AGENT BASED DYNAMIC INFORMATION GATHERING AND ORGANIZATION FOR DISTRIBUTED PRODUCT DEVELOPMENT

Shusheng Zhang and Yaqi Chen

College of Electro-Mechanical Engineering
Northwestern Polytechnical University, P. R. CHINA
zssnet@nwpu.edu.cn, chenyaqi@263.net

Weiming Shen

Integrated Manufacturing Technologies Institute
National Research Council Canada, CANADA
weiming.shen@nrc.gc.ca

Hamada Ghenniwa

Dept. of Electrical and Computer Engineering
University of Western Ontario, CANADA
hghenniwa@eng.uwo.ca

The concept of Web-based Visualized Product Structure Tree (VPST) has been proposed to realize a platform-independent, visualized and integrated product information sharing. This paper extends this concept by developing an agent based dynamic information gathering and organization framework to support updating the VPST dynamically. The objective is to enable different users to perform real time collaboration over the Web. The proposed framework is composed of three modules: Information Gathering, Information Organization Server and Information Sharing Server. Each of them is composed of a number of agents. The paper presents a detailed design and implementation of the proposed framework and its three modules.

1. INTRODUCTION

In distributed product development environments (for Virtual Enterprises or Internet based design and manufacturing), each participant (group member or enterprise member) needs to continuously exchange product information with others to keep the development process progress smoothly. Each participant may work in different software environments or hardware platforms and may have different interpretations to the exchanged information. Therefore the information sharing methods must efficiently resolve the platform and semantic heterogeneity.

Drawings are the most frequently used information interchange medium in enterprises. As each member obeys the same national or industry standard, the drawings' semantics is recognized by everyone and therefore with the same meanings. In drawings, 3D models have the advantages of visualization and can hold more information than other formats. Therefore 3D models are very suitable for information sharing and would be taken as the main carrier for integrating other shared information.

The authors previously proposed a strategy of integrated and visualized sharing approach based on the Web-based Visualized Product Structure Tree (VPST) (Zhang

et al., 2003). VPST is a tree like information structure based on the assembly structure of products. This structure can be used to integrate the product lifecycle related information. Each node in VPST is a part or component and represented with its 3D model. A branch in the tree denotes the semantic relation between components or parts, such as assembly relation and so on. Each 3D model is visualized using Web-enabled VRML (Virtual Reality Markup Language). The hyperlink mechanism of VRML is used to organize all kinds of related product information centered by 3D models. Each participant can logon on to Information Sharing Server using any Java-enabled Web browser to view the VPST. Each participant can expand the VPST nodes and click on a part or component to open its information view. Visualized information representation and hyperlink mechanism make the product information be presented in front of participants in a visual and direct manner with unambiguous meanings. Semantic and platform heterogeneity can be addressed effectively. Therefore this strategy is very suitable for product information sharing in Virtual Enterprise environments.

In order to update the VPST dynamically in a way that enables different users to perform real time collaborative product development over the Web, this paper proposes an agent based dynamic information gathering and organization framework. This paper begins with a brief literature review in Section 2. Section 3 introduces the proposed framework for dynamic product information gathering and organization. Sections 4, 5 and 6 respectively describe the three major modules of the proposed framework: Information Gathering, Information Organization Server, and Information Sharing Server. Section 7 briefly presents a prototype implementation. Section 8 concludes the paper with some perspectives.

2. RELATED WORK

Currently, the Web is being viewed as the most advanced information system deployed on the Internet. It provides designers with the means to combine multimedia information and to publish information relevant to the different stages of the design process. This has motivated the adoption of the Web as a collaborative design platform. It is now playing increasingly important roles in developing collaborative product development systems. It can be used as a medium to share product data, information and knowledge among members of the design team (Toye et al., 1993). It can be integrated with other management technologies to manage projects and product data effectively and collaboratively (Numata, 1996). Although there are a number of Web-based collaborative design frameworks available (Pahng et al., 1998; Numata, 1996; Huang and Mak, 2000; Roy and Kodkani, 1999; Cheng et al., 2001), most of them are still in the stage of prototype development or proof-of-the-concept. Recently, some leading CAD software vendors have released Internet/Web based versions of their software products with limited functions. A comprehensive review of related projects and systems can be found in (Shen and Wang, 2003). The rest of this section provides a review of the projects that are the most related to our work reported here.

InfoSleuth (Nodine et al., 1999) is a system of collaborating software agents that can be configured to perform active information gathering and analysis tasks across heterogeneous information sources. InfoSleuth agents advertise semantic constraints about themselves to InfoSleuth brokers using global domain ontology. The

InfoSleuth brokers can find the right agents according to these constraints. A network of information sources contains data resources, at varying levels of abstraction, which provide partial evidence or facts for elements in the ontology. Resource agents wrap and actively monitor external information sources. Each resource agent is responsible for extracting content from the source and registering it to one or more domain ontologies. An InfoSleuth application instance consists of a network of agents capable of performing information routing, analysis, extraction, and integration.

Pattison and Phillips (2001) proposed an architecture for the implementation of information view coordination and several coordination techniques. This architecture uses the MVC (Model/View/Controller) architecture to generate multiple views from a single data model. This approach provides for portability of the view across, and its simultaneous display on, multiple graphics platforms. Using this approach, coordination between views is mediated by coordination components, which look for state changes in an observed view and trigger resultant changes in a dependent view.

3. A FRAMEWORK FOR DYNAMIC VPST INFORMATION SHARING

The main idea of agent-based dynamic information gathering and organization is to use software agents to dynamically collect product information in a distributed environment on behalf of the development teams. In the context of this paper, an agent can be defined as a software system that communicates and cooperates with other systems to solve a complex problem, which is beyond the capability of each individual software system (Shen et al., 2001). These agents can automatically organize gathered information into information views of VPST and update these information views as soon as new information is generated. Our ultimate goal is to let participants get the real time product development information so that they can collaborate in a more effective way.

From information generated to the formation of VPST, the processing process can be divided into three main stages: product information gathering, product information organization and product information sharing. The product information gathering stage mainly focuses on the collection of dynamically generated product information in participants' local working environments. The product information organization stage mainly focuses on organizing gathered information into VPST. In the stage of information sharing, participants can get access to the shared product information in VPST. Corresponding to the three stages, the proposed framework is composed of three modules as shown in Figure 1. Each of them is composed of a number of agents.

Figure 1. System structure

The abbreviations used in Figure 1:

- ICA: Information Collection Agent;
- FTA: Format Transformation Agent;
- CSA: Coordination/Scheduler Agent;
- SMA: Structure Management Agent;
- IOSA: Information Organization Service Agent;
- HMA: Hyperlink Management Agent;
- IOA: Information Organization Agent;
- UNA: Update Notification Agent;
- CMA: Configuration Management Agent.

In Figure 1, the proposed framework is composed of three modules: Information Gathering, Information Organization Server (IOS) and Information Sharing Server (ISS). Product information gathering task is performed at the local development environment of each participant. Product information organization task is performed in the IOS and the service of product information sharing is provided by the ISS. The Information Gathering module is located in the local working environment of each participant. The IOS and ISS may be located in the local site of the leader enterprise of VE (Virtual Enterprise). These three modules communicate with each other through the Internet/Intranet/Extranet (IIE).

In the proposed framework, each function unit (such as Coordination/Schedule Agent) is a multi-agent system. In order to reduce the communication overload between agents and simplify the agent's internal structure, we employ the "mediator" policy in coordination theory (Malone, 1994) to coordinate the actions between agents. Each multi-agent system is composed of one master agent and several slave agents. Master agent manages the coordination and communication of slave agents. Agents communicate with each other using SOAP (Simple Object

Access Protocol) messages, which are composed of request parameters and operation data.

In the following sections, we will present these three modules in details.

4. PRODUCT INFORMATION GATHERING

Individual participants work in their own local working environments and communicate with each other through the IIE. The shared product information should be gathered from these local working environments.

In commonly adopted methods, the task of gathering information and sending it to the information sharing module is performed either by information processing specialist or by the information producer. In a Web-based collaborative product development environment, there is a huge amount of information to be shared among team members. Neither information processing specialists nor information producers can manually perform this job efficiently. To deal with this issue, we employ software agents to collaborate with the participants to perform this task effectively.

In the product development environment of each participant, product information is always managed by information management systems (e.g., a PDM system). Therefore agents can get product information from these systems. In the simplest situation, a product development member stores the information into a fixed location in their working environment (computer) from where agents can get the updated information.

In each local working environment there resides an ICA responsible for collecting product information in real time. The ICA decides when to start information collection process according to two situations: (1) a product development member saves the generated information; (2) a product development member submits a new version to the information management system. In the first situation, the ICA is configured so it knows that the product information is stored in a fixed location and the semantics of the information conforms to standard definition (domain ontology). When the participant saves (updates) product information, the ICA will know this by monitoring changes of information's system properties (such as the last saved time) and decide to start a new information gathering session. In the second situation, the ICA will also know that a new version is generated and will start a new information gathering session.

The information gathered by the ICA will be sent to the CSA for further processing. The CSA is responsible for managing the information gathering process and coordinating the communication among agents in the Information Gathering module. The CSA is also responsible for resolving the possible conflicts and sending the gathered information to the IOS. The CSA itself is composed of one Coordination Agent (CA) and several Scheduler Agents (SA). Each SA corresponds to an ICA in a member's personal working environment. Each information submission request from an ICA to the CSA is managed by a SA so that several ICAs can submit information simultaneously. The responsibility of the CA is to coordinate the operation of SAs and manage the communications between SAs and external agents (e.g. FTA). The results generated by CAD software are usually not Web-enabled (e.g., UGII generated part file and AutoCAD generated DWG file). Moreover, the file size may be very large (e.g. UGII part file may be several MBs) and it is not reasonable to transfer the file through the network directly. The SA will

send a format transformation request message to FTA through the CA. The transformation results will be sent back to the SA with the help of the CA.

FTA is an important member in the Information Gathering module. Its responsibility is to transform the product information into the corresponding Web-enabled format, since some file types, such as Microsoft Word document and UGII part file, are not suitable for direct transfer through the network or cannot be viewed in a Web browser. Each file type used to hold product information has its corresponding Web-enabled format. Some commonly used file types in product information sharing and their Web-enabled formats are listed in Table 1.

Table 1. Some commonly used file types and their Web-enabled formats.

Media type	Original data type	Web-enabled data format
3D model	Part	VRML
2D drawing	DWG, DXF	DWF
Image	BMP, TIFF	GIF, JPEG
Non-formatted document	TXT	XML, HTML
Formatted document	Word	PDF

FTA itself is composed of one master FTA and several slave FTAs. Each file type has at least one slave FTA and the most frequently used file format (e.g. UGII part file and Microsoft Word doc file) has several slave FTAs. The master FTA receives format transformation request messages from the CA. In simple situations, the master FTA assigns transformation tasks to corresponding slave FTAs according to the original file types in the SOAP messages. In complex situations, a negotiation based task allocation mechanism will be used.

Having finished the task of information gathering and Web-enabling, Information Gathering module will send the formatted information to the IOS by the CA.

5. PRODUCT INFORMATION ORGANIZATION

The task of product information organization is performed by IOSA located in IOS. IOSA asks SMA to set up the VPST's node structure according to a pre-defined product assembly structure (or based on assembly information from a STEP file of the product). SMA can create a node structure description file for each VPST and store it in IOS. This file will be loaded by the Java applet in Web browser to display VPST as an actual tree like structure. Figure 2 shows the content of a sample node structure description file.

In the node structure description file, tag <VPST> denotes a VPST; tag <component> denotes a component in VPST and tag <element> denotes a part. The nesting relation of tags represents the relations between components and parts. The *id* property of tag denotes the identifier of a part or component and its value is the same as the assembly expressed in the STEP file. This property is also used by SOAP messages sent from CA to identify the specified part. The *name* property denotes the semantic name of a part and its value will be displayed in VPST as the node name. The *href* property denotes the stored location of the main information view of the specified part or component. Java applet in Web browser loads this node structure description file and displays the VPST as an actual tree. The nesting relations of tags are mapped into the relations of tree nodes and *href* properties are mapped into hyperlinks that links to information views.

```
<VPST id="01" name="VPST-1" href="/product1/vpst/01.xml">
  <component id="011" name="chasis" href="/product1/vpst/011.xml">
    <component id="0111" name="comp1"
href="/product1/vpst/0112.xml"/>
    <element id="0112" name="part1" href="/product1/vpst/0115.xml"/>
    <element id="0113" name="part2" href="/product1/vpst/0116.xml"/>
  </component>
  <component id="015" name="base" href="/product1/vpst/012.xml"/>
```

Figure 2. Example of node structure description file

The specification of VRML provides the hyperlink mechanism. However, the dynamic change of the VRML entity's properties can only be achieved through external Java classes. We employ a HMA to perform these additional tasks. HMA adds hyperlink statements to a VRML file and links these hyperlinks to the specified information view pages. Then HMA will create an action response Java class to execute the property-changing action.

Some nodes have only assembly information but no product information if it is a component. Only parts can possess product information. Parts are the leaf nodes in VPST. Each part's development information can be organized into information views according to its type, such as CAD, CAM and CAPP and so on. Each information view (e.g. CAD, CAM or CAPP) is expressed in XML and constrained by corresponding XML Schema. Each information view has its own XML file and XML Schema file so that the changes made to one information view will not affect others. As the same information type usually has the same structure, the same information view shares the same XML Schema.

It is the IOA that takes the responsibility of product information organization. A SOAP message sent by CA consists of the component or part's id, product information and its semantic classification. Therefore IOA can place the information view at the right node by querying the VPST structure description file with component or part id. With the help of semantic classification the IOA can insert the information into the right information view. As for the Web-enabled file, only URLs are inserted into XML file. Text information can be recorded into XML file directly as tagged text. The main information view (e.g. 0115.xml in Figure 2) holds all hyperlinks to each sub information views.

We have discussed the set up process of VPST in details. However, in many cases agents are needed to perform real time VPST information view updating operations. The structure of VPST will not suffer from big changes in the long run of product development process. On the contrary, the information views corresponding to leaf nodes in VPST need to be updated in time in order to hold the latest product information.

Once the IOA receives SOAP messages from CA and finds that there already exists the corresponding content in the XML file, the updating operation will be performed. The IOA uses SAX (Simple API for XML) to parse the existing XML file and update the corresponding part by replacing old content with new information. If the newly generated information is in text format the above process must be performed. If the newly generated information is a drawing file or document file, it is only necessary to replace the old file with the new file and no change is needed to the information view XML file.

The formatted information will be sent to the ISS by the IOS and ready for sharing.

6. PRODUCT INFORMATION SHARING

Considering the system performance we employ an independent ISS. If the ISS is installed together with the IOS, the information sharing service will be influenced because of the heavy duty real time information processing.

Each information view in the ISS has at least one XSL (Extensible Stylesheet Language) file that acts as the display template for product information. XSL file is used to control the display format of information in XML file on a Web browser. Several XSL files can provide users with a personalized customization function.

When a participant wants to check the current state of a product development, he/she can open the information sharing URL in a Java enabled Web browser, logon to the information sharing page where the VPST is displayed by a Java applet. The user can expand the tree structure to find the related component or part. Simply by clicking on the node in the VPST, the 3D model and its related information will be displayed in an information window. A click on the part in 3D model displayed in the information window links to the part's information view and the VPST will synchronously opens the corresponding node. In the information view, all related information is organized around the 3D model. By clicking on the hyperlinks in the information page, the user can get access to all kinds of shared product information. For each Web-enabled file format there is a specially designed plug-in for a Web browser to display it.

The information sharing pages of the ISS also provide a personalized customization function. If the users are not satisfied with the display effect, they can configure a new display format using the customization function (displayed in the information page as a button which triggers a pop up Java applet window) or select a new XSL file from the pre-defined XSL set. In the customization window (Java applet window), the user can select an existing XSL file or create a new XSL file by customizing the preferred color or font and save the XSL file on the ISS. This new XSL file will be assigned to the specific user. When the user logon to the ISS, the display effect will be changed to the customized or selected XSL automatically. A special CMA together with the Java applet is responsible for this customization task.

The users can also set the default information view. By simply clicking on the "Set as default" button, the current information view will be set to the default view. After doing this, when a user logons to the ISS, the default information view will be displayed first and VPST will be expanded to the corresponding part.

There is also an UNA in the ISS, which communicates with the VPST display Java applet in a Web browser. UNA notifies the Java applet in Web browser to refresh the information views when information views in the ISS are updated.

7. PROTOTYPE IMPLEMENTATION

In the current implementation of the system, the three subsystems and the system processing logics are being developed as a FIPA-compliant [<http://www.fipa.org>] agent-based system using the JADE platform [<http://sharon.cselt.it/projects/jade/>].

The prototype system is set up in the product development environments of two manufacturing companies connected by the Internet. Information Gathering module resides in individual members' local sites. The IOS and ISS reside in the leader company. In the client side, only a Java-enabled Web browser (Netscape Navigator is preferred) and several plug-ins are needed. This prototype system works well in the test-running phase and will be put into practical use soon.

8. CONCLUSION

The system of agent based dynamic product information gathering and organization has been implemented into the Web based VPST. It has been proved to be efficient and useful in a collaborative product design process in VE environments. Note that most agents in the current implementation are highly coupled. Although such an implementation does not profit from some important features of agent technology (e.g., autonomy and proactiveness), the solution is reliable, stable and easy to be accepted by industrial partners. This also differs the proposed approach from other systems/projects mentioned in Section 2.

The paper reports some results of an ongoing project. Final prototype implementation and improvements are still underway. A generic framework for agent based dynamic information gathering and organization, including directory services, domain ontology for semantic integration, is being built. It is expected that the proposed generic framework can be used to implement distributed product development environments in different industrial sectors.

Acknowledgements

This work was partially supported by the China High-Tech R&D Program for CIMS through a Grant (No. 2002AA414210) and by the Ministry of Education of China through a Visiting Scholarship for the first author to visit National Research Council Canada and University of Western Ontario, Canada.

9. REFERENCES

- [1] Cheng, K., Pan, P.Y. and Harrison, D.K. "Web-based design and manufacturing support systems: implementation perspectives", *International Journal of Computer Integrated Manufacturing*, 2001, 14(1), 14-27.
- [2] Huang, G.Q. and Mak, K.L. "WeBid: A web-based framework to support early supplier involvement in new product development", *Robotics and Computer Integrated Manufacturing*, 2000, 16(2/3), 169-179.
- [3] Malone T.W. and Crowston, K. "The Interdisciplinary Study of Coordination", *ACM Computing Surveys*, 1994, 26(1), 87-119.
- [4] Nodine, M. and Fowler J. An Overview of Active Information Gathering in InfoSleuth, MCC Technical Report, 1998.
- [5] Numata, J. Knowledge Amplification: An information system for engineering management, Sony's Innovation in Management Series, Vol.17, Sony Corporation, Japan, 1996.
- [6] Pahng, G.D.F., Bae, S. and Wallace D. "A Web-based collaborative design modelling environment", *Proceedings of the IEEE Workshops on Enabling Technologies Infrastructure for Collaborative Enterprises (WET ICE'98)*, 1998, pp.161-167.
- [7] Pattison, T. and Phillips, M. (2001) "View Coordination Architecture for Information Visualization", *Proceedings of the Australian Symposium on Information Visualization*, Sydney, 2001.

- [8] Roy, U. and Kodkani, S.S. "Product modelling within the framework of the World Wide Web", IIE Transactions, 1999, 31(7), 667-677.
- [9] Shen, W., Norrie, D.H. and Barthes, J.P., Multi-Agent Systems for Concurrent Intelligent Design and Manufacturing, Taylor and Francis, London, UK, 2001.
- [10] Shen, W. and Wang, L. "Web-Based and Agent-Based Approaches for Collaborative Product Design: an Overview", International Journal of Computer Applications in Technology, 2003, 16(2/3), 103-112.
- [11] Toye, G., Cutkosky, M., Leifer, L., Tenenbaum, J. and Glicksman, J. "SHARE: A methodology and environment for collaborative product development", Proceeding of 2nd Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, Morgantown, WV, 1993, pp.33-47.
- [12] Zhang, S., Shen, W. and Ghenniwa, H. "A Framework for Internet Based Product Information Sharing and Visualization", Proceedings of ASME DETC/CIE 2003, Chicago, IL, 2003, DETC2003/CIE-48270.