A Communication Protocol for Holding Global Consistency in a Multi-Agent Based Multiple Representation System

Abdelouahed ZAKARI(*), Janah SAADI(**) and Marouane CHRISS(***)

Abstract -- The paper presents a simple communication protocol intended to keep global consistency in a distributed, heterogeneous multiple representation system (as for Computer Integrated Manufacturing CIM), where various specialized engineering teams collaborate for the development of the same product.

The system can be viewed as a network of CAD models, considered as a unified and unique representation of a given solid object. To keep consistency between CAD Models, which are different representations for the same physical object, the communication protocol is defined between Agent-Modules.

Key words – Online Collaborative Work, Communication protocol, Global consistency, multi agent systems, multiple representation, CAD-CAM, Finite State Machines.

I. INTRODUCTION

In computer Aided Design and Manufacturing industry and because of economical concurrency, changing requirements, integration and interoperability are main issues that allow for reduced development costs, efforts and time. These goals are accomplished through design information sharing among multi-disciplinary engineering teams (CAD engineers, analysis engineers, and so on) located in different sites while keeping consistency of the representation (local model) hold by each team of the global design and manufacturing process.

This paper is based on the Multiple Representation works [1][5], where multiple products modeling approach is presented. This approach is based on methodological elements associated to the proposed multiple models paradigm and on previous work in defining components, for instance Intelligent modules [3], Intelligent Agents and agent based communication [21], [22], [23] and [24].

Though the protocol is showed here in the CIM domain, its principle is general and can be applied to other domains such as image processing.

The Module concept was originally defined in object based design and programming (interactive component that use communication port and connections), while Agent concept was defined in Multi-Agent Systems (MAS). The component concepts of Intelligent Module and Intelligent Agent are similar in our approach.

In a multiagent system, communication enables the agents to exchange information so they can coordinate their actions and cooperate with each other. This raises the important question of what communication protocols and mechanisms are to be used to enhance collaboration between communicating agents.

II. PRESENTATION OF THE APPLICATION DOMAIN

In manufacturing activities, engineers use geometric models (instances in CIM domain terminology), which represent the part shape, originally with interconnected edge segments, and later by using 3D solid models which model the volume enclosed by the shape of the physical design. BRep (Boundary Representation) and CSG (Constructive Solid geometry) are two solid modeling techniques that have received most attention in CAD/CAM community [6]. Product modeling is an evolution of solid modeling. Geometric solid models were enriched to take into account semantic information (information other than geometrical ones) attached to a part in various engineering activities like assembling and manufacturing [13] [14].

Used principally in centralized architectures, and within
Product modeling, features-based modeling [14] [13] is the emerging technology for describing semantic product data, which support the integration of design and engineering activities. It enriches the product data representation with semantic (parametric, functional, technological) information allowing more efficient communication between engineering processes than before.

Product modeling and simulation, is still an open area in distributed and heterogeneous systems. We believe that the main challenge remains methodological rather than technological, i.e. we believe that only a simple use (without methodology) of standards and technological tools (CORBA [14], STEP [17], etc) is not sufficient. More particularly in CIM environments, it is unusual to have a unique and sufficient representation (model) for all the CAD/CAE/CAM applications associated to the manufacturing product lifecycle. As part of the challenges, we can point out the issues of managing, with higher degree of interoperability, various representations (geometrical models) for the same physical object, in different and autonomous CIM environments. We must give a representation scheme associated to a paradigm (multiple models paradigm in our case) to facilitate the use of Software Engineering and Artificial Intelligence tools in design and implementation levels with the chosen standard tools like CORBA and STEP.

Our proposition takes into account both CIM requirements through the application paradigm and the principles of Software Engineering. The continuity of the approach, from specification to execution, and the generality of results are as important as the proposed paradigm (the work described in [15] is related to ours). Particular attention is given to new fundamental bases (mathematical formalization) that gives more general solution and constitute at least a methodological guide in the construction of communicating components.

III. BRIEF OVERVIEW OF AGENTS TECHNOLOGY

According to the agent paradigm, a software system is thought of as a set of pieces of software, called agents. An agent can possess a certain number of characteristics [16]:

- **Autonomous** - capable of acting without direct external intervention.
- **Interactive** - communicates with its environment and other agents.
- **Adaptive** - capable of responding to other agents and/or its environment to some degree.
- **Proactive** - goal-oriented, purposeful; does not simply react to the environment.
- **Intelligent** - state is formalized by knowledge (i.e., beliefs, goals, plans, and assumptions) and interacts with other agents using symbolic language.

- **Rational** - able to choose an action based on internal goals and the knowledge that a particular action will bring it closer to its goals.
- **Coordinative** - able to perform some activity in a shared environment with other agents; activities are often coordinated via plans, workflows, or some other process management mechanism.
- **Cooperative (or collaborative)** - able to coordinate with other agents to achieve a common purpose.
- **Competitive** - able to coordinate with other agents except that the success of one agent implies the failure of others (the opposite of cooperative).

Some of these characteristics are mandatory (such as autonomy, reactivity and proactivity); others are optional (such as intelligence, mobility, etc.)

Multi-agent Systems

A Multi-agent system is an environment where agents can exist, communicate, interact, and coordinate their work in order to achieve a common plan or goal. Tools have been developed for building such environments, for example madKit [20], IBM Aglets Workbench [12].

Speech-Act Theory

An important implementation issue in a multi-agent system is interaction. Since action in the system is usually goal-directed, many interactions are derived from goals. While working with distributed agents, it is important to design an expressive common language for communication with an agent-independent semantics, where agents can communicate with their peers by exchanging messages and interact together through explicit linguistic actions [8]. The agent communication language in a multi-agent system should be independent of the agents. In the multi-agent systems community speech-act [7] theory is one of the most common methods used to construct the linguistic layer and formalize the linguistic actions of agents. Speech Act Theory uses the concept of performatives to allow an agent to convey its beliefs, desires and intentions. The performatives are the speech-act component of the language and determine what an agent can “do” or “perform” with the content of the message. For example, performatives “assert,” “affirm,” “state,” convey a belief, performatives “ask,” “order,” “enjoin,” “pray,” or “command” convey a wish or a desire, and performatives “vow,” “pledge,” or “promise” convey an intention [11].

A speech-act language which is commonly used in the multi-agent community is KQML (Knowledge Query and Manipulation Language). KQML is a DARPA Knowledge Sharing Initiative contribution. KQML performatives play an important part in the language. KQML supports many different performatives.

IV. CONTRIBUTION

The network of CAD models is considered as a unified and unique representation of a given solid object. To keep consistency between CAD Models, which are different representations for the same physical object, a simple
application protocol is defined between Agents. In this protocol a central Agent is a server (called Initial Agent, associated to the Initial algebra) and the other Agents are clients. Each modification in the network is broadcasted later from the server for all client Agents.

Two types of consistency are managed in the network of Agents:
- Local consistency: it concerns the management of local modifications or constraints due to the local characteristics of the associated CAD Model without changing the remainder of the network of Agents;
- Global consistency: it is supported by all the network of Agents thanks to the broadcasting protocol; it affects fundamental modification associated to physical object. The fundamental properties of the initial algebra enable to give simple protocol.

For this purposes, each Agent contains three subagents:
- The Structure agent: it contains the physical representation of the object. First implementation was given as a class hierarchy that holds the object. Later this agent was designed as an interface for a given modeler or a database (repository);
- The Constraints agent: it holds knowledge of the local representation in a given type of CAD representation (CSG or BRep for example). It permits to solve local constraints and holds local consistency;
- The Interface agent: initially assigned for only physical object interaction and visualization, this subagent evolved to take into account dialogue management, broadcasting algorithm and translation. Thus it contributes to hold global consistency.

Some implementations were given for the three subagents (in Lisp, C and C++ languages). We show briefly the operational design with the Interface agent of the two works below:

**Interface agent:**
At the origin of this work was a principal objective of demonstrating that a tool like interactive planning, [10], can contribute to enhance the quality of the dialog and increase interactivity in CAD/CAM systems for different types of users, expert or not. The architecture of the Interface agent is based on cooperation between interactive planning (or dynamic planner), knowledge base management agent and a dialog manager. Later the same development of this agent was used in translating a CSG representation to a BRep representation for the same part (a mechanical object). The Interface agent shows that it can be used in various interactive and dynamic contexts [2].

The independence of the implementation of Interface agent is based on a First Order Predicates Language (FOPL) through defined primitives. Translation between CSG and BRep modelers is based on dynamic planning and normalized formulas which use Boolean operations, and are well defined from the notion of equivalence of terms in the initial algebra (see [22]).

The proposed approach is independent from the used (standard) tools. Current development is based on the use of Agents in the environment of both CORBA and STEP standards [23-24], other environments can also be used under the same multiple modeling approach (e.g. J2EE [18] and XML [19]).

**Hypotheses**
We assume that the communication is initiated first by an Interface Agent associated to a CAD model when it has just updated its local model. Then a notification message is sent to the Initial Agent. Subsequently a broadcast message is sent from the Initial Agent to all the Agents seeking the update of all local models. We also assume that not more than one global update can occur at a time.

The communication model is synchronous, which means that messages are not queued. This means the collaborative environment targeted here is of closely coupled type (also called Hot Distributed Collaborative). The protocol is centralized; no messages are exchanged between Agents. No priority is defined over the network of Agents by the Initial Agent. Table 1 shows the set of performatives derived to meet the communication requirements.
<table>
<thead>
<tr>
<th>Message</th>
<th>Source agent</th>
<th>Destination agent(s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Request Update</td>
<td>AGENT</td>
<td>INITIAL AGENT</td>
</tr>
<tr>
<td>ACK</td>
<td>AGENT</td>
<td>INITIAL AGENT</td>
</tr>
<tr>
<td>Update Start</td>
<td>INITIAL AGENT</td>
<td>AGENT</td>
</tr>
<tr>
<td>Master_Model_Update_Complete</td>
<td>INITIAL AGENT</td>
<td>AGENT</td>
</tr>
<tr>
<td>Local_Update_Complete</td>
<td>INITIAL AGENT</td>
<td>AGENT</td>
</tr>
<tr>
<td>Ready_for_Update</td>
<td>AGENT</td>
<td>INITIAL AGENT</td>
</tr>
<tr>
<td>Update</td>
<td>INITIAL AGENT</td>
<td>AGENT</td>
</tr>
<tr>
<td>Begin_Update</td>
<td>INITIAL AGENT</td>
<td>AGENT</td>
</tr>
<tr>
<td>Update_Start</td>
<td>INITIAL AGENT</td>
<td>AGENT</td>
</tr>
</tbody>
</table>

Table 1. The set of Messages (performatives)

Signification of messages
- **Request Update**: the Agent has just updated its local model and wants to inform the network of module agents.
- **Update Start**: update can be started.
- **Master_Model_Update_Complete**: the process of updating the master model is completed.
- **Local_Update_Complete**: the process of updating the local model is completed.
- **Ready_for_Update**: the local interface agent is ready to perform local update from the master model.
- **Update**: the Initial Agent informs the Agent that the Master model has been updated to set it in a ready state.
- **Begin_Update**: the Initial Agent notifies the Agent to start.
- **Update_Start**: informs the Initial Agent that no other update is ongoing.
- **ACK**: Acknowledge message to end local model update.

Specification of the protocol
We shall model our communication protocol by two Finite state machines synchronously coupled; one running in the Initial Agent (Server side) and the other running in each Agent (client side). We shall use the Mealy finite state machine type because it is the most suitable for modelling protocols.

The Agent machine is represented in figure 2. The signals of input or output of the machine are messages exchanged between agents. The loop denotes timeout condition which happens when an Agent wants to propagate a change in its local model while not yet being aware that the Master model is being updated due to change in another local model. The possibility of a retransmission after a timeout is not modelled.

The other Finite state machine (Figure 3.) is quite similar to the previous but with input and output signals exchanged. The right leg of the machine is repeated for each Agent. We can be certain that for all possible executions not more than two Agent machines can be in one of the states Q1 or Q2.

Though the states in two finite state machines hold the same labels, the tasks performed by the interface agents are different. For example in the state Q3 labelled ‘Updating Master model’, it means that the interface agent is being uploading the changes in its local model while the Initial agent is being receiving these changes to update the central model called ‘Master model’.
Figure 2. The Agent finite state machine

Figure 3. The Initial Agent finite state machine
V. CONCLUSION

The protocol described in this paper is intended to be simple so that the possibilities of application remain open. The same approach can be applied to other application areas where multiple models paradigm is operated.

Further work should be the support of transactional update of the master model and the local models alike, the possibility of deferred update, and the possibility of defining priorities among the agents either statically or dynamically under the assumption of cooperation.

REFERENCES