (4) Flexibility of the system is highly improved.

Acknowledgement

The authors wish to thank Engineer Yu Shiyuan and Meng Qinghua of Bohai shipyard for their rich process knowledge contributions.

References

1. **Hu Shungeng, Zhang Li, Theories, Technologies and Applications of Multi-agent System, Computer Sciences (China), vol. 26, No. 9, pp. 20-23, 1999.**
2. **Zhang Shijie, A computer aided assembly process planning system, 5th International Conference on Manufacturing Technology, 1999.**
3. **Susan E. Lander, Issues in multi-agent design systems, IEEE expert, pp. 18-26, 1997.**
4. Zhao Fuliang and Paul S. Y. Wu, A cooperative framework for process planning, Int. Jln. Computer Integrated Manufacturing, vol. 12, n2, pp. 168-178, 1999.
5. Finin, T, et al., The KQML information and knowledge exchange protocol, Third Intl. Conf. on Information and Knowledge Management, 1994.