A Feature-oriented Database Framework for Web-based CAx Applications

Tang S. –H.[1], Ma Y. –S.[2] and Chen G.[3]

[1]Nanyang Technological University, pg02104852@ntu.edu.sg
[2]Nanyang Technological University, mysma@ntu.edu.sg
[3]Nanyang Technological University, pg02198079@ntu.edu.sg

ABSTRACT

In this paper, a multi-application and feature-oriented database framework for web-based CAx applications is proposed based on client-server architecture. The web server provides a multi-view data access interface (MDAI) through which distributed users can access product and process information via data network. Application server can provide feature-modeling facilities on the basis of a geometric modeling kernel as well as DB manager. To enable information sharing among different applications for collaborative engineering, a four-layer information model is proposed. The entire product model (EPM) covers information of product entire life cycle. Sub-models, coming from CAx applications, can be accommodated as specific views of the EPM from an application-specific viewpoint. Mapping mechanisms are investigated to convert the EXPRESS-defined feature object information model to the database schemas. The generic feature representation and geometrical data representation in database is given on the basis of the proposed mapping mechanisms. The information workflow and mechanisms to control concurrency and to validate features are also discussed.

Keywords: Feature modeling; Database; STEP; CADCAM; Collaborative engineering

1. INTRODUCTION

1.1 Literature Review

Competition in the global market forces companies to develop product with the highest quality, at the lowest price, and more importantly, in the shortest time-to-market. In order to speed up the product development, the concept of collaborative engineering occurs. In a collaborative engineering environment, it is common for engineering tasks to be carried out by a group of engineers who may be distributed in terms of both time and space. Furthermore, different engineering partners need to use different applications, which means a product model generated from an application system has to be shared directly by other ones. Therefore, information sharing among CAx applications becomes the bottleneck for collaborative engineering.

ESPRIT's project 322 "CAD Interface", was among the pioneers to realize data exchange among different CAD systems [19]. The implementation of CAD*I ORACLE database, which covers geometric data and some administrative data, enabled data exchange among AIS modeler, STREAM100 CAD system and CAD*I neutral files. The results of the project showed that with a relational database management system, different kinds of data, applications and operations can be well managed. The security of the data can be ensured. In addition, the communication and productivity within the company has been improved. However, such preliminary results are not upto the expectations from CAD suppliers and CAD users. A lot of work has yet to be done to complete and improve the capabilities and performances of a product information DBMS. For example, the information model for CAD*I database covers only part of the geometric and some administrative data. It is not enough to represent different aspects of the entire product life cycle.

In order to describe a product's entire life cycle, International Standard Organization (ISO) has been working on its STandard for the Exchange of Product model data (STEP) effort since 1984. Currently STEP contains information, from design to analysis, manufacture, quality control testing, inspection, and product support functions [12].

Under the STEP framework, a web-based collaborative design system is proposed and implemented as Cyberview by Kang and Kim [14][15]. They presented an overall architecture, which adopted an open data standards (STEP and VRML) to allow users from a wide variety of platforms to access and visualize product information. Two algorithms, for transforming STEP geometry schema into an object-oriented database

schema and for mapping geometry schema to VRML, are presented. However, only geometric data can be shared among different applications in their research. High-level information such as features cannot be shared; it means semantic information was lost.

To retrieve feature information from exchanged data file, Bhandarkar developed a feature extraction system, which takes a STEP file as input and produces a form-feature STEP file [1]. This STEP file can be exchanged between various companies over the Internet and can serve as input for further downstream tasks. Similarly, Fu et al [10] proposed an approach to identify design and machining features from an exchanged part model. A multi-level feature taxonomy and hierarchy is described. Although feature extraction and identification can partially recognize some feature information from the exchanged part model, information loss still exists because these approaches depend on pure geometric data. For example, feature relationships can not be recovered from the geometric data model.

Currently, most of the CAx systems are feature-based. Therefore, feature information must be represented such that engineering meaning is fully shared among CAX applications.

To support collaborative feature modeling, a web-based collaborative system [2], webSPIFF, has been developed on the basis of the semantic feature modeling system [3]. In the system, a multi-view enhanced feature model [4][5] is adopted which can maintain feature semantics among CAx applications. Mechanisms for feature model validity and feature conversion are described. Martino [17] proposed to use a multiple view intermediate modeler to integrate design and engineering process in a distributed object-oriented system environment. The intermediate model is both multiview supported and feature-based by incorporating design-by-feature and feature recognition approach. Therefore, it can achieve high-level, semantic data communication among engineering processes. However, these researches didn't mention how to manage the product data, whether in file format or as database objects. Database allows the handling of a large volume of data and is generic for reading, writing, updating and deleting operations. The DBMS can ensure the security and transparency for the users of CAD data. Therefore databases are appropriate tools for information sharing among CAx applications.

Hoffman proposed a product master model to integrate CAD systems with downstream applications for different feature views in the product life cycle [11]. A change protocol proposed can maintain the link between application proprietary data and the shared product master model. However, the proposed product master model contains only shared geometrical data, namely, the net shape of the product model. To maintain the association between product master model and the distributed application proprietary feature semantics will be very difficult practically.

Kim [16] describes an interface (OpenDIS) between the geometric modeling kernel and the DBMS for the implementation of CAD system that uses the STEP database as the native storage. A prototype CAD system has been implemented using the OpenCascade geometric modeling kernel and ObjectStore. The STEP methodology is used for the database schema. However, currently, STEP cannot fully cover information for different CAx applications, particularly for feature-based design.

Except research effort, there also exist commercial effort which can support collaborative engineering to some extend. For example, CAD web portals such as CAD/CAM-E by CAD/CAM-E Inc. [6], OpenDXM by ProSTEP [18]. All these web portals are in fact operated by translator providers. They take certain kinds of file format as input and generate another file format. During data translation, useful information such as features may be lost since different CAx applications define features in different ways. Therefore, it is not a kind of meaningful information sharing. There is one commercial system, OneSpace by Cocreate [9] currently offering some collaborative modeling capabilities. However, its modeling facilities are severely constrained by the modeler at the server, SolidDesigner, and by the model format into which it converts all shared models [2].

1.2 Existing Problems

Although a lot of research and development work has been done on data exchange/sharing to enable collaborative engineering, problems still exist.

- *Information loss.* Although many proposed systems claim to be CAD-neutral based on STEP and CORBA, they lack a certain level of interoperability so far. In the process of data exchange, useful information such as features is often lost. Therefore it is not complete information sharing.
- *Duplicated data and conflicts.* CAD data is often stored in a file format, which means duplicated data and potential conflicts. In addition, files are not flexible enough to support the multiview functions required by different applications. For example, multiple end-users or applications cannot synchronize definitions and modifications easily for the data stored in a file.

Therefore, the research work presented in this paper can be justified, that is to create a web-based feature-oriented database framework to enable information sharing among CAx applications.

2. SYSTEM ARCHITECTURE

2.1 Overall System Architecture

To enable information sharing among CAx applications in a web-based environment, database-driven, feature-oriented system architecture is proposed as show in Fig. 1.. The proposed system adopts client-server architecture which includes client, application servers and database server. The application servers are separated into web server, application object server and feature object server.

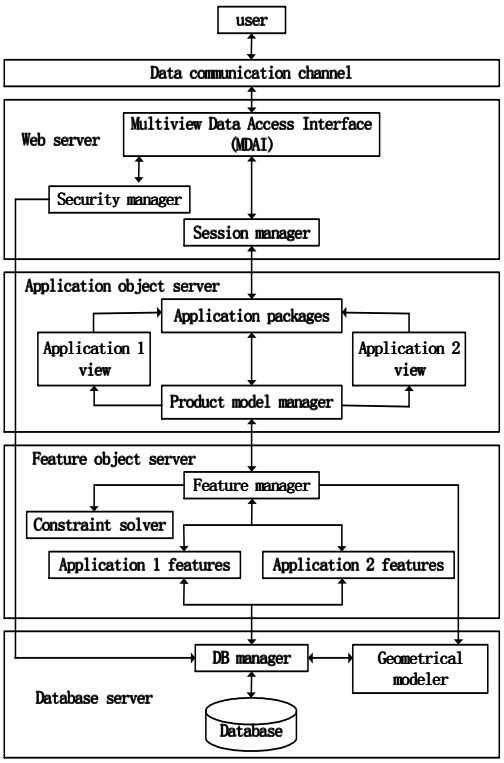

Fig. 1. Overall system architecture

2.1.1 Web Server

The web server contains a multiview data access interface (MDAI), security manager and session manager. MDAI provides shared access for multiple users. It can instantiate different views for different users according to users' requirement. Security manager is used to check whether the user has the right to access the product model data and what kind of access right he may have. Users are separated into several groups. Each group has different access right. All this management data are stored in the database. Session manager is responsible for controlling concurrent access by multiple users of the same data.

2.1.2 Application object server

The application object server provides different application packages for different users such that users can interactively carry out feature-based operations on the basis of feature object server. The product model manager is also responsible for organizing information for multiple applications according to the user's requirements. This information includes feature model and solid model (B-rep).

2.1.3 Feature object server

The feature object server can provide feature objects for application packages. To maintain the meaning of a feature during each feature operation, such as adding feature, deleting feature and modifying feature, feature manager will call the constraint solver and the geometrical modeler to validate the feature. The constraint solver can check the validation of all constraints, which are part of the feature definition. The geometrical modeler can validate feature geometry.

2.1.4 Database server

The database server provides physical storage for all kinds of data including product model data, security management data and so on. Within the database, geometrical data and features for different applications are stored as data elements across tables. In this manner, Database manager can reorganize these data elements for different applications with great flexibility.

2.1.5 Geometrical modeler

The geometrical modeler proposed here provides general functions such as geometry construction functions, modification functions, computation functions and so on for creating, modifying and interrogating solid models. Therefore, higher-level feature modeling functions (e.g. *create_feature*, *modify_feature* and so on) can directly call these modeler-provided functions. For example, a "*create_feature*" function calls geometry construction functions (e.g. *surface.add()*) provided by geometrical modeler to construct feature geometry. A feature validate function (e.g. *feature.is_feature()*) call modeler-provided functions (e.g. *face.is_face()*) to check if the feature geometry is valid.

2.2 Information Flow

In the proposed system, when a user accesses the MDAI via data network (LAN or Internet), the security manager, by connecting to the database manager, will check the user information stored in database, and decide whether the user has an access right, what kind of access right he has, and what kind of information (application-specific view) he wants. Then the application model is established for the user and a new session is registered in the session manager. After this, the user can work on an existing product model via the

database manager, or create a new model to carry out feature modeling operations. After each feature operation, such as insertion, modification and deletion, the geometrical modeler will be called to validate the feature geometry. Then the feature manager will call constraints solver to check all the feature constraints (intra-application constraints and inter-application constraints) to validate the feature model. Finally, the finished product model can be stored into the database by the database manager.

3. INFORMATION MODEL

3.1 Four-layer Information Model

In this research, information model is built on the basis of an extended STEP framework since STEP is the international standard and widely accepted by both vendors and users. However, using only STEP-based product specification cannot ensure integration, because inter-relationships and constraints between applications are not defined in STEP. To achieve integration among CAx applications, the sharing of a common product model representation is crucial. The shared product model provides different views for various applications. These views are context-dependent interpretations of self-contained subsets of information of the entire product model (EPM). Based on Zha's work [25], we propose to use four-layer information model as showed in Fig. 2.. The four layers are application, information, representation and physical layers. The information layer contains four components, i.e. EXPRESS specification, application features, unified features and EPM.

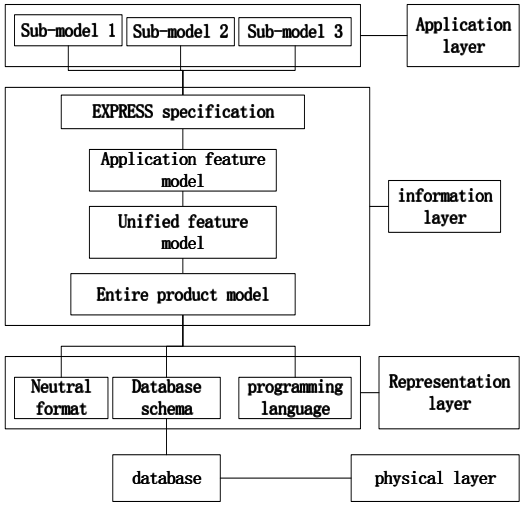

Fig. 2. Four-layer information model

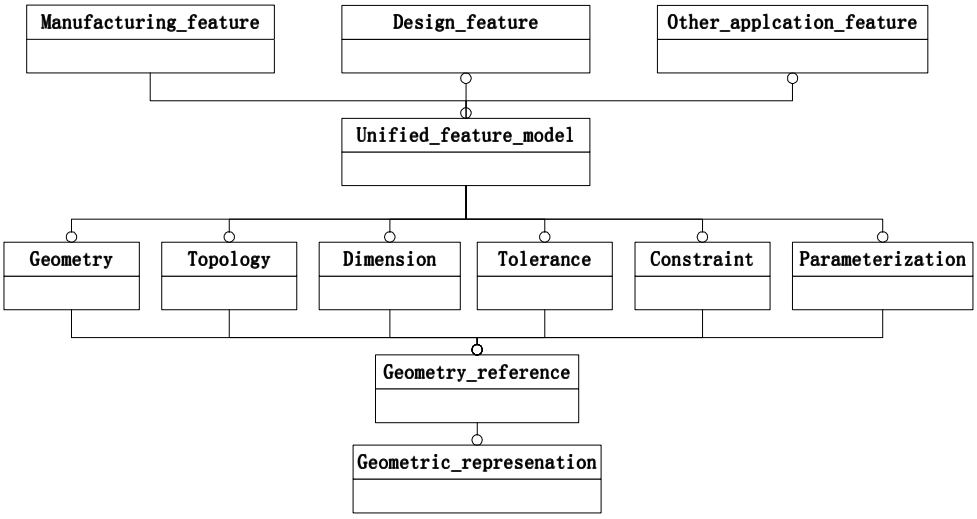

Fig. 3. Unified feature model [7]

EPM describes information across applications, and contains the domain classification ontology and metadata; the detailed high level feature objects are organized by different sub-models in the application feature layer. Application feature sub-models can provide specific view of the EPM. Next, the unified feature model specifies a generic feature modeling framework and some common definitions for different applications as described in Fig. 3.. Each application defines its feature model on the basis of unified feature model. All EPM, application feature model and unified feature model are described in EXPRESS language. For

implementation, this EXPRESS-defined information model must be mapped to database schema (for data storage), programming language (workform format) and neutral format (for data communication).

3.2 Unified Feature Model

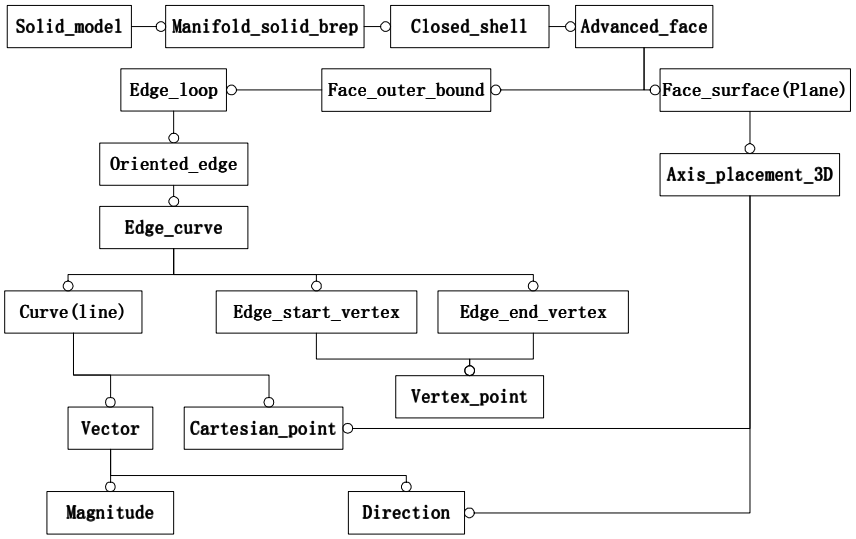

Fig. 4. Partial geometrical representation schema [13]

Although different applications define features in different way, their features are common in having geometry, topology, dimensions, tolerances, constraints and parameters as their attributes. In addition, different application features refer to the same final product geometry. Therefore, Chen proposed a unified feature model in 2003 as described in EXPRESS-G in Fig. 3. [7]. The unified feature model uses a five-layer architecture. The top layer is the application model layer, which consists of different application feature models. The second layer is unified feature layer. A unified feature has the geometry, the topology, dimensions, tolerances, constraints and parameters as its attributes, which form the third layer. The bottom layer is the geometrical representation layer. An intermediate geometry_reference layer is used for connecting higher-level feature specifications with lower-level topological entities to solve persistent naming problem. Persistent naming problem comes from the inconsistency between different representations of solid (e.g. CSG and B_rep) during model modification and has been well documented in the literature [8][20]. A lot of effort has been made to solve the problem, such as B_rep to CSG conversion [22, 23, 24], consistency verification [21] and so on. It is one of the most important research topics, but will not be addressed here. Such a unified feature model provides a generic feature definition for different application features.

3.3 Geometrical Representation

Unified feature model still allows different applications to define features in different ways, but their definition candidate types, such as geometrical representation, and the common processing methods are the same. In this research, STEP part 42 will be adopted, and extended in the aforementioned aspects, as the geometrical representation schema for all feature models. Fig. 4. is a partial geometrical representation schema of manifold_solid_brep [13].

4. MAPPING MECHANISMS

Under the four-layer information model structure, the entities at different levels can be mapped into schema definitions for a potential comprehensive product database such that the unified flexible feature object structure can be represented. The following are the rules for mapping.

- Each entity shall be mapped to an object type. An object table shall be created for each concrete entity.

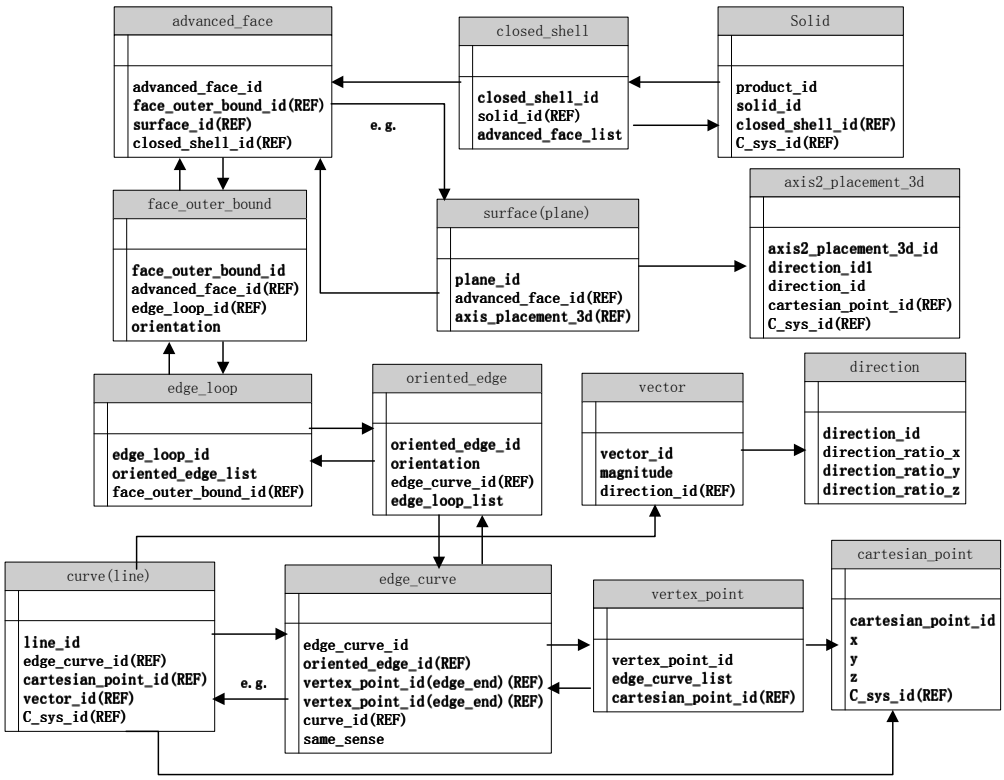

Fig. 5. Partial database schema for geometrical representation

- An entity attribute shall be mapped to a column with *REF* data type pointing to the referenced object of another object table.
- By defining object types, inheritance is directly supported in the database.
- Enumeration data type is simulated by defining type array or nested tables. All the enumerators can be stored in an array or nested table.
- The select data type can be simulated as an attribute of the object type whose real data type is decided by the method of the object type. A method of object type is simple SQL or PL/SQL, which is used as a discriminant to decide the select data type.
- Object-relational database supports aggregate data types by the definition of type array and nested tables. Array types are suitable for fixed-size aggregate data type. Nested tables are suitable for aggregations whose sizes are not fixed.

A partial geometrical database schema is created according to STEP 42 using the proposed mapping mechanism, see Fig. 5.. All attributes with suffix id (but without *REF*) represent object identifier (OID), which is the globally unique and immutable object identifier generated by DBMS. An OID allows the corresponding row object to be referred to from other objects. A built-in data type called a *REF* represents such references. A *REF* encapsulates a reference to a row object of a specified object type. An arrow here represents such *REF* relationship between object types. For example, in the *oriented_edge* table, attribute *edge_curve_id* has the *REF* data type, which is used as a reference pointing to the *edge_curve* object in the *edge_curve* table. Aggregate data type (one to many relationship) is expressed as an attribute with suffix *list*. Here, we use a generic schema to collect each member of a list from the target object table, see Fig. 6.. An attribute with suffix *list* (aggregate data type) in Fig. 5. and Fig. 7. shall be defined as *REF* data type with name *list_id* which refers to list object in the *entity_list* object table by *list_id*. A nested table called *id_list* stores all the list members' ids in the nested table. Within the nested table, *entity_type* is used as a vector to decide from which object table we can get the list members. *Entity_id* uniquely identify entities from entity table. An implicit system generated *nested_table_id*, correlates the parent row object with the row objects in the nested table. Detailed explanations on geometry and topology of a *manifold_solid_brep*, please refer to [13].

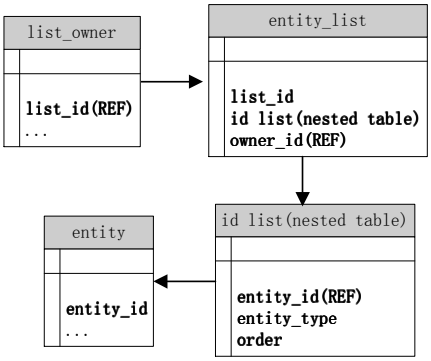

Fig. 6. Generic schema for aggregate data type

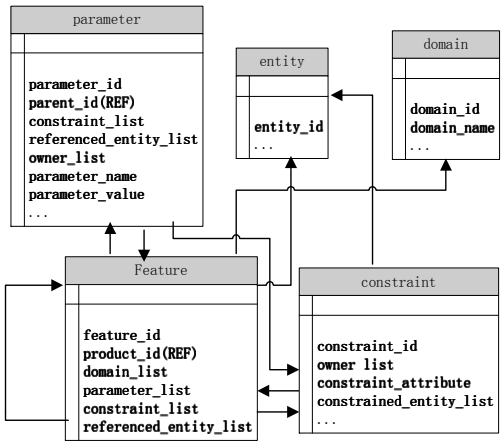

Fig. 7. Generic feature representation in database

A generic feature representation in database can be expressed as Fig. 7. under the framework of unified feature model described in Fig. 3.. A feature has *feature_id*, *product_id* and *domain* as its attribute. The *feature_id* attribute is an OID, which can uniquely identify a feature object in database. *Product_id* specifies which product a particular feature belongs to. *Domain* has select data type, which can be design, manufacturing, CAE and so on that are stored in domain table. A feature shall also contain a list of referenced entities, a list of constraints and a list of parameters. Dimensions and tolerances in Fig. 3. are regarded as a kind of constraint bounded to certain geometrical entities. Using the generic schema described in Fig. 6., each members of the parameter list can be uniquely identified by a *parameter_id* from parameter table. A parameter has a list of constraints which is stored in constraint table identified by *constraint_id*. Constraint here is used to bound parameter to other constrained entity. *Referenced_entity* of feature includes entities (e.g. faces, edges and vertices) or other features. *Entity_id* can uniquely identify the referenced entities stored in entity table. A constraint of a feature can be uniquely identified by *constraint_id*. *Constrained_entity_list* identifies constrained entity from the entity table by *entity_id* and *entity_type*.

5. FEATURE MODEL VALIDATION CHECK

Feature validity must be checked during feature modeling operations in order to maintain the feature semantics. A feature is valid as long as the feature satisfies all the relevant constraints (whether intra-application or inter-application ones) and the feature geometry is valid. After each feature modeling operation, the geometrical modeler will be called to validate feature geometry. Then feature manager will call constraints solver to check all the relevant constraints to determine if all features are valid. There are two kinds of constraints solvers. Local constraints solver processes intra-application constraints within application-specific view. If conflict of intra-application constraints occurs, local constraints solver can determine automatically which constraint should be satisfy first according to the value of *constraint_strengh*, which is an attribute of constraint. It is an enumeration data type, which may include several levels, such as *required*, *strong*, *medium* or *weak*. It represents the extent that the constraint needs to be imposed when constraints conflict with each other. Global constraints solver can solve inter-application constraints according to the value of *domain_strength*. Its value can be predefined which regulates priority sequence of different domains, or is set by an authorized user. Any conflict of inter-application constraints will be detected by global constraints solver after which the constraints solver can trigger corresponding applications to reevaluate the product model according to domain_strength. Only when all constraints are checked and feature geometry is validated, does feature validation finish.

6. COLLABORATIVE MODEL OPERATION

As the proposed the system is supposed to support shared accesses for multiple users, data integrity of EPM must be maintained while enabling maximum concurrent access to the data.

Concurrent access of data at the lower level is controlled by the database locking mechanism and managed at higher-level by a session manager. Database locks are used to control concurrent accesses to a data object and prevent destructive interaction between users accessing data. The granularity of locking entities will be one of the many possible research topics to enhance the performance of the database; no further discussion is given here. A session manager control the database locks according to predefined *domain_strength*. If multiple

users from different domain are requiring for the same data, the user with highest domain_strength has the priority to use the data while other users have to wait for him (can only view the data) until his session is finished. There are two kinds of sessions, view only session and edit session. If a user executes a view only session, he can execute as many queries as he like against any tables. Other users of the data do not need to wait for him. When a user creates an edit session to do modeling operation, all related tables are locked at the row level after the data of that view is checked out. Therefore, the session of any other users who want to use the same data will be suspended by the session manager before the ongoing session finishes, unless the user has higher *domain_strength* (if so, he can send request to session manager to apply for the control of the data). Multiple users in the same domain can also access to the same data concurrently, but only one user can edit the data under the control of session manager, others can only view the data until he releases the control. When the ongoing session is finished and data is checked in, database will trigger a program automatically to inform other views to reload the model for further reanalyze.

7. CONCLUSION

In this paper, the web-based, database-driven, and feature-oriented system architecture is proposed. It enables information sharing among CAx applications. The client-server architecture can provide shared access for multiple users. The proposed four-layer information model can integrate different applications with EPM, and allow the manipulation of application-specific information with sub-models. Building the information model with reference to STEP makes the proposed system easier to be implemented and integrated with other STEP-based applications. With the database support, product and process information can be organized for multiple applications with great flexibility. Geometrical modeler is incorporated into the system to provide lower level geometrical modeling service, with which feature-based modeling can be realized. In order to implement the proposed system, mapping mechanisms, from STEP-like information model to the target database schema, is investigated such that flexible EXPRESS-defined data structure can be converted into database schema. On the basis of proposed mapping mechanisms, we further proposed a generic feature representation scheme and a geometrical data representation scheme in databases. The mechanism to control data concurrency is suggested so that multiple users can concurrently access product and process information; at the same time, data consistency can be maintained.

8. REFERENCES

[1] Bhandarkar, M. P., *STEP-based Feature Extraction From STEP Geometry for Agile Manufacturing*, Computer in Industry, Vol. 41, pp. 3-24, 2000.

[2] Bidarra, R., van den Berg, E. and Bronsvoort, W. F., *Collaborative modeling with features*, Proceedings of DETC'01 2001 ASME Design Engineering Technical Conferences September 9-12, 2001, Pittsburgh, Pennsylvania.

[3] Bidarra, R., Bronsvoort, W.F., *Semantic feature modeling*, Computer-Aided Design Vol. 32, pp. 201–225, 2000.

[4] Bronsvoort, W. F., Bidarra, R. and Dohmen, M., *Multi-view feature modeling and conversion*, 1997.

[5] Bronsvoort, W. F., Bidarra, R., and Noort, A., *Semantic and multiple-view feature modeling: towards more meaningful product modeling*, 1998.

[6] CAD/CAM-E website. http://www.cadcam-e.com/

[7] Chen G., *Unified Feature Model for the Integration of CAD and CAX*, First-year report, School of MPE, 2003.

[8] Chen, X. and Hoffmann, C. M., *On Editability of Feature-based Design*, Computer-Aided Design, Vol. 27, No.12, pp. 905-914, 1996.

[9] Emmel, J., *OneSpace Integrating Collaboration Technology and Enterprise PDM*, Technical Whitepaper of CoCreate Software GambH, 2000.

[10] Fu, M. W., Ong, S. K., Lu, W. F., Lee, I. B. H. and Nee, A. Y. C., *An approach to identify design and manufacturing features from a data exchanged part model*, Computer-Aided Design, Vol. 35, pp. 979–993, 2003.

[11] Hoffman, C. M. and Arinyo, R. J., *CAD and the product master model*, Computer-Aided Design, Vol. 30, No. 11, pp. 905–918, 1998.

[12] ISO 10303, Industrial Automation Systems and Integration — Product Data Representation and Exchange — Part 1: *Overview and Fundamental Principles*, ISO 10303-1:1994 (E), ISO, Geneva, 1994.

[13] ISO 10303, Industrial Automation Systems and Integration — Product Data Representation and Exchange — Part 42: *Integrated Generic Resources: Geometric and Topological Representation*, ISO 10303-42:1994 (E), ISO, Geneva, 1994.

[14] Kang, S. H., Kim, N., Kim, C. –Y., Kim, Y. and O'Grady, P., *Collaborative design using the World Wide Web*, Technical Report TR 97-02 of Iowa Internet Laboratory, 1997.

[15] Kim, C. –Y., Kim, N., Kim, Y., Kang, S. H. and O'Grady, P. *Distributed concurrent engineering: Internet-based interactive 3-D dynamic browsing and markup of STEP data*, Technical Report TR 98-02 of Iowa Internet Laboratory, 1998.

[16] Kim, J. and Han, S., *Encapsulation of geometric functions for ship structural CAD using a STEP database as native storage*, Computer-Aided Design, Vol. 35, pp. 1161–1170, 2003.

[17] Martino, T. D., Falcidieno, B. and Habinger, S., *Design and engineering process integration through a multiple view intermediate modeler in a distributed object-oriented system environment*, Computer-Aided Design, Vol. 30, No. 6, pp. 437-452, 1998.

[18] OPENDXM overview. http://www.prostep.de/en/solutions/opendxm/

[19] Raflik, M., *CAD*I Database-An Approach to an Engineering Database*, ECSC-EEC-EAEC, Brussels-Luxembourg, 1990.

[20] Raghothama, S. and Shapiro, V., *Boundary representation deformation in parametric solid modeling*, ACM Transactions on Graphics, Vol. 17, No.4, pp. 259--286, October 1998.

[21] Raghothama, S. and Shapiro, V., *Consistent updates in dual representation systems*, Computer-Aided Design, Vol. 32, pp. 463--477, 2000.

[22] Shapiro, V. and Vossler, D. L., *Construction and Optimization of CSG Representations*, Computer-Aided Design, Vol. 23, No. 1, pp. 4-20, 1991.

[23] Shapiro, V. and Vossler, D. L., *Efficient CSG Representations of Two Dimensional Solids*, Transactions of ASME, Journal of Mechanical Design, 113: 292-305, 1991.

[24] Shapiro, V. and Vossler, D. L., *Separation for Boundary to CSG Conversion*, ACM Transactions on Graphics, Vol. 12, No. 1, pp. 35-55, 1993.

[25] Zha, X. F. and Du, H., *A PDES/STEP-based Model and System for Concurrent Integrated Design and Assembly Planning*, Computer-Aided Design Vol. 34, pp. 1087-1110, 2002.