INT. J. PROD. RES., 2003, VOL. 41, NO. 8, 1621–1638

A collaborative agent-based infrastructure for Internet-enabled collaborative enterprises

WEIMING SHEN†‡*, ROB KREMER†, MIHAELA ULIERU† and DOUGLAS NORRIE†

The Internet has evolved very rapidly from an information space to a market space over the past few years. There is a tendency towards implementing real-world agent-based applications based on the Internet. This paper presents some results of our recent research work towards an infrastructure for Internet-enabled collaborative agent systems. The infrastructure and the related supporting services, components, prototypes and mechanisms are initially proposed and developed for Internet-enabled collaborative agent systems in all kinds of application areas, but they are primarily targeted for implementing Internet-enabled collaborative enterprises or supply chain management systems. The general collaborative agent system architecture with the basic communication and cooperation services, domain independent components, prototypes and mechanisms are described. The benefits of implementing Internet-enabled collaborative enterprises with the proposed infrastructure are discussed. A case study on multi-plant production planning is presented. Some important implementation issues are discussed.

1. Introduction

The Internet has evolved very rapidly from an information space to a market space over the past few years. According to the strategic research report from the Manufacturing Technologies Group (MTG) of the National Research Council Canada (http://web.mtg.nrc.ca/): 'Internet services spending in the manufacturing industry is expected to grow from $975 million in 1998 to $9.17 billion in 2003 (in Canada).' There is a tendency towards implementing real-world agent-based applications based on the Internet and web, and using Java as a primary programming language to realize this implementation. However, Internet-based (or web-based) agents are quite different from typed-message agents (Petrie 1996). Petrie (1996) provided an excellent overview of some related techniques proposed in the literature for connecting agents to the Internet, described several possible solutions and pointed out some existed problems to be solved. The proposed solution is primarily based on the JAT (http://cdr.stanford.edu/ABE/JavaAgent.html).

We do not intend to develop client–server type Internet-based multi-agent systems, but we are trying to develop collaborative agent systems using the Internet technology and the Java programming language, which raises a challenge to integrate typed-message agents with Internet-based client–server type agents. This paper

Revision received June 2002.
† University of Calgary, Calgary, Alberta, Canada.
‡ Present address: National Research Council Canada – IMTI, 800 Collip Circle, London, ON, Canada, N6G 4X8.
* To whom correspondence should be addressed. e-mail: weiming.shen@nrc.ca

International Journal of Production Research ISSN 0020–7543 print/ISSN 1366–588X online © 2003 Taylor & Francis Ltd
http://www.tandf.co.uk/journals
DOI: 10.1080/00207540210163946

presents some results of our recent research work in this area, including an Infrastructure for Collaborative Agent Systems (ICAS), a general Collaborative Agent System Architecture (CASA), basic communication and cooperation services, domain independent components, prototypes and mechanisms (section 3). This paper also discusses the benefits of implementing Internet-enabled collaborative enterprises with ICAS/CASA (section 4); and presents a case study on multi-plant production planning (section 5).

2. Research literature

Agent technology has been recognized as a promising approach to implementing collaborative enterprises (or virtual enterprises) and supply chains. Fox *et al.* (1993) proposed organizing the supply chain as a network of cooperating, intelligent agents. A similar proposal has been made by Swaminathan *et al.* (1996) using a multi-agent framework for modelling supply chain dynamics. Parunak and VanderBok (1998) proposed using three species of agents to model the supply chain: *Company agents* to represent the different firms that trade with one another in a supply chain; *PPIC agents* to model the production planning and inventory control algorithms used by company agents; and *Shipping agents* to model the delay and uncertainty involved in the movement of both material and information between trading partners. MetaMorph II (Shen *et al.* 2000a) proposed using a hybrid agent-based mediator-centric architecture to integrate partners, suppliers and customers dynamically with the principal enterprise through their respective mediators within a supply chain network. Brugali *et al.* (1998) proposed applying mobile agent technology to implement supply chain networks. However, all these systems are typical agent-based applications to manufacturing enterprise integration and supply chair management. The infrastructure proposed in this paper is for implementing agent-based Internet-enabled collaborative enterprises and supply chain management systems. The following paragraphs compare the proposed approach with some other related approaches in the literature.

The proposed collaborative agent system architecture (CASA) has some similarity to MIX (Iglesias *et al.* 1996), which is a federated architecture for the design of multi-agent systems that allows an integration of both connectionist and symbolic subsystems. MIX consists of the agent model describing the structure of an individual agent, and the network model defining the communication infrastructure used by agents. A distinction is drawn between agents offering network-related services (so-called network agents) and application agents. The network model offers various facilities, like a yellow-page service, coordination facilities, and rudimentary knowledge facilities supporting communication among heterogeneous information agents. However, the communication and cooperation services provided by CASA and ICAS are quite different from those proposed by MIX.

IMPS developed by Crow and Shadbolt (1998) is also an Internet-based multi-agent system, being a basic application of JAT for knowledge acquisition through the Internet. ICAS and CASA can be used for developing more complex Internet-enabled collaborative agent systems.

Some ICAS components (e.g. Interface Agents) and mechanisms (e.g. planning, scheduling and execution monitoring) are similar to those developed in RETSINA (Sycara *et al.* 2001). The RETSINA infrastructure consists of a system of reusable agent types that can be adapted to address a variety of different domain-specific problems. It focuses on agent architecture. Each RETSINA agent draws upon a sophisticated reasoning architecture that consists of four reusable modules for communication, planning, scheduling and execution monitoring. A collection of RETSINA agents forms an open society of reusable entities that self-organize and cooperate in response to task requirements. However, the ICAS provides more basic components and services for implementing Internet-enabled collaborative agent systems. RETISINA focuses on agent architecture, while ICAS focuses on the supporting infrastructure for collaborative agent systems.

Baker *et al.* (1997) discussed the feasibility of agent technologies and Internet communications in supporting the expanded online commerce (e-commerce) and described the potential support of AARIA's architecture, scheduling approach and simulation capabilities in developing Internet-based manufacturing systems (or supply chains). Although some interesting results of AARIA and related projects have been published recently, no further results on Internet-based system implementation have been reported.

It is not possible to include an extensive review in this paper due to space limitations. A recent review of agent system architectures, frameworks, infrastructures and related issues on agents can be found in Huhns and Singh (1998). Jennings (1999) provided a systematic analysis of advantages and disadvantages of agents. Wortmann and Szirbik (2001) presented an overview of applications of agent-based technologies to virtual enterprises. A state-of-the-art survey of agent-based manufacturing systems with a detailed discussion of key issues and an extensive annotated bibliography can be found in Shen and Norrie (1999). More detailed discussions can be found in Shen *et al.* (2001a).

3. An infrastructure for collaborative agent systems

The primary objective of this research work is to demonstrate that a generic architecture can be developed for multi-agent systems, with collaborative and adaptive capabilities, and with an efficient type of communication and cooperation infrastructure.

Figure 1 shows the proposed Infrastructure for Collaborative Agent Systems (ICAS). It can also be considered as a high-level architecture for collaborative agent systems composed of both vertical and horizontal functional layers. Each layer is comprised of entities that are considered to be agents, since in some measure they will exhibit autonomy and intelligence and communicate using messages. Entities such as databases and hardware, which might otherwise be regarded as non-agents, are here considered to be 'wrapped' with an agent wrapper so they

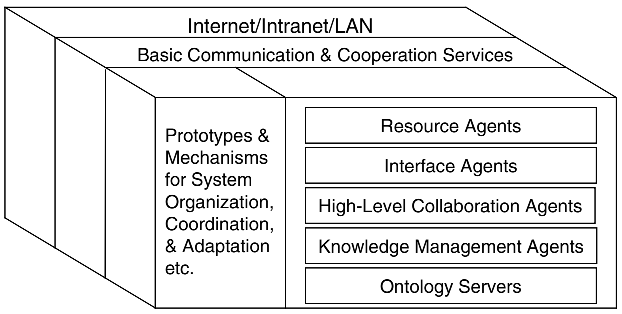
Internet/Intranet/LAN
Basic Communication & Cooperation Services
Prototypes & Mechanisms for System Organization, Coordination, & Adaptation etc.
Resource Agents
Interface Agents
High-Level Collaboration Agents
Knowledge Management Agents
Ontology Servers

Figure 1. Infrastructure for collaborative agent systems.

appear as agents, even in cases where their autonomy and intelligence may be low or nominal. Such agents are classified into the category of resource agents (or application agents), which includes all kinds of domain specific agents of a real world collaborative agent system. The reason for using a layered architecture is that it has been found, from both our own and other researches that specialization within an agent system yields a more efficient and comprehensible structure and facilitates implementation.

Agents are connected and communicate using the Internet/intranet or local networks. The basic communication and cooperation services are developed on top of the Internet and some lower-level network communication primitives (detailed in section 3.1), and thus provide a communication and cooperation infrastructure for collaborative agent systems. The proposed infrastructure includes a number of domain independent components or modules such as collaborative interface agents, high-level collaboration agents, knowledge management agents (or servers), and ontology server(s) etc (detailed in section 3.2), which utilize the basic communication and cooperation services. It also includes several prototypes and mechanisms for system organization, coordination and adaptation (detailed in section 3.3).

3.1. *Basic communication and cooperation services in CASA*

Communication among agents in CASA is supported by the Basic Communication and Cooperation Services layer, which offers a generic communication and cooperation infrastructure for message passing, archiving (data, knowledge bases, and transaction histories), agent lookup, and remote agent access. It is an open system infrastructure that may be easily extended as the need for future services becomes apparent.

As shown in figure 2, the *server* acts as a central 'hub' for multi-agent conversations such that all participants may send messages directly to the server for point-to-point, multi-cast, and broadcast communication within the *cooperation domain* (a

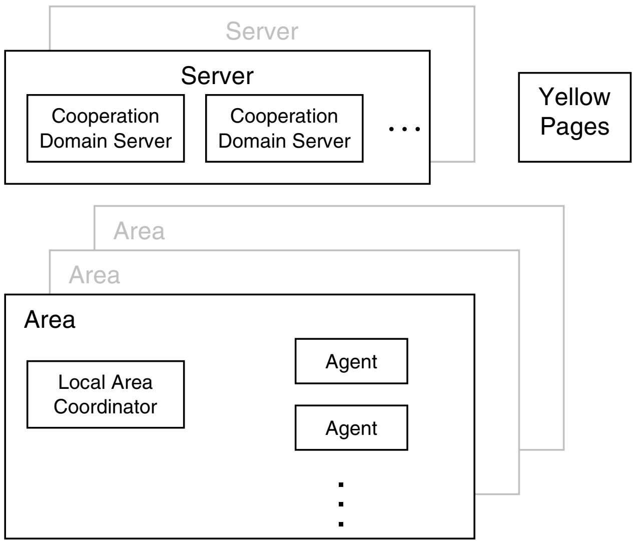

Figure 2. Basic communication and cooperation services.

group of agents working together on some task). Agents within a cooperation domain may also use the server to store persistent data that will permanently be associated with the conversation, giving the conversation a lifetime beyond the transient participation of the agents, as is often required. Servers may also store transaction histories for future playback of the chronological development of the conversation artefacts. Servers may perform all these tasks because all messages use KQML (Finin and Labrou 1997), which flexibly provides a basic semantic 'wrapper' on messages that may be otherwise domain specific: both the utility of generic services and the efficiency of domain specific languages are therefore provided.

Yellow Page servers (also called Yellow Page Agents, Yellow Pages, or YPs) allow agents to look up other agents by the services they offer. An area is defined nominally as a single computer (but could be a cluster of several computers, or a partition on a single computer). There is exactly one *local area coordinator* (LAC) per area, which is responsible for local coordination and tasks such as 'waking up' a local agent on behalf of a remote agent. All application agents reside in one or more areas.

Conversation policies are patterns of communication to be adhered to by collaborating agents. The functions of conversation policies are: first, they define causal-relation sequences of messages; and, second, they describe how agents react to these messages during interactions. The following protocols are defined for the architecture:

- *Registration*: for registering an agent as part of an area;
- *Advertisement*: for advertising agent services;
- *Search*: for querying about advertised services;
- *Service provision*: for requesting services from other agents;
- *Cooperation domain subscription*: for joining a cooperation domain;
- *Cooperation domain invitation*: for requesting an agent to join a cooperation domain.

Messages are needed to support interactions among agents. Messages are generically defined either as *request, reply* or *inform* messages; where *requests* are used to ask for the provision of a service, *replies* to answer requests, and *informs* to notify agents without expecting a response. Since these messages are too ambiguous for the definition of interaction protocols, other more meaningful message types are derived from general definitions of messages. For example, *advertisement_request* is derived from *request* for an agent to request to be registered (with a yellow page agent) as providing some particular agent service.

Conversation policies are represented by a notation consisting of nodes and directed arcs. One node represents one state of an interaction. There is one initial node marking the beginning of the conversation, and there might be one or more nodes representing terminal states. Arcs connect nodes, and represent the messages that could be chosen to proceed from one state to another. We have developed new approaches for Conversation Policies, using schemas for representation and execution and coloured Petri nets for design and verification. A detailed description of these approaches has been reported separately (Lin *et al.* 2000).

An alternative approach for clearly representing interactions and dialogues among collaborative agents has also been proposed (Flores-Mèndez *et al.* 1999). The detailed representation of conversation (or dialogue) among collaborative agents using the state transition diagram (STD) can facilitate the agent and agent system implementation.

3.2. *Domain independent modules/components*

Some of our recent work is focused on defining and developing several domain independent components such as collaborative interface agents, high-level collaboration agents, knowledge management agents (or servers), ontology server(s) and so on. As shown in figure 1, these components utilize the basic communication and cooperation services provided by CASA.

All these components will become domain specific when implemented in a specific application and be filled with domain specific information and knowledge. Detailed descriptions of these components are being written in separate reports and papers by other group members, and are out of scope of this paper. This section provides a brief introduction to these components.

3.2.1. *Collaborative Interface Agents*

An essential feature of a Collaborative Agent System for industrial or commercial application will be effective human/system interfaces. It is therefore proposed to incorporate Collaborative Interface Agents as an integral part of the system design. An interface software layer will be developed as a 'wrapper', so that an interface will appear to the system as just another agent. Concept mapping techniques (Kremer 1997) will be used to generate screen representations of information received by the interface. Constraint graph techniques (Kremer 1997) will provide support for manipulation of visual items and enable interface agents to interpret (the meaning of) such user actions. Categories of visual representations will be selected from recommendations of high-priority usage, put forward by the industrial partners for the proof-of-concept prototypes.

We consider that (domain independent) collaborative interface agents may have the following characteristics: communicative, semi-autonomous, collaborative, reactive, pro-active, adaptive, self-aware, and mobile. However, the implementation of interface agents in a real application does not need all these features.

3.2.2. *High-level collaboration agents*

Although several CASA components such as YPs and LACs also provide basic collaboration (cooperation) services, some complex large collaborative agent systems often need more high-level and domain specific collaboration services that cannot be covered by YPs and LACs. In order to meet such requirements, the high-level collaboration agent is proposed as one of the important components for CASA. An example of this type of collaboration agent can be found in the case study described in the following section, i.e. the HCA for coordinating the production panning of two factories. Such collaboration agents may be implemented by mediators as proposed and developed in our previous projects (Maturana *et al.* 1999). In most cases, they are static, but can also be implemented using dynamic mediators.

3.2.3. *Knowledge management agents*

Knowledge management is one of the most important issues in developing multi-agent systems. Typically, there are two approaches to knowledge management in multi-agent systems: knowledge is distributed among agents; some knowledge management agents are used to centralize knowledge management. Our approach to this issue is to combine the above mentioned two approaches, i.e. in addition to distributing some knowledge among agents, several knowledge management agents are developed for specific problems. A knowledge management agent is usually associated with one or more databases and knowledge bases. The key issues to develop knowledge management agents are to develop efficient mechanisms for knowledge acquisition, representation, learning and reasoning. In fact, a Yellow Page agent in CASA is a simplified knowledge management agent. In the case study presented in section 4, a knowledge management agent (KMA) is used for corporate level task decomposition and allocation.

3.2.4. *Ontology server(s)*

As one of the important components for our proposed infrastructure, an ontology server integrates a repository of ontologies, an inference and query engine and different translators. The ontology server structure and related mechanisms are initially domain independent. It becomes domain specific when it is filled with domain specific ontologies for a specific application, e.g. mechanical design, or production planning, etc. It is not very difficult to define the structure and mechanisms for an ontology server, but it is extremely difficult to develop and complete an efficient ontology server for an application domain. However, it has not been developed during our prototype implementation.

3.3. *Prototypes and mechanisms for system organization, coordination and adaptation*

The organization, coordination, adaptation and some other domain independent mechanisms, as shown in figure 1, have been separately proposed and developed during our previous research projects. This section provides a summary of these mechanisms.

As mentioned above, the primary objective of this research is to support collaborative software agents by providing easy-to-use domain independent services, which can be used to develop collaborative agent systems in all kinds of application areas. However, our research focus is still on the applications of this architecture in developing intelligent manufacturing systems.

3.3.1. *Agent-based mediator-centric organization*

An agent-based mediator-centric approach uses the federation architecture. In this particular type of federation organization, intelligent agents can link with mediator agents (also called mediators) to find other agents in the environment. Additionally, mediators assume the role of system coordinators by promoting cooperation among intelligent agents and learning from the agents' behaviour. Mediators provide system associations without interfering with low-level decisions unless critical situations occur. Mediators are able to expand their coordination capabilities to include mediation behaviours, which may be focused upon high-level policies to break decision deadlocks.

Mediators can use brokering and recruiting communication mechanisms to find related agents for establishing collaborative subsystems (also called virtual clusters). The brokering mechanism consists of receiving a request message from an agent, understanding the request, finding suitable receptors for the message, and broadcasting the message to the selected group of agents. The recruiting mechanism is a superset of the brokering mechanism, since it uses the brokering mechanism to match agents. Once appropriate agents have been found, these agents can be directly linked. The mediator then can step out of the scene to let the agents proceed with the communication themselves. To use these mechanisms efficiently, mediators need to have sufficient organizational knowledge to match agent requests with needed resources. The brokering and recruiting mechanisms generate two relevant types of collaboration subsystems. The first corresponds to an indirect collaboration subgroup, since the requester agent does not need to know about the existence of other agents that temporarily match the queries. The second type is a direct collaboration subgroup, since the requester agent is informed about the presence and physical location of matching agents to continue with direct communication. One common activity for mediators involved in either type of collaboration is interpreting messages, decomposing tasks, and providing processing times for every new sub-task. These capabilities make mediators very important elements in achieving the integration of dissimilar intelligent agents. Federation multi-agent architectures require a substantial commitment to support intelligent agent interoperability through mediators.

A generic model for the design of mediators, based on the specification of various meta-level activities, was proposed and implemented during the MetaMorph I project (Maturana *et al.* 1999).

3.3.2. *Task decomposition mechanism*

The task decomposition mechanism developed during the MetaMorph I & II projects is one of the core mechanisms needed in agent-based manufacturing systems. Through this mechanism, high-level tasks are decomposed initially by mediators acting at the corresponding information level. Each sub-task is subsequently distributed to agent clusters to determine the best solution plan. Mediators can learn dynamically from the agent interactions and identify clusters that can be used for the distributed resolution of tasks. Sub-tasks are then further decomposed, and allocated through negotiation among resource agents (Maturana *et al.* 1999, Shen *et al.* 2000a).

3.3.3. *Virtual clustering mechanism*

The virtual clustering mechanism was proposed and developed during the MetaMorph I project (Maturana *et al.* 1999). To work cooperatively, agents may form clusters that bond dissimilar agents into harmonious decision groups. Multistage negotiation and coordination protocols that can efficiently maintain the stability of these clusters are required. Each agent has its individual representation of the external world, goals and constraints, so diverse heterogeneous beliefs interact within a cluster through distributed cooperation models.

Through the virtual clustering mechanism, agents can be dynamically contracted to participate in a problem-solving group (cluster). In the situation where the agents in the problem-solving group (cluster) are only able to complete the task partially, the agents will seek help outside their cluster. This results in sub-clusters being formed for sub-tasks. This process is repeated, with sub-clusters being formed and then sub-sub-clusters as needed, within a dynamically interlinked structure. As the respective tasks and sub-tasks are solved, the related clusters and links are dissolved. However, mediators will store the most relevant links with associated task information for future reuse. This clustering process, as implemented, provides scalability and aggregation properties to the system.

3.3.4. *Partial agent cloning mechanism*

The partial agent cloning mechanism was proposed and developed during the MetaMorph I project. Through this mechanism, resource agents can be partially cloned as needed for concurrent information processing. These clone agents can then participate in virtual coordination clusters where agents negotiate with each other to find the best solution for a production task. A detailed description of this mechanism can be found in Maturana *et al.* (1999).

3.3.5. *Adaptation and learning*

Two fundamental learning mechanisms have been implemented in MetaMorph to enhance the system's performance and responsiveness, with mediators playing an essential role in both mechanisms (Maturana *et al.* 1999). First, a mechanism that allows mediators to learn from history was developed at the resource mediator level to capture significant multi-agent interactions and behaviours. Second, a mechanism for propagating the system's behaviour into the future is implemented to help mediators 'to learn from the future'.

A 'learning from history' mechanism based on a distributed case-based learning approach was developed for capturing agents' behavioural patterns at the resource mediator level and storing these in its knowledge base. Such knowledge is then reused for later manufacturing requests, through an extended case-based reasoning mechanism.

The main purpose of 'learning from the future' is to modify promissory schedules at the resource agent level for otherwise unforeseen perturbations and changes in production priorities on the shop floor. The forecasting process simulates the behaviour of the virtual model, which emulates the shop-floor activities. By partially projecting 'unpredictable behaviours' and agent interactions, the agent-based manufacturing system is able to correct its real-world model and provide more accurate plans.

4. Benefits of implementing Internet-enabled collaborative enterprises with ICAS

Although the Infrastructure for Collaborative Agent Systems (ICAS) is initially proposed as a general approach for developing Internet-enabled collaborative agent systems in all kinds of application areas, most components and mechanisms proposed and developed under this infrastructure are very useful and suitable for implementing Internet-enabled collaborative enterprises or supply chain management systems. This section discusses the benefits of implementing Internet-enabled collaborative enterprises with the ICAS components and mechanisms.

- *Cooperation Domain Servers (CDSs)*: in an Internet-enabled system for collaborative enterprises, one of the most important services is the customer-supplier or business-business negotiation. The *Cooperation Domain Servers* provide a natural way together with media for such negotiation. This method becomes more effective when it works together with the Virtual Clustering mechanism described in section 3.3.
- *Yellow Page Agents (YPs)*: because of the dynamics and complexity of Internet-enabled systems for collaborative enterprises, *Yellow Page Agents* will play a very important role for providing look-up services.
- *Local Area Coordinators (LACs)*: LACs can significantly facilitate the organization and management of the Internet-enabled collaborative enter-

prises. They are especially useful and important for widely geometrically distributed collaborative enterprises or the Internet-enabled supply chain of large international manufacturing enterprises.

- *Collaborative Interface Agents (CIAs)*: different types of user interfaces are needed for the Internet-enabled supply chain or collaborative enterprises: interfaces for customers to input orders; interfaces for marketing and operation managers, production managers, enterprise general managers, etc. *Collaborative Interface Agents* proposed and developed under ICAS can be employed to meet such requirements. Each type of interface may be composed of several modules of the general CIA model with optional components.
- *High-Level Collaboration Agents (HLCAs)*: this type of *collaboration agent* can be used to provide collaboration services that cannot be covered by the basic CASA cooperation services, e.g. for regional marketing coordinators, regional operation coordinators, local or regional production coordinators, etc.
- *Knowledge Management Agents (KMAs)*: managing a complex system for collaborative enterprises needs a lot of knowledge, ranging from customer requirements, marketing, product modelling, and project management etc. Several *knowledge management agents* may be developed to facilitate such management.
- *Ontology Server(s)*: for an Internet-enabled system for collaborative enterprises the ontology problem is very crucial because it is related to highly heterogeneous environments, e.g. different terminology used by multidisciplinary users; multiple communication languages such as KQML and FIPA-CAL; multiple knowledge/data/information interchange formats such as KIF and EDI; and even different natural languages, such as English and French, etc. Thus, one or more *ontology server(s)* should be employed to provide a standard or translation service for the system.
- *Virtual Clustering*: as mentioned above, negotiation is the primary process in a system for collaborative enterprises for raw materials (or parts) procurement, parts fabrication, parts transportation, and product assembly etc. A number of collaborative agents including collaborative interface agents for customers and managers, automatic software agents for cost calculation, production planning, conflict detection and resolution and so on are involved in this negotiation process. The *Virtual Clustering* mechanism provides an efficient way to form a virtual collaboration group to facilitate such negotiation.
- *Agent Cloning*: similar to the Virtual Clustering mechanism described above, the agent cloning mechanism will be very useful for some collaborative agents to be involved in several collaboration groups (clusters) simultaneously, which may significantly reduce the network communication load.
- *Adaptation and Learning*: a complex system for collaborative enterprises needs dynamically to adapt to its changing environment, including changing marketing conditions, collaborative enterprise configuration, intra-/inter-enterprise interactions, financial policies etc. The implementation of learning mechanisms, e.g. at the high-level agents such as LACs, CIAs, HLCAs and KMAs, can enhance the system performance.

5. A case study—multi-plant production planning

Several case studies using the proposed architecture and related components and mechanisms are underway in collaboration with industrial partners, including Internet-enabled supply chain management (Shen *et al.* 1999) and multi-plant production planning. This section provides a brief description of the case study on multi-plant production planning. Due to space limitations, this section only gives an extraction of the case study to show the functions of the above mentioned components in the prototype implementation.

5.1. *Conceptual description*

An internationally distributed manufacturing enterprise (or a virtual enterprise) in this case study is composed of a headquarters (with a Marketing and Operation Manager—MktOM), a Production Planning Centre (with a Production Manager), and two Plants (each with a Plant Manager) (figure 3). This case study can easily be extended to a large manufacturing enterprise composed of several production planning centres and more worldwide distributed plants. The production order used to test the prototype is as follows.

- 100 products B with due date D;
- One product B is composed of one part X, two parts Y and three parts Z;
- One part Z has three manufacturing features (Fa, Fb, Fc), and needs three operations (Oa, Ob, Oc).

5.2. *Scenario at a glance*

- MktOM receives a Production Order A from some customer for 100 products B with delivery due date D.
- MktOM sends the Production Order A to the Production Manager.
- The Production Manager finds some competent agent for task composition, then the Production Order A is decomposed into parts production requests.

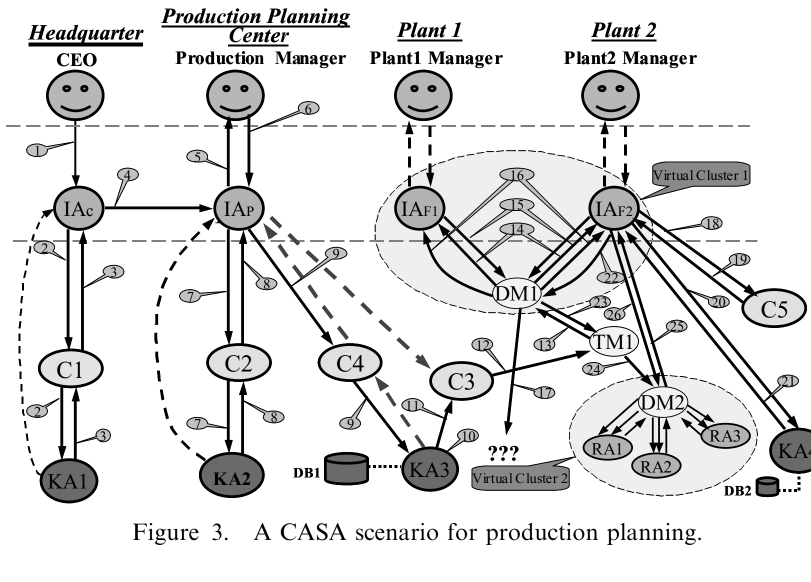

Figure 3. A CASA scenario for production planning.

- The Production Manager sends parts production requests to competent Factories for parts production.
- The Plant Manager receives a part production request, and finds a competent agent for further (sub-)task decomposition for decomposing a part production request into manufacturing features (with corresponding machining operations).
- The Plant Manager negotiates with resource agents for machining operations, awards machining operation tasks to suitable resource agents, and then sends related information back to Production Manager.

5.3. *Description of agents*

The agents involved in the multi-plant production-planning scenario are described as follows (see figure 3)

IAc	Interface Agent designed for MktOM.
IAp	Interface Agent designed for Production Manager.
IAf1 & IAf2	Interface Agents designed for Plant Managers 1 & 2.
C1, C2, C4 & C5	Collaboration Agent (implemented as LACs in CASA).
C3	Production Collaboration Agent.
KA1	Knowledge Management Agent containing information such as 'who does what?'
KA2	Knowledge Management Agent containing information such as 'who to collaborate for task decomposition?'
KA3	Knowledge Management Agent having skills (capabilities) for product-level task decomposition (using information from a Product Model Database—DB1).
KA4	Knowledge Management Agent having skills (capabilities) for part-level task decomposition (using information from a Product Model Database—DB2).
TM1	Template Mediator (containing knowledge and data for creating dynamic mediators).
DM1	Dynamic Mediator for coordinating a group of factories.
DM2	Dynamic Mediator for coordinating a group of resource agents.
RA1, RA2, & RA3	Resource Agents. One Resource Agent is a software agent used to model a manufacturing resource (e.g. a machine). It contains information about this machine's capability, its plan, its cost ($/hour), etc.
DB1	Product Model Database (Information about what parts comprise a product).
DB2	Product Model Database (Information about what manufacturing features a part has).

5.4. *Operational description*

5.4.1. *Placing a customer order*

Step 1. After receiving a Production Order A (100 products B with delivery due date D), MktOM places this order through the Interface Agent IAc.

Step 2. MktOM sends a request to C1 asking 'Who to collaborate for the Production Order A'; and C1 forwards this request to KA1.

Step 3. KA1 finds the necessary information, and sends a reply 'Contact IAp for Production Order A' to C1, which then forwards this reply to IAc. An alternative solution is that KA1 sends the reply directly to IAc.

Step 4. IAc sends the Production Order A (100 products B with due date D) to IAp.

Step 5. IAp displays the Production Order A to the Product Manager at the Production Planning Centre.

5.4.2. *Product-level task decomposition*

Step 6. The Production Manager places a request 'Find agent(s) for task decomposition and factories (shop floors) for production' through Interface Agent IAp.

Step 7. IAp sends a request 'Who to collaborate for task decomposition' to C2 which then forwards this request to KA2.

Step 8. KA2 finds the necessary information, and sends a reply 'Contact KA3 for task composition' to C2, which then forwards this reply to IAp. An alternative solution is that KA2 sends the reply directly to IAp.

Step 9. IAp sends the Production Order A (100 products B with due date D) (for task decomposition) to C4, which then forwards this message to KA3.

Step 10. KA3 does the task decomposition (using the information from its Product Model Database—which is connected with a Product Modelling or Engineering Design System). The Production Order A (100 products B with due date D) becomes the Production Request A (100 parts X, 200 parts Y and 300 parts Z with due date D).

5.4.3. *Virtual clustering*

Step 11. KA3 sends the Production Request A (100 parts X, 200 parts Y and 300 parts Z with due date D) to C3 for parts production. A Dynamic Mediator needs to be created for coordinating a group of factories (shop floors). As shown in figure 3 with dashed lines, an alternative solution is that KA3 sends a reply with decomposed parts information to C4, which then forwards this reply to IAp. IAp then sends the Production Request A (100 parts X, 200 parts Y and 300 parts Z with due date D) to C3. This alternative solution may increase some communication, but it can ensure the stability and reliability of the system.

Step 12. C3 sends a request for dynamic mediator creation to TM1 (Template Mediator) with the information about the Production Request A (100 parts X, 200 parts Y and 300 parts Z with due date D).

Step 13. TM1 creates a Dynamic Mediator DM1. (Here DM1 needs get some information about IAf1 and IAf2. We assume here DM1 also gets such information from TM1.) A Virtual Cluster composed of DM1, IAf1 & IAf2 1 is created.

5.4.4. *Negotiation and task allocation*

Step 14. DM1 sends out a Call for Bids for Production Request A to IAf1 and IAf2.

Step 15. IAf1 and IAf2 send bids back to DM1: (IAf1: I can do 100 parts X and 200 parts Y by due date D. IAf2: I can do 200 parts Z by due date D and 100 more parts Z by date D4 (late)).
Step 16. DM1 awards IAf1 and IAf2: (IAf1: Do 100 parts X and 200 parts Y. IAf2: Do 200 parts Z.)
Step 17. DM1 subcontracts 100 parts Z by due date D to some partner(s) outside of the enterprise.

5.4.5. *Part-level task decomposition and allocation*

Step 18. After receiving the production request for producing 200 parts Z, IAf2 should find some agent(s) for part-level task decomposition. (What features does a part Z have?) IAf2 sends a request 'Who to ask for part-level task decomposition' to C5.
Step 19. C5 sends a reply to IAf2 'Contact KA4'.
Step 20. IAf2 sends a request for part-level task decomposition 'What features does a part Z have?' to KA4.
Step 21. KA4 does the part-level task decomposition (using the information from its Product Model Database—which is connected with a Product Modelling or Engineering Design System): one part Z has three features Fa, Fb and Fc. KA4 sends the results to IAf2. Another dynamic mediator needs to be created for coordinating resource agents RA1, RA2 & RA3.
Step 22. IAf2 sends a request to DM1 for creating a dynamic mediator to form a virtual cluster for coordinating resource agents.
Step 23. DM1 sends a request for dynamic mediator creation to TM1.
Step 24. TM1 creates a Dynamic Mediator DM2, and also asks DM2 to 'Contact IAf2 whose address is xxxx' for the information about resource agents.
Step 25. DM2 sends a request to IAf2 for information about resource agents.
Step 26. IAf2 sends a reply to DM2 with the information about resource agents RA1, RA2 and RA3. A virtual cluster composed of DM2, RA1, RA2 and RA3 is formed.
Step 27. DM2 negotiates with resource agents RA1, RA2 and RA3 for manufacturing features Fa, Fb and Fc (correspondingly, operations Oa, Ob and Oc) using the Contract Net Protocol (Call for Bids, Bids, Awards, etc). When all operations are scheduled, DM2 sends an inform message to IAf2.

IAf1 should do the same as IAf2 does for part-level task decomposition and allocation. When all tasks are allocated, IAf1 and IAf2 each sends an inform message to IAp (for the Production Manager) which then sends an inform message to IAc (for MktOM).

This scenario has been implemented in a simulated environment to validate the proposed approach. Most ICAS/CASA components (e.g. LACs, CIAs, KMAs, etc) and mechanisms (e.g. task decomposition and virtual clustering) are used in the case scenario implementation. The Partial Agent Cloning mechanism is not used in the scenario, but it will be absolutely needed for a slightly more complex case.

6. Some implementation issues

Although the proposed approach is proved feasible through a number of case studies, its implementation in industry may still encounter many different difficulties.

This section discusses some important issues in implementing the proposed approach in real industrial applications.

6.1. *Integration of agent and Internet/Web technologies*

Although the original objective of the Internet was to send and receive pure textual information, the Internet gave birth to a new computation paradigm for implementing globally distributed systems for collaborative enterprises. It also provides a great opportunity for the further development and wide application of agent technology.

The popularity of the Internet is largely due to the influence of the Web, which has made resources on the Internet accessible and available to a mass population. Powered by the ever-improving information technologies, especially those promoting platform- and language-independent, such as HTTP (HyperText Transfer Protocol), HTML (Hyper Text Markup Language), XML (eXtensible Markup Language) and Java technologies. Through these technologies, the Web has provided us with another familiar interface and gave us a common 'look and feel' to information exchange.

With the open system architecture, and some interesting features such as flexibility, modularity, reconfigurability, scalability, and robustness etc, collaborative agents provide a promising way to implement systems for collaborative enterprises. However, due to security concerns (see section 6.5) and some technical difficulties (e.g. agent communication across a firewall) as well as the wide application of Internet/Web technologies, collaborative agents must be integrated with emerging Internet/Web-based technologies. Web-based user interfaces will be widely used to implement interface agents. Many ICAS components such CDSs, YPs, LACs and Ontology servers, etc will be implemented as Web servers or application servers. XML will be used for inter-agent communications (as detailed in the next section).

6.2. *XML for inter-agent communication*

Since XML documents can be exchanged and used across dissimilar platforms and applications, when applications are related to business transitions, e.g. for electronic commerce, supply chain management or virtual enterprises, XML is a good choice as a standard for data exchange (Shen *et al.* 2001b). Although XML is still evolving and its supporting tools are still under development, it is being promoted by leading companies such as Microsoft (http://www.microsoft.com/xml/) and IBM (http://www.ibm.com/xml/). It will become a widely accepted international standard for all kinds of Internet-based applications. XML will be used for formatting messages among all collaborative agents. According to our recent R&D work in other related projects (Shen *et al.* 2001b), XML holds several advantages over traditional Java Object Serialization, including:

- XML is easily understood and modified by both humans and agents, while Java Serialization is unreadable by humans;
- XML is platform and implementation neutral. This allows for agents to be implemented in virtually any language and any platform and still maintain 100% interoperability;
- XML is extensible by design, and offers a natural versioning mechanism. As new agents and data fields are introduced to the system, legacy agents who do not use these new options do not have to be recompiled and redeployed. This

type of version control comes very easily with XML (in the form of XML Namespaces), while it requires much more effort to implement with Java Object Serialization.

6.3. *Legacy system integration*

The product design and production management systems used by today's manufacturing enterprises consist of a set of separate applications, each for a different part of an enterprise's business, e.g. product design, process planning, scheduling and execution control. For example, Capacity Analysis software determines a Master Production Schedule that sets long-term production targets. Enterprise Resource Planning (ERP) software generates material and resource plans. Scheduling software determines the sequence in which shop floor resources (people, machines, tools, materials, etc) are used in producing different products. A Manufacturing Execution System (MES) tracks the real-time status of work in progress, enforces routing integrity, and reports labour/material claims. Most of these applications are legacy systems developed over the years. These existing legacy systems provide little interoperability among isolated individual applications, but they need to be integrated into a distributed system for collaborative enterprises.

Using the proposed approach, these legacy systems will be integrated into an open environment. Traditional technologies do not have a satisfactory solution to this problem. Agent technology provides a natural way to reuse legacy systems, provided that they possess adequate programmable interfaces (APIs). In this case, the systems can be encapsulated into agents, and integrated into more complex systems (Shen and Norrie 1999).

6.4. *Heterogeneous computing environments*

The proposed infrastructure for Internet-enabled collaborative enterprises needs to accommodate highly heterogeneous software and hardware environments. Such heterogeneous computing environments may operate in different computing platforms, run software developed with different programming languages, and represent data with different models and representation languages. Software agents are particularly suitable for implementing distributed collaborative systems across heterogeneous computing environments due to their important features, such as open system architecture, multiple communication protocols, and self-configuration, etc. Such implementations are further facilitated by platform-independent programming languages, such as Java, and platform-independent agent development tools (e.g. most Java-based agent building tools (http://www.agentbuilder.com/AgentTools/)).

Our initial prototype was implemented using the Java programming language. FIPA-compliant platforms such as FIPA-OS (http://fipa-os.sf.net/) and JADE (http://jade.cselt.it/) are being considered for the implementation of future prototypes and pilot applications. These platforms have been validated in heterogeneous computing environments (http://www.agentcities.org/).

6.5. *Security and privacy*

A major concern of implementing Internet-based systems is the assurance that proprietary information about the intellectual property owned by the organization or information about the company operations is available only to authorized individuals. Internet-based manufacturing involves sharing intellectual property in the form of detailed engineering and manufacturing information as well as competitive information in the form of order and costing details. For general acceptance of the Internet-based manufacturing approach, the secrecy of the proprietary or competitive information must be maintained.

In addition to maintaining secrecy, Internet-based manufacturing must accommodate the privacy of the individuals and organizations involved in collaborative manufacturing activities. Gathering and processing information about the activities of individuals or groups while managing or operating processes or machinery via computer networks can provide a great deal of detail concerning the ways in which the individuals interact, as well as process-related information. In a highly competitive manufacturing environment, we must ensure that information about the operations of, or the information provided by, individuals or organizations is only shared in a fashion dictated by those involved. This issue is addressed more in details in Shen *et al.* (2000b).

7. Conclusions

Our previous research experience and some results of our recent research work related to collaborative agents have shown that agent-based approaches potentially offer many advantages for implementing collaborative enterprises or supply chains on the Internet. These advantages include open and dynamic system architecture, modularity, reconfigurability, scalability, upgradeability, and robustness (including fault recovery).

The Collaborative Agent System Architecture (CASA) and the Infrastructure for Collaborative Agent Systems (ICAS) are initially proposed as a general approach for Internet-based collaborative agent systems. The theoretical analysis and the preliminary results of case studies have shown that the proposed architecture and infrastructure with domain-independent components and mechanisms are quite suitable for implementing Internet-enabled systems for collaborative enterprises.

Although the infrastructure and related services are proposed and developed for Internet-based collaborative agent systems, and several types of agents (e.g. YP agents and ontology servers, etc) are Internet-Web-based, most agents are typed-message agents using peer-to-peer communication. Furthermore, the integration of typed-message agents with Internet-based agents is one of the key issues (objectives) of our ongoing projects.

Other ongoing research work includes the development of proposed domain independent components and mechanisms into generic software modules for related applications. More case studies are underway in collaboration with industrial partners, initially in laboratory prototype form, and then these prototypes will be transferred into real industrial applications.

References

Baker, A. D., Parunak, H. V. D. and Erol, K., 1999, Agents and the Internet: infrastructure for mass customisation. *IEEE Internet Computing,* **3**(5), 62–69.

Brugali, D., Menga, G. and Galarraga, S., 1998, Inter-company supply chains integration via mobile agents. In G. Jacucci (ed.), *Globalization of Manufacturing in the Digital Communications Era of the 21st Century: Innovation, Agility and the Virtual Enterprise* (Kluwer), pp. 43–54.

Crow, L. and Shadbolt, N. R., 1998, IMPS—Internet agents for knowledge engineering. *Proceedings of the 11th Knowledge Acquisition Workshop,* Banff, Alberta, Canada. Also available at http://ksi.cpsc.ucalgary.ca/KAW/KAW98/crow/

Finin, T. and Labrou, Y., 1997, KQML as an agent communication language. In J. M. Bradshaw (ed.), *Software Agents* (Cambridge, MA: MIT Press), pp. 291–316.

Flores-Méndez, R. A., Wijngaards, N. and Kremer, R., 1999, Interaction protocols for a multi-agent system architecture. *Proceedings of KAW'99,* Banff, Alberta, Canada. Available at http://sern.ucalgary.ca/KSI/KAW/KAW99/

Fox, M. S., Chionglo, J. F. and Barbuceanu, M., 1993, The integrated supply chain management system. Internal Technical Report, Department of Industrial Engineering, University of Toronto, Canada.

Huhns, M. N. and Singh, M. P., 1998, Agents and multiagent systems: themes, approaches, and challenges. In M. N. Huhns and M. P. Singh (eds), *Readings in Agents* (San Francisco, CA: Morgan Kaufmann), pp. 1–24.

Iglesias, C. A., Gonzalez, J. C. and Velasco, J. R., 1996, MIX: a general purpose multiagent architecture. In M. Wooldridge, J P Muller and M. Tambe (eds), *Intelligent Agent II,* Lecture Notes in Artificial Intelligence 1037, pp. 251–266.

Jennings, N. R., 1999, Agent-based computing: promise and perils. *Proceedings of the 16th International Conference on Artificial Intelligence (IJCAI-99),* Stockholm, Sweden, pp. 1429–1436.

Kremer, R., 1997, Constraint graphs: a concept map meta-language. PhD Thesis, Computer Science Department, University of Calgary, Canada. Also available at http://www.cpsc.ucalgary.ca/˜kremer/dissertation/

Lin, F., Norrie, D. H., Shen, W. and Kremer, R., 2000, Conversation specification—a schema-based approach to specifying conversation policies, In F. Dignum and M. Greaves (eds), *Issues in Agent Communication,* Lecture Notes in Computer Science 1916 (Springer), pp. 193–204.

Maturana, F., Shen, W. and Norrie, D. H., 1999, MetaMorph: an adaptive agent-based architecture for intelligent manufacturing. *International Journal of Production Research*, **37**(10), 2159–2174.

Parunak, V. and Vanderbok, R., 1998, Modeling the extended supply network. *Presented at ISA-Tech'98,* Houston, TX. Available at http://www.erim.org/˜vparunak/isa98.pdf

Petrie, C., 1996, Agent-based engineering, the Web, and intelligence. IEEE Expert, 11(6), 24–29. Also available at http://cdr.stanford.edu/NextLink/Expert.html

Shen, W. and Norrie, D. H., 1999, Agent-based systems for intelligent manufacturing: a state-of-the-art survey. *Knowledge and Information Systems, an International Journal,* **1**(2), 129–156. An extended version in HTML format is available at http://isg.enme.ucalgary.ca/publication/abm.htm

Shen, W., Ulieru, M., Norrie, D. H. and Kremer, R., 1999, Managing Internet enabled supply chain with CASA. *Proceedings of Autonomous Agents'99 Workshop on Agent Based Decision-Support for Managing the Internet-Enabled Supply-Chain,* Seattle, WA. Available at http://www.research.ibm.com/CoopDS/Agents99/shen.pdf

Shen, W., Maturana, F. and Norrie, D. H., 2000a, MetaMorph II: an agent-based architecture for distributed intelligent design and manufacturing. *Journal of Intelligent Manufacturing*—Special Issue on Distributed Manufacturing Systems, **11**(3), 237–251.

Shen, W., Lang, S., Korba, L., Wang, L. and Wong, B., 2000b, Reference architecture for Internet based intelligent shop floors. *Proceedings of SPIE,* Vol. 4208, pp. 63–71.

Shen, W., Norrie, D. H. and Barthes, J. P., 2001a, *Multi-Agent Systems for Concurrent Intelligent Design and Manufacturing* (London: Taylor & Francis).

Shen, W., Brooks, C., Li, Y., Lang, S. and Wang, L., 2001b, XML-based message services for Internet-based intelligent shop floors. *Proceedings of SPIE,* Vol. 4566, pp. 135–144.

Swaminathan, J. M., Smith, S. F. and Sadeh, N. M., 1996, A multi-agent framework for supply chain dynamics. *Proceedings of NSF Research Planning Workshop on AI & Manufacturing,* Albuquerque, NM.

Sycara, K., Paolucci, M., Van Velsen, M. and Giampapa, J. A., 2001, The RETSINA MAS infrastructure. Technical Report CMU-RI-TR-01-05, Robotics Institute, Carnegie Mellon University. Available at http://www.ri.cmu.edu/pubs/pub_3509.html

Wortmann, H. and Szirbik, N., 2001, ICT issues among collaborative enterprises: from rigid to adaptive agent-based technologies. *Production Planning and Control,* **12**(5), 452–465.