Improving PDM Systems Integration Using Software Agents

Yinsheng Li[1], Weiming Shen[1], and Hamada H. Ghenniwa[2]
[1]*National Research Council Canada,* [2]*University of Western Ontario, Canada*
weiming.shen@nrc.ca

Abstract This paper addresses challenges in integrating distributed and heterogeneous Product Data Management (PDM) systems, introduces a virtual enterprise oriented agents-based integration solution, and presents an infrastructure with interactive and communicative agents as community and domain services to support distributed management of documents, BOMs, workflows and engineering knowledge, and secure communication in a collaborative product development environment. Considering the standardized Web-based agents and fundamental product data managed by PDM systems, the proposed paradigm is scalable and can also be used to integrate business systems in manufacturing enterprises.

1 INTRODUCTION

With the globalization of economy, an industrial organization should be able to react quickly and correctly to external changes and proactively manage the internal changes. Therefore, establishing an effective and efficient collaborative product development environment within an organization and with its partners becomes not only a requirement, but also an essential feature of its business model.

To that end, several tools and design paradigms have been proposed, such as Product Data Management (PDM). PDM is becoming an efficient facility for collaboration within and across industrial organizations. A PDM system takes product databases as basic information infrastructures; takes product structures (generally bills of materials) as a core organizing backbone; combines a series of engineering data and documents with each other,

The original version of this chapter was revised: The copyright line was incorrect. This has been corrected. The Erratum to this chapter is available at DOI: 10.1007/978-0-387-35621-1_43

K. Kosanke et al. (eds.), *Enterprise Inter- and Intra-Organizational Integration*

and allows for effective access, integration, management of product data and business workflows. Moreover, it supports integration between different applications, and facilitates collaborative activities among designers. A designer, with development workspaces in a PDM system, is able to take advantages of all factors related to a product lifecycle in a way that improves the quality of the end products, reduces dramatically trials, errors, costs and time-to-market of new products (Interleaf, 1998; ENOVIA, 1998; SDRC, 1996; PTC, 2002).

A number of challenges exist in designing and implementing PDM systems especially in cross-organization collaborative product development environments, such as virtual enterprises (VE), where there are different proprietary PDM systems (e.g., ProductManager by IBM (ENOVIA, 1998), IMAN by EDS (Interleaf, 1998), Metaphase by SDRC (SDRC, 1996), and Windchill by PTC (PTC, 2002)) preferred by their member companies due to certain historical and economical reasons. Therefore, providing integrity and consistency services for such collaborative environments with multiple PDM systems becomes very important. This integration will enable participants of a collaborative product development environment within and across organizations to access different product data seamlessly; extend their workflows to run through across the distributed collaborative workspaces; and come into being an integrated, efficient and unified product development platform for multidisciplinary collaborations (Peltonen et al, 1996; Jasnoch, Haas, 1996).

Although that some of the existing PDM systems have few successes in collaborating with one another, they still require tons of customization and high maintenance cost. This might be attributed to the complexity of the environment and to several key issues that need to be addressed in a comprehensive analysis and design approach, such as:

- Distributed management of documents, bills of materials (BOM), and workflows;
- Security issue on communication over the Internet within a single or multiple PDM systems (e.g., ProductManager transfers data over the Internet using built-in FTP with ASCII format);
- Security issue on access authorization based on distributed multidisciplinary teams;
- Privacy issue when sharing information.

In a previous project, a Web and coordination services-based integration paradigm for distributed PDM systems has been proposed and implemented (Li, 2000). Furthermore in this paper, we propose an agent-based approach to facilitate the integration in multiple PDM systems. With agents as interaction services layer, PDM systems get harmonized and combined with each other, especially in dealing with data consistency, communication security,

and business privacy. Moreover, this approach provides a "virtually" homogenous environment for the essential PDM services such as document management, product structure management and workflow management across multiple organization platforms.

The paper is organized as follows: Section 2 presents our agent-based PDM systems integration approach; Section 3 describes distributed transactions under such integration; and Section 4 gives some conclusions.

2 AGENT-BASED PDM SYSTEMS INTEGRATION

2.1 Integration system upon domain and community services

Previously in (Li, 2000), we proposed two types of integration facilities for multiple PDM systems, namely, domain services and community services. Domain services collectively act as brokers between local and remote PDM systems, in the sense to make communication transparent between them. Community services, like in public human services, are managed by a group with representatives from all partners, and provide common services across the PDM systems community. While both the domain and community services provide the integration services across multiple PDM systems, the underlying components of each PDM system, manage all their product structures, workflows and applications in local development workspaces as depicted in Fig. 1.

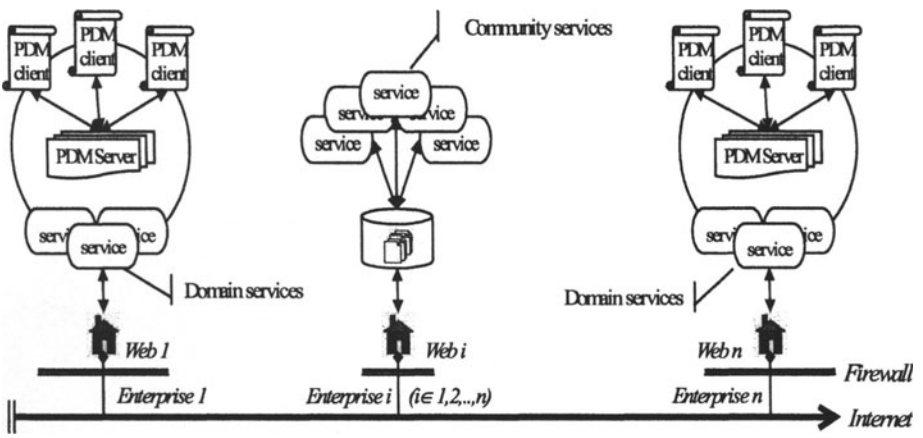

Figure 1: Integration schema based on community services

2.2 Agent-orientation and system integration

In order for PDM systems to enable and support collaborative product development, both community services and domain services require a technology that will package them transparently, and support interaction among them. The criteria for evaluating this technology include degree of automation, intelligence, independence and robustness. Especially, for community services, which require to be fair and impartial to get all partners' commitments to a public facility that is autonomous and neutral. To deal with these issues we propose an agent-based design for domain and community services. Agent-Orientation (AO) is a very promising design paradigm with a growing appeal in the distributed systems community. It is an emerging software engineering technology, which provides an adequate solution for modeling and designing collaborative distributed PDM systems. Particularly, AO is rich in providing several design features that are required for the integration of multiple PDM systems in a virtual enterprise or a supply chain, such as autonomy, proactiveness, flexibility, reconfigurability, etc. (Shen, et al., 2001; Ghenniwa, Kamel, 2000).

2.3 Integration architecture

As illustrated in Fig. 2, the proposed web- and agent-based architecture for integrating PDM systems is logically viewed as six layers, with each layer built on the next one:

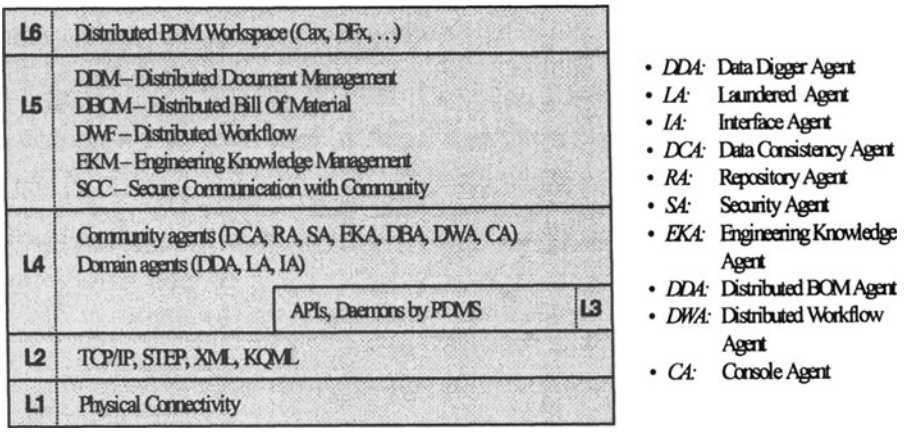

Figure 2: PDM systems integration architecture

1. Physical layer, the underlying network system connectivity with Intranet/Internet;
2. Communication protocols layer, including exchange protocols and specifications, such as TCP/IP for Intranet/Internet, STEP for inter-application product data exchange, XML for distributed content exchange, KQML for interaction between agent-based subsystems (Finin et al., 1993);
3. Resources layer, including programming resources in the product development environment, especially those professional APIs, Daemons provided by PDM systems vendors;
4. Integration services layer, i.e., agents layer, including community services and domain services;
5. Distributed transactions layer, including distributed management of documents, product structures (usually bills of materials), workflows and engineering knowledge, and secure communication within collaborative communities;
6. Applications layer, accommodating applications integrated into PDM system platforms such as CAD, CAM, DFM, DFA, and other application tools.

2.4 Agent identifications

This work focuses on the integration services layer. This layer provides services that seamlessly integrate PDM applications with physical connectivity resources, distributed communication protocols and application developing resources. Also, it supports the integration among the PDM applications at runtime. We categorize the roles of the agents based on the type of the service they participate in, either community service or domain service. Community agents usually locate at a Web server to provide public services and expertise across the community, such as virtual enterprise. These agents include:

1. Data Consistency Agent (DCA)
2. Repository Agent (RA)
3. Security Agent (SA)
4. Engineering Knowledge Agent (EKA)
5. Distributed BOM Agent (DBA)
6. Distributed Workflow Agent (DWA)
7. Console Agent (CA)

Domain agents usually locate at the partner enterprises and work together with other agents for system integration and information sharing. These agents include:

1. Data Digger Agents (DDA)
2. Laundered Agents (LA)
3. Interface Agents (IA)

The deployment scenario for these agents is shown in Fig. 3. The following paragraphs provide a brief description of the agents' roles.

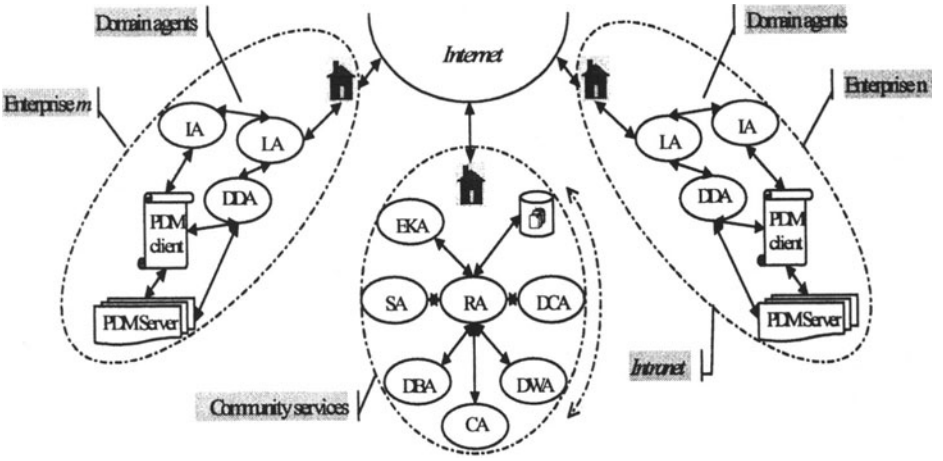

Figure 3: Integrating agents and their deployment

Data Digger Agents (DDAs) detect changes in product data and workflows, provide inputs of local partner PDM systems, and provide information to IAs. Almost all distributed PDM systems transactions involve DDAs. They run locally with PDM systems sides, get different knowledge and strategies to abstract, analyze and store the data about changes from specific local applications (e.g., PDM system clients, CAD, CAM, ...).

Laundered Agents (LAs) provide data format translation and standardization services, and facilitate intelligent communication between domain agents and community agents. As services on local ontology, they manage all information flow in and out of the environment.

Interface Agents (IAs) proactively assist their human-users in navigating the distributed PDM environment and identifying people and items that interest its human-users. IAs provide a notebook client for engineers, it can not only automatically capture engineers' design history and record them for future reuse with DDA's cooperation, but also allow engineers interactively write down design notes for reuse or reference by collaborators, and support online discussion with their remote associates efficiently and with a desired level of security.

Repository Agent (RA) is a foundational facility which maintains a complete meta-data model of the integration environment, including:

- Properties and status of agents, involved PDM systems and applications integrated into them;
- Member enterprises models built on personnel, activities, product information, and privileges;
- Standard elements and templates of distributed workflows and product structures (BOMs);
- Trigger rules and conflict resolution policies.

Data Consistency Agent (DCA) identifies any potential for data inconsistency in the distributed PDM environment. Then, it collaborates with DDAs and the RA to analyze situations, to determine the required update policies and procedures, and to execute the appropriate process.

Distributed BOM Agent (DBA) is responsible for BOMs decomposition and composition.

Distributed Workflow Agent (DWA) is responsible for workflow decomposition and composition.

Security Agent (SA) is a security service for agents' communication all over the community, similar to the Internet certificate authority.

Engineering Knowledge Agent (EKA) analyzes, translates and maintains design notes and other knowledge derived from DDAs and IAs at partners' domains, supplies engineers with these design experiences and history knowledge in the scope of the community, and allows engineering knowledge reuse and exchange. Engineers access these design knowledge and history by notebook-like IAs as if they were working in a single studio.

Console Agent (CA) acts as an administration interface console to all agents. By this interactive administration interface, administrators can initiate their specific environments when setting up, and allows further adjusting environment parameters and managing the system at runtime. All these administrations are through interactions between this agent and other agents in the community.

3 DISTRIBUTED TRANSACTIONS

3.1 Distributed documents management

In the current paradigm, the main concern for distributed documents management is data consistency, i.e., distributed version management and update. DCA, RA, DDA, and LA work together to keep documents consistent everywhere in the current distributed PDM community:

- The RA's database stores global descriptions about subject products, i.e., documents (in a PDM system, all files are documents), and their distribution across the distributed PDM community.

- When initiating the system, DDA in local web servers of member enterprises works together with PDM system services, retrieves the collaboration-related documents meta data (e.g., locations, and some other features) of product data from local PDM databases according to predefined mapping lists, which are retrieved by interacting RA.
- Before DDA interacts with outside, it firstly requests LA to translates and supplements submitted data into XML files with conventional schemas that receiver agents can recognize.
- DDA interacts with DCA in regular or real-time basis, and DCA uses the latest document meta data from RA to determine if subject documents have changed.
- As soon as a change is detected, DCA interacts with RA, provides the document with a new version based on conventional version policies, stores new meta data into the RA, finds out the impacted data objects, their locations and their DDAs by consulting product structure trees, bills of materials, and organization-organization relations lists, and then determines an updating program based on data consistency rules.
- Next, DCA contacts all the related DDAs in local web servers, notifies them to change the correlated data in their PDM databases by PDM system daemons and change mechanism, to keep them consistent with each other.

3.2 Distributed BOMs management

There are usually different BOM views in different life phases of a product. Moreover, in different PDM systems, there are different representations of BOMs even in the same phase. As a result, when integrating distributed PDM systems by agents, what distributed BOMs management concerns is transforming BOMs between different PDM systems. To this end, it is essential to define a global sharing BOM model, individual BOM models corresponding to the different product phrases and PDM systems, and furthermore work out the translating methods between the global BOM and individual BOMs, as an essential part of enterprise model in the RA.

DBA, RA, DDA, and LA work together to translate product structures in the community:

- When a BOM releases to another PDM system, the action is intercepted by DDA.
- DDA interacts with DBA through LA's standardization service.
- DBA interacts with RA, finds out the corresponding global BOM, individual BOMs and their mapping relationship, and executes the translation.

- The translated BOM is transferred to receiver DDA via LA, which forwards it to the goal PDM system (technically, usually by utilizing the exchange directory of the PDM systems and giving a fake notifying message to it, which responds the message to receive the data).

3.3 Distributed workflows management

Like BOMs, there are different representations of workflows in different PDM systems in a collaborative engineering environment. Therefore, when we get down to interlink these distributed, especially heterogeneous PDM systems, it is important to understand processes and their basic elements in different PDM systems, analyze distributed and collaborative design and engineering processes. And then, based on the understanding of them, a global workflow and a collection of translating relationships from the global workflow to specific workflows in related PDM systems are built as a main activity object of a common model of collaborative action in the RA. With above common object and translation lists, a distributed workflow can be executed remotely:

- When a distributed workflow, whose destination involves remote developer, is activated, DDA can find it by interacting with the PDM system at certain messages directory.
- DDA submits the workflow meta data to the DWA through LA.
- After interacting RA and getting the related global workflow template object and translating relationships, DWA analyzes, decomposes, and translates it into a specific common workflow as executing subject.
- Then, the DWA translates its to-be-executed element into a specific task that the predefined destination PDM system can understand, submits it to the PDM system to execute by LA/DDA.
- DWA holds and monitors the workflow execution. When a step is finished, DWA gets response from DDA/LA, and then gets to the next one.

3.4 Collaborative engineering knowledge management

The goal of the collaborative engineering knowledge management system is to combine distributed design workspaces into a virtual studio, allow engineers to collaborate and communicate their tips and decision-making evidences as convenient as they were in a same office. In this paper, all of its data detecting and collecting clients, end-user interfaces, and a centralized management console, have been improved as three standard agents: DDA, IA and EKA:

- DDAs detect predefined data changes and designers' operations, analyze and determine what should be submitted, recorded, and useful for reuse and communication.
- DDAs interact and submit captured information to IAs for users to confirm or modify. IAs display predefined items and provide engineers with interactive interfaces.
- DDAs get records and notes from IAs, whether they are input by engineers or edited content based on their submission.
- Periodically or activated by engineers' release, DDAs submit such data to EKA through LA's standardization.
- The formatted XML schema files are transferred to EKA through KQML. EKA checks, allocates and maintains them in a secure, orderly and classified manner. Sometimes, EKA interacts with RA to search related privileges and product features, and determine maintenance structure and access control list.
- With maintained information growing, a valuable facility for design knowledge referencing, reusing, and communicating is created. Now engineers can look up what have been recorded by all of them, what tips their co-workers have left, what is happening on their collaborators' sides, and with IAs supporting multimedia discussions, they can communicate with each other as they were working in the same office. Note that all above actions happen only they have valid priorities.

3.5 SA-based communication security between sites

Currently, even in a distributed collaborative product development environment using multiple PDM systems of the same type for product data sharing, there are few professional safeguarding considerations for site-site communications, without mentioning those happening between different types of PDM systems. The proposed paradigm provides a potential way to secure and maintain remote communications between PDM systems. With the SA, an Internet-like certificate authority can be built up by developing a special authenticating service within a community. Thereafter, the certificate authority can authenticate communicators, and with a public-key encryption solution (e.g., a symmetrical method, an unsymmetrical, or their integration), and offer them a certificate, a public key and digital signature. Finally, a set of secure infrastructures comes into being, and data security and integrity are subsequently guaranteed across all over the community.

4 CONCLUSIONS

In this paper, an agent-based solution for integrating distributed and heterogeneous PDM systems is proposed, and the details about the integration architecture, agent identifications and integrated mechanisms are presented. This is an attempt at applying agent technology to enhance the interoperability among existing PDM system entities. A promising collaboration structure as featured by community and domain agents has been applied to meet the requirements of a virtual enterprise. Due to the space limit, more details, particularly related to the system implementation issues, could not be included in this paper.

The proposed paradigm is an enhanced and systematic improvement of the previous research presented in (Li, 2000). Major related implementations were done during our previous research and the feasibility of multiple PDM systems integration based on community and domain services had been proved. As a result of software agents being good at wrapping up legacy systems as a methodology, the proposed paradigm is feasible in such an environment, by showing more autonomous, flexible, intelligent, and scalable features as various types of its agents come up with inherently. Nevertheless, a lot of research and implementation efforts are still necessary. Some of those identified include: determining the nature and model for each of community and domain agents; working out well-accepted implementation methods for each of distributed transactions that involves multiple even heterogeneous PDM systems; building data and security infrastructure and facilities for the agent-integrated layer; researching global product structure, workflow/processes; analyzing concurrent accesses and conflicts resolution strategies and facilities; selecting implementation technologies; and developing prototype systems.

Noted that the current architecture and system is towards integrating a product development environment based on PDM systems. However, a further extension can be expected. For example, with the similar methodology, many other enterprise business systems such as MRP II, ERP, SCM, and CRM etc., can be interlinked despite they are distributed and heterogeneous. At the same time when they are integrated and abstracted, with their agents having similar architecture and interaction protocols, a complete paradigm for virtual enterprise and supply chain can be expected.

5 REFERENCES

ENOVIA Corp. (1998), *ProductManager: Advanced Customization Environment,* Script, Reference and Operations Manual, Version 3.2.0, IBM Corporation.

Finin, T. Fritzon, R. McKay, D. McEntire, R. (1993), *KQML - A Language and Protocol for Knowledge and Information Exchange.* Tech. Report, University of Maryland, Baltimore, MD.

Ghenniwa, H. Kamel, M. (2000), *Interaction Devices for Coordinating Cooperative Distributed Systems,* Automation and Soft Computing, Vol.6, No.2, pp.173-184.

Interleaf Inc. (1998) *IMAN Online Help 2.1.0/*sunos5, Interleaf Inc.

Jasnoch, U. Haas, S. (1996), *A collaborative environment based on distributed Object-oriented databases,* Computers in Industry, Vol. 29, pp. 51-61.

Li, Y. (2000), *Integrated Infrastructures for Collaborative Development in a Virtual Enterprise, PhD Thesis,* Tsinghua University, Beijing, China.

Peltonen, H. Pitkänen, O. Sulonen, R. (1996), *Process-based view of product data management,* Computers in Industry Vol. 31, pp. 193-203.

PTC, (2002), http://www.ptc.com/products/windchill/

SDRC Corp. (1996), *Online Books for Metaphase 2.3,* SDRC Corp.

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