Separating Data and View: Support View-wandering Between Different Trades during Engineering Design

Liping Gao, Bin Shao, Ning Gu
Department of computing and Information Technology, Fudan University, Shanghai, P.R. China
{lipinggao, binshao, ninggu} @fudan.edu.cn

Abstract

Based on the analysis of the two forms during collaboration of engineering design, which are named as time collaboration and space collaboration, the paper introduces a TDM (Total Data Model) centered framework to support both time and space collaboration of hybrid CAD applications. By separating data and view the framework supports designers' view-wandering between different trades so that they can adjust their behavior to avoid space conflict. The maintenance of time consistency is achieved by utilizing AST (Address Space Transformation) algorithm, and in order to adapt AST to the new large-scale design environment, an effective garbage collection strategy is introduced. We have developed a prototype system to prove the validity, efficiency, feasibility of our framework and experiments result gives a clue of good prospect both for theory research and practical application.

Keywords: CSCW, Consistency Maintenance, Total Data Model, Heterogeneous Collaboration

1. Introduction

With the development of tense completion of current infrastructural industry, more and more requirements [1] have been brought forward to guarantee enterprises to survive in the world: The time between original conceptual design to final product and marketing should be as short as possible; The quantity of the product should be as excellent as possible; The product cost should be as low as possible. In order to meet the above needs, different computer-aided design (CAD) applications have been used to reduce cost and augment efficiency. Traditional CAD software is mainly used during the process of detail design. As a good auxiliary drawing and entity modeling tool, it plays an important role. However, with the further development of computer hardware and software, people began to realize that the using of computer to plot or model can not be called CAD; practical design is the design of total product with full lifecycle, including not only the functional design and structural design, but also the conceptual design, machining infrastructure and data management. In the process of design, designers coming from different specialties or different teams in the same specialty are constrained by concrete engineering rules. Yet these design processes are often concurrent, which necessarily leads to the outcome of design which is contradicted with engineering constraint rules. In order to prevent that, communication and collaboration tools or strategies must be provided for designers to design products with high coincidence, rationality and performance.

Single-user interactive computer applications are pervasive in design industry, almost used in every line. However, heterogeneous groupware system is in more great demand. There are two reasons that can be used to explain the phenomenon. First, the trade of CAD requires that CAD tools should provide convenient tools for designers to view, search, locate and switch during design process; so that designers coming from different departments may share information by *view-wandering* between different trades and starts communication more freely. However, tools used by designers of different trades are often dissimilar. For example, tabling designers may like to switch from a certain pipe of the 3D model to its corresponding arts workflow sketch of 2D model; second, not all designers coming from the same specialty are experienced with a particular application (or same application but not same version). Each participant may favor his or her own choice and be not familiar with the others'. Different single-user applications may have completely different interfaces, command sets and ways of customization, even if they are mostly compatible in terms of functionality. For example, consider popular CAD software such as AutoCAD, Inventor and 3DMax. One becomes a second-class citizen in the collaboration if one's favorite application is not selected for sharing [2]. Forcing all participants to use the same application is often counterproductive and results in group resistance.

What's more, the collaboration form in engineering design field is different from that of traditional group editing. In the process of group editing, data consistency can be achieved by only considering the relationship of data operations, which can be got by using time-related consistency maintenance algorithms such as Lock, Serialization, OT or AST; while in engineering design, data consistency embodies two aspects: time consistency and space consistency, which can not be achieved by the only-time-related data maintenance algorithms. Time consistency means that

1-4244-0963-2/07/$25.00 ©2007 IEEE.

the final status of data copy of every site should be kept consistency. Space consistency means that data objects from different or same trades should be correlated and constrained by each other so that conflict between different models can be detected including location conflict, topology relation conflict, project attributes conflict (For example, different materials or outlook dimension are set in the same model), project rules conflict (For example, the distance between models cannot satisfy the requirements of the design) etc.

On the basis of the above analysis, we can find that in order to realize the real-time collaboration and *view-wandering* between different applications, a mediated model must be provided to negotiate between the hybrid data model of different applications so that the execution of operations sent by all designers can be reflected on that model and designers can get their interested view by data projection through data proxy. Once the Total Data Model (TDM) is designed, time collaboration can be achieved by modifying only the data structure of the traditional consistency maintenance algorithm while the space consistency can be achieved by executing real-time space conflict detection on that model and detection result can be projected to user view by highlighting technology. Centered on the TDM, a layered transformation model is provides to support heterogeneous collaboration between different applications.

The rest of this paper is organized as follows. Section 2 reviews related work to support heterogeneous collaboration between hybrid applications. Section 3 introduces the framework of the TDM centered model. Section 4 discusses two critical issues related with the framework. Section 5 summarizes the paper and gives an outlook for the future work.

2. Related Work

There have been numerous research efforts in interoperable heterogeneous groupware systems and applications. The Disciple framework by Marsic [3], which is built on Java and XML, provides some features to handle heterogeneity. However, its purpose is to match the disparities in computing and communication capabilities or the participants, not to transparently share and interoperate heterogeneous single-user applications. The DistEdit [4] provides a library of functions so that existing single-user editors can be converted to group editors. The disadvantage of this method is that the source code of the original single-user editors must be obtained and extended with the provided collaboration-aware functions, which is not always available in practice. Dewan and Sharma [5] proposed building a bridge between two systems, which are to be interoperated. Yet, not all systems have radically compatible collaboration protocols. The PSI platform [6] and the Placement Documents [7] system provides a middleware approach to monitor the applications' accesses to a shared data repository and trigger user-supplied programs when interesting operations are performed. However, in this method, the operations monitored are coarse-grained and insufficient to implement awareness mechanisms for the tightly-coupled, synchronous collaborations traditional application sharing is usually supposed to support. Du Li and Rui Li [8] proposes a new approach called intelligent collaboration transparency, which is based on application black box assumption to transform heterogeneous single-user applications into collaborative groupware by series of event capture, event reduction, consistency maintenance, event reproduction and event replay. This method is novel enough and can solve the conversion of some relatively easy application with simple function set. However, to realize the powerful functions of commercial off-the-shelf applications by event capturing is almost impossible. Since many mature commercial applications have provided plenty of APIs, Chengzheng Sun [9] first introduces the idea of transparent leveraging single-user application for multi-user application. However, in his framework, only homogeneous applications are supported, no method for solving the incompatibility of data model of different applications is introduced. What's more, all the above research work aims mainly at field of group editors. There has been no research work in heterogeneous collaboration in the field of engineering design, to the best of our knowledge. So based on transparent approach introduced by Chengzheng Sun, we propose to build an adapter layer to transform current different CAD applications into heterogeneous groupware system which uses total data model as the core mechanism to realize the compatibility of different applications and support both time collaboration and space collaboration.

3. TDM Centered Layered Architecture Model

3.1 Analysis of Typical Work Scenario

Generally speaking, during engineering design, designers are often be divided into small groups according to their design phase, work space, role and task field etc. Due to the different role of different designers, constraints are often changeful. For example, tabling designer can only view the design result of building designer but not modify it, and vice versa. Fig 1 shows a representative scenario of design collaboration.

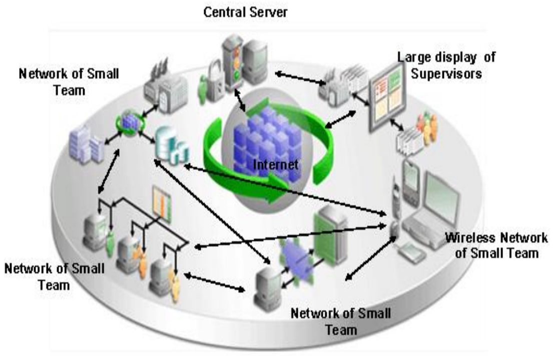

Figure1: Work scenario of the designers all over the world

Designers throughout the world construct a peer-to-peer network in their work team. Group networks are then connected to realize the effect of *view-wandering* (which means that designers can view the total model or any subset of the total model, which is made up of all data coming from all specialties. For example, tabling designers can switch from a certain pipeline in 3D view to its corresponding arts and flow graph in 2D view.). Also, one or several design servers are often be equipped to provide the functions of access management, task division, project evaluation, version control or other whole-project correlated work. Supervisors can observe the development of the whole project through the large display with high clarity.

3.2 TDM Centered Layered Architecture Model

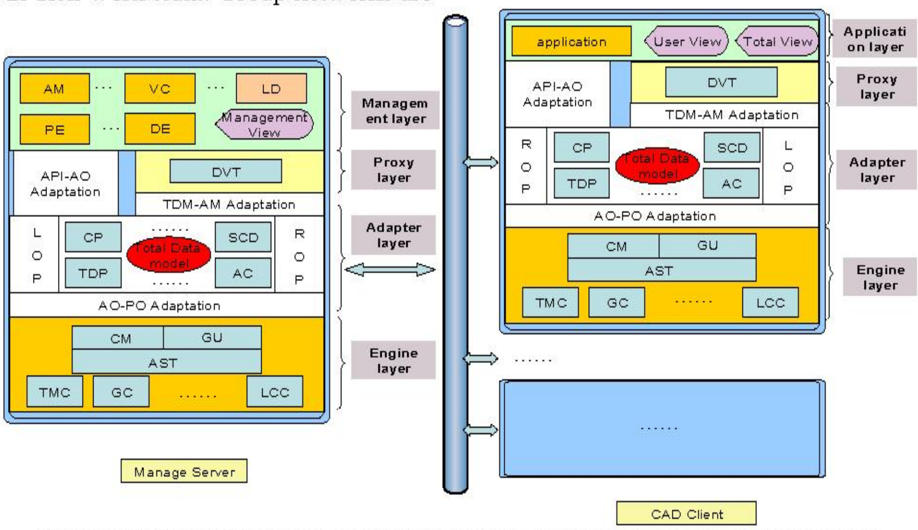

Figure 2 TDM centered layered architecture model to Support Heterogeneous Collaboration

The abbreviation of the Fig2 is described as following: ***AM***: Access Management; ***PE***: Project Evaluation; ***VC***: Version Control; ***DE***: Designer Evaluation; ***LD***: Large Display; ***DVT***: Data-View Transformation; ***TDM-AM Adaptation***: Total Data Model-Application Model Adaptation ; ***AO***: Adapted Operation; ***API***: Application Programming Interface; ***TDM***: Total Data Model; ***LOP***: Local Operation Processing; ***ROP***: Remote Operation Processing; ***CP***: Collaborative Policies ***TDP***: Total Data Processing; ***SCD***: Space Conflict Detection; ***AC***: Access Control; ***PO***: Primitive Operation; ***CM***: Consistency Maintenance; ***GU***: Group Undo; ***AST***: Address Space Transformation; ***TMC***: Time Stamp Control; ***GC***: Garbage Collection; ***LCC***: Late-Comer Control.

The system allows multiple designers to edit the same TDM at the same time over the Internet, as shown in Fig2. A collaborative editing session consists of multiple CAD Client instances. Each site maintains and manipulates a local copy of the shared TDM document. Shared TDM documents and design rules database can be located in any collaborating designer's local database with corresponding design right. This framework adopts the combination of C/S architecture and Peer-to-Peer architecture. The organization between Server and Clients is realized by using the architecture of C/S mode while that between different clients is realized by architecture of peer-to-peer. Both the Management Server and the CAD client maintain the same copy of the TDM while the data proxy module will provide different view. The implementation of client and server is realized by layer architecture. Four layers of Client are named as Application Layer, Proxy Layer, Adapter Layer and Engine Layer, while four layers of Server are named as Management Layer, Proxy Layer, Adapter Layer and Engine Layer. More detailed description will be given in the following subsections.

3.2.1. CAD Client

CAD Client achieves the transparent and convenient transition of single-user CAD applications to heterogeneous multi-user groupware by constructing Proxy Layer, Adapter Layer and Engine Layer, based on APIs of single-user application. TDM combines all the information of all correlated fields (whose detailed description will be given in section 4.1), which is loaded from the module of Version Control of the Management Server when a CAD Client first joins. Proxy Layer is introduced to separate data model from user view, providing the convenience of multi-view wandering and visual clue of space conflict by highlighting technique.

The operations from user interface will be captured by the LOP module of Adapter Layer. These Operations will be first tested whether it is allowed by user right of current designer, and then the allowed operation will be explained and decomposed into atomic operations. The atomic operations will be transmitted to the Engine Layer to be attached by timestamps. Then the Ops attached with timestamp are returned to the Adapter Layer and be broadcast to other Clients and Server by ROP module. Meanwhile, the ROP module of Adapter layer receives remote operations, passes them to Engine Layer to accomplish time consistency maintenance. When the operation is executed, the TDM is also changed, which will trigger the SCD module to process space conflict detection and the check result will be highlighted on the users' view through the conversion of data proxy.

Engine Layer is responsible for judging the causal relationship of operations based on the attached timestamps and completing operation transformation, to guarantee the consistency of model copy. Traditional OT algorithm [10] (such as dOPT, adopted, GOT, LBT etc.), Mu3D, dARB, treeOPT as so on have some deficiency in solving the problem of copy consistency maintenance. AST algorithm, as the previous research fruit of our lab, has some advantages in efficiency and maturity, so we choose it as our work basis. And based on it, we introduce efficient garbage collection mechanism to support the work of large amounts of designers with tens of thousands objects, which will be described carefully in section 4.2.

3.2.2. Management Server

The project supervisors complete access management, task management, rule management, project evaluation, efficiency assessment as well as version management etc. through the Management Layer of the Server port. Access management is responsible for role creation, right deployment and user identification validation. Task deployment divides design region, design task and design flow, with a tree described workflow as the storage structure. Information about project's design quantity and design cost is maintained by the project evaluation module. Supervisors can get the information about workers' efficiency and quantity according to the real-time observation of the whole work group and the session logs analysis tools. Adjust of the role, tasks and efficiency assessment of the engineers will be propagated to corresponding Clients.

The function of other layers is similar to that of Client Port, and we'll not describe them again.

4 Two Critical Issues to Discuss

4.1 TDM Based on B-rep

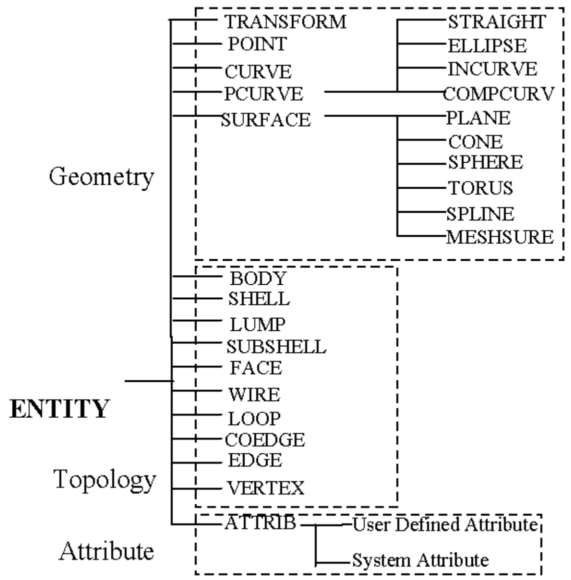

Figure 3 Relationship of layered class of TDM

We use boundary representation, or B-rep modeling method to construct the 3D model of TDM, which separately represents the geometry (detailed shape) and the topology (connectivity) of objects. This provides the ability to determine whether a position is inside, outside, or on the boundary of a volume. B-rep modeling allows TDM to integrate wireframe surfaces and solids. Geometry, topology and attribute constitute the three factors of DTM. They are all derived from the abstract ENTITY class, which defines the common data and methods such as save, insert, delete, and retrieve and so on. The architecture of the DTM class is showed in Fig 3.The following gives more detailed description about the three factors: geometry, topology as well as attribute.

4.1.1 Geometry

Geometry refers to the physical items represented by the model (such as points, curves, and surfaces), independent of their spatial--or topological--relationships. TDM supports free-form geometry routines based on non-uniform rational B-splines (NURBS). Not only manifold geometry but also non-manifold geometry can be expressed. Geometry can be bounded, unbounded, or semi-bounded, allowing for complete and incomplete bodies.

Each type of geometry has its own unique set of data. For example: *Point*: A point holds a count of the number of vertices that refer to the point, and records the coordinates of the point; *Curve*: A curve record holds a count of the number of edges that refer to the curve, and provides a route to the details of individual curve types (such as straight lines, ellipses, etc.).

4.1.2 Topology

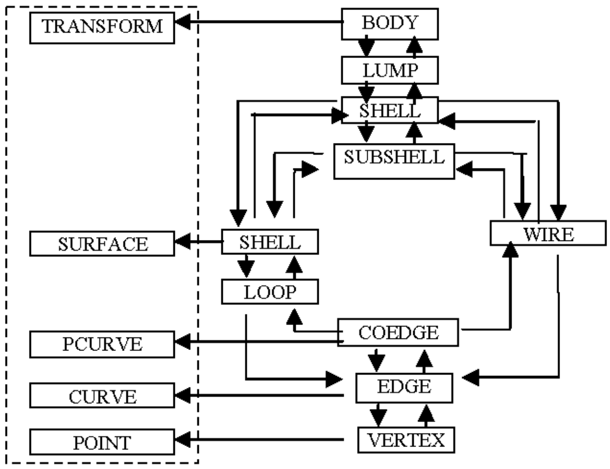

Figure 4 Relationship of classes of topology and geometry

Topology refers to the spatial relationships between the various entities in a model. It describes how geometric entities are connected. The topology of TDM includes BODY, LUMP, SHELL, SUBSHELL, EDGE and VERTEX. TDM adopts unified structure to store wireframe, surface and solid. Fig4 describes the relationship between topology objects and between topology and geometry objects.

4.1.3 Attributes

Attributes are used to attach data to entities. Every entity may have zero or more attributes. The class ATTRIB, which is derived directly from the ENTITY class, provides the data and functionality that all attributes share, for both user-defined attributes and system attributes. Attributes carries not only simple data, pointers to other entities, but also links to project-specific, variable length rule data. Many attribute classes that perform specific tasks are derived from the parent ATTRIB class.

4.2 Garbage Collection based on AST

AST Algorithm doesn't adjust the operation itself, which is adopted by tradition OT algorithm, but retraces the document's address space to the state at the time of the operation's generation. Then the operation can be executed directly in this address space. It can not only achieve the same goal of consistency maintenance but also provide a better support for Undo and the execution efficiency as it can reach O(logn). More detailed description can be got from [11] However, in AST algorithm, it doesn't provide efficient garbage collection strategy to process the invalid node and redundant link node of every node, which becomes a great obstacle in real engineering design process with huge amounts of objects to be handled.

During the design process, designers insert or delete objects frequently, which lead to large amounts of *forever invalid nodes*. These nodes make no effect to future work once its delete effect has been reflected on all sites. We consider that in fact the designer, who has deleted an object, will generally undo that delete operation in limited steps and that after having committed N_{undo} operations after that deletion, he maybe don't want to undo it any more because there already have been many midst operations between his current operation and that deletion operation or even if he wants to undo that deletion, he can remedy that by re-insert it.

The resolution is to introduce a garbage collection process to check the validity of the nodes of the document. Maintain a State Vector Table (SVT) at every site and suppose the state vector table of the k_{th} site is SVT_k. Initially, $SVT_k[i][j] = 0$ for all i, $j \in 1,\dots,N$(N denotes the number of Clients who joined in the design work). After executing an operation O from a remote site r, time stamped by SV_r, the maintenance of SVT is described by procedure1 and the new garbage collection method is described by Procedure 2.

Procedure 1 Maintain_SVT (SVT_k, SV_r), maintain the SVT after the execution of Op timestamped by SV_r

1：for i=1 to N do //updates all the elements of the vector of remote site r.
2： $SVT_k[r][i]=SV_r[i]$
3：End for

Procedure 2 Garbage_Collect (Docs, SVT_k), collect forever invalid node on Doc_s

1：for all $i \in 1,\dots,N$ *//get the minimum of the vectors to judge whether all the sites have executed the delete Op*
2： $MSV_k[i]=\min(SVT_k[1][i], SVT_k[2][i],\dots, SVT_k[N][i])$
3：End for
4：for all character node CN_i of linear structure of Doc_s do
5： Consider the last *O* of CN_i
6：   If *O*.type='Insert' or "update" then
7：   Goto Line 16
8：   End if
  //The last Op of CNi is delete operation
9：   if *O*.SV> $MSV_k[i]$ //not all the sites have executed the O

```
10：        Goto Line 16
11：    step=SVTk[r][r]- O.SV[r]  //r indicates the site id of the birth of O
12：    if step >Nundo   // remote r has created at least N operations after that deletion
13：        Delete Node i and its corresponding operation table
14：        Continue
15：    End if
16：    for all O of CNi which O.type="update"// to compress update Op
17：        step=SVTk[r][r]- O.SV[r]   // r denotes the site id of the O generated
18：        if step > Nundo
19：          Delete O from the operation table of CNi
20：        End if
21：    End for
22：end for
```

5. Conclusions

In this paper, we present a TDM centered framework to support heterogeneous collaboration of hybrid CAD applications and analyze the characteristics of consistency in CAD domain. In order to support both time and space consistency maintenance, we design B-rep based TDM to combine not only the information of entities but also relationships between the entities with a data proxy to separate data model from user view so that different users will have different views according to their roles. In order to adapt to the new applying environment with large-scale project to design, we introduce garbage collection to compress *forever-invalid* node. We have made experiments on Microsoft Windows platform with two applications AutoCAD and 3DMax. During the experiment, the threshold of supported undo step N_{undo} is set according to different scale of designers in a work scenario. With 100 designers, the statistics showed that 15 is the best choice. More statistics data will be given in future research paper. The prototype system has completed generic Engine component based on AST and solved the problem of time consistency. Next, we will do more research work into the description of the total data model as well as the automatic explanation of the captured operation. Based on the total data model, the model-view projection is also a research direction and our partner of engineering field will do more work as for that direction.

6. Acknowledgements

The work is supported by National Natural Science Foundation of China (NSFC) under Grant No.90612008, National Grand Fundamental Research 973 Program of China under Grant No. 2005CB321905, Shanghai Science and Technology Committee Key Fundamental Research Project under Grant No. 05JC14006 and AMD University Cooperation Project.

7 References

[1] Liu Hong, Liu Xi-yu, Ma Li-na. Supporting Evolution in A Computer-Aided Design Environment[J]. *Journal of Computer-Aided Design & Computer Graphics*, 2003,15(2):167-173.

[2] Jonathan Grudin. Why CSCW applications fail: Problems. in the design and evaluation of organizational interfaces. In *Proceedings of ACM Conference on Computer Supported Cooperative Work (CSCW'88)*, pages 85–93, Portland, Oregon, September 1988.

[3] Ivan Marsic. An architecture for heterogeneous groupware applications. In *Proceedings of the 23rd IEEE/ACM International Conference on Software Engineering (ICSE 2001)*, pages 475–484, Toronto, Canada, May 2001.

[4] Michael J. Knister and Atul Prakash. DistEdit: A distributed toolkit for supporting multiple group editors. In *Proceedings of ACM CSCW'90 Conference on Computer Supported Cooperat e Work*, pages 343–355, Los Angeles, California, October 1990.

[5] Prasun Dewan and Anshu Sharma. An experiment in interoperating heterogeneous collaborative systems. In *Proceedings of European CSCW Conference*, pages 371–391, 1999.

[6] Kevin Palfreyman, Tom Rodden, and Jonathan Trevor.PSI: A platform for shared interaction. In *ECSCW99: Proceedings of the Sixth European Conference of Computer Supported Cooperative Work*, pages 351–370, Copenhagen, Denmark, September 1999.

[7] Anthony LaMarca, W. Keith Edwards, Paul Dourish,John Lamping, Ian Smith, and Jim Thornton. Taking the work out of workflow: Mechanisms for document-centered collaboration. In *Proceedings of the Sixth European Conference on Computer-Supported Cooperative Work (ECSCW'99)*, pages 1–20, Copenhagen, Denmark, September 1999.

[8] D. Li and R. Li. Transparent sharing and interoperation of heterogeneous single-user applications. *In Proc. of the ACM Conf. on Computer-Supported Cooperative Work*, pages 246-255, Nov. 2002.

[9]Steven Xia, David Sun, Chengzheng Sun and David Chen. Leveraging Single-User Applications for Multi-user Collaboration: the CoWord Approach. *In Proc. of the ACM Conf. on Computer-Supported Cooperative Work*, pages 162-171, Nov. 2004.

[10] C. A. Ellis and S. J. Gibbs. Concurrency control in groupware systems. *In Proceedings of the ACM Conference on the Management of Data 1989*, pages 399–407, Portland Oregon, May 1989. ACM.

[11] Ning Gu, Jiangming Yang, Qiwei Zhang. Consistency maintenance based on the mark & retrace technique in groupware systems. *In ACM GROUP 2005 Conference, pages*264-273, *Sanibel Island, USA, Sept 2005.*