A Web-based Collaborative Feature Modeling System Framework

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

[1] DRC, School of MPE, Nanyang Technological University, Singapore
[2] CAD/CAM Lab, School of MPE, Nanyang Technological University, Singapore
[*] Email address: mysma@ntu.edu.sg (Ma Y. –S.)

Abstract: In this paper, a web-based collaborative feature modeling system framework is proposed. To integrate CAx applications, a four-layer information model is proposed. STEP (STandard for the Exchange of Product model data) is extended to build the product model structure for information sharing. Mapping mechanisms are also investigated to convert the EXPRESS-defined information types into the database schemas. The generic feature representation and geometrical data representation in database are given. Mechanisms for feature validation are explained.

1. Introduction

In a collaborative engineering environment, engineering tasks are always carried out by a group of distributed engineers who may use different applications. Therefore, information sharing among product development team members becomes the bottleneck. Although many research work [2, 3, 4, 5, 6] and commercial products [7, 8] have been carried out in this area, problems still exist. They fall into the following two aspects, information loss and data conflict.

Although many proposed systems are claimed to be CAD-neutral based on STEP and CORBA, they lack the necessary interoperability so far. In the process of data exchange, useful information such as features is often lost. Therefore it is not complete information sharing. On the other hand, CAD data is often stored in a file format, which means duplicated data and potential conflicts.

2. System Architecture

To enable information sharing among CAx applications, a web-based, database-driven, and feature-oriented system architecture is proposed as shown in Figure 1. The proposed system includes clients, application servers and a database server. The application servers include a web server, an application object server and a feature object server.

The web server contains a multiview data access interface (MDAI), a security manager and a session manager. MDAI provides shared access for multiple users. It can instantiate different views for different users according to user's requirements. Security manager is used to prevent unauthorized access to the product data. Session manager is responsible for controlling concurrent access by multiple users of the same data. The application object server provides different application packages for different users on the basis of feature object server. The product model manager is responsible for organizing information for multiple applications according to the user's requirements. This information includes feature model and

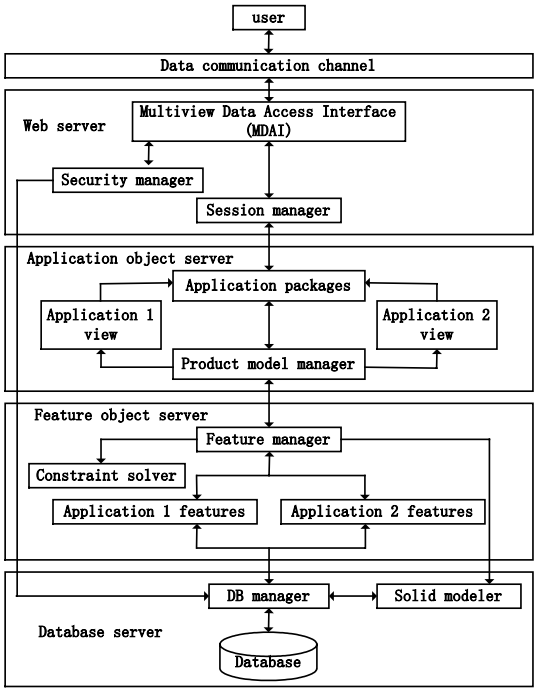

Figure 1. Overall system architecture

solid model (B-rep). To maintain the feature semantics during modelling operation, such as adding, deleting and modifying features, the feature manager will call the constraint solver and the solid modeler to validate the feature. Details for feature validation are explained in section 5. The constraint solver can check the validation of all constraints, which are part of the feature definition. The geometrical modeler can validate feature geometry. 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 so that they can be reorganized with great flexibility. The solid modeller provides general functions such as geometry construction, modification, and computation to support higher-level feature modelling.

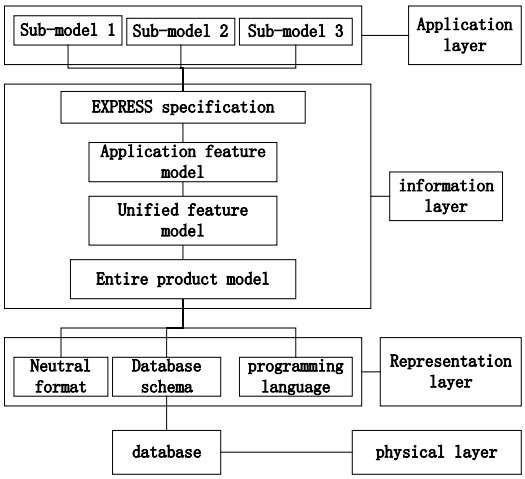

Figure 2. Four-layer information model

3. Information Model

In this research, the information model is built on the basis of an extended STEP framework. To achieve integration among CAx applications, the sharing of a common product model is crucial. The shared product model provides different views for various applications. Based on Zha's work [9], we propose the four-layer information model as showed in Figure 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 entire product model (EPM).

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 features can provide specific view of the EPM. Next, a unified feature model [10] is used to specify the generic feature-modelling framework for common definitions of different application features.

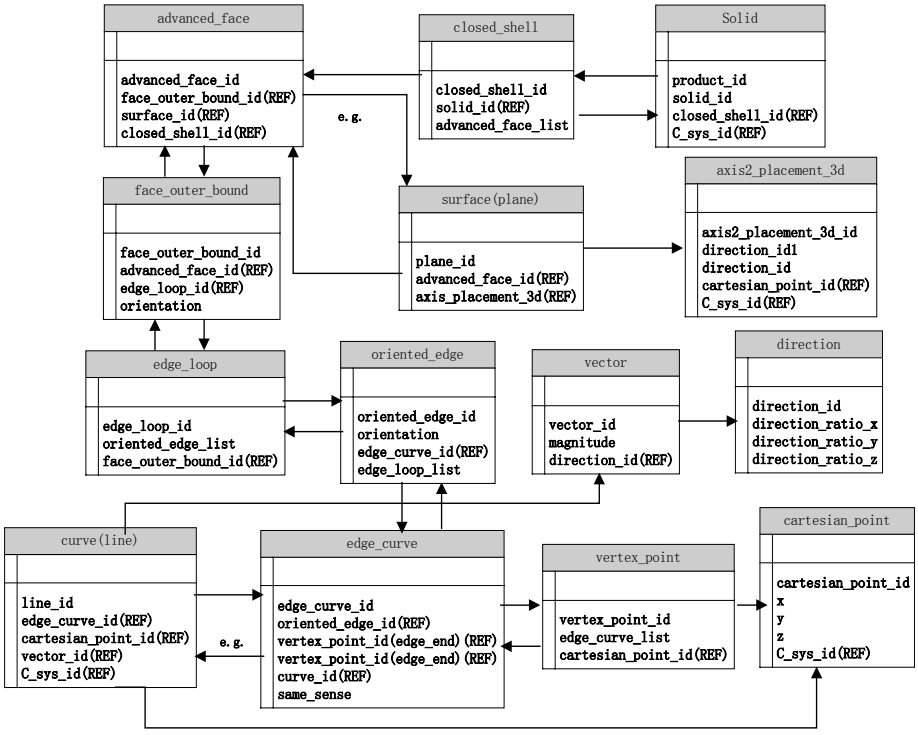

Figure3. Partial database schema for geometrical representation

Unified feature model allows different applications to define features with different configurations based on the predefined candidate types, such as geometrical representations, and the common processing methods. All the contents of EPM, application feature model and unified feature model are described in EXPRESS language.

4. Database Schemas

Under the four-layer information model structure, the entities at different levels shall be mapped to schema definitions for a potential comprehensive product

database such that arbitrary feature object structure can be represented. Details of mapping mechanism are given in [11].

A partial geometrical database schema is created according to STEP 42 [12] as shown in Figure 3. All attributes with suffix *id* (but without *REF*) represent object identifier (OID). A built-in data type called a *REF* represents the reference to OID. An arrow here represents such *REF* relationship between object types.

The generic feature representation in database can be expressed as Figure 4 under the framework of unified feature model. A feature has *feature_id*, *product_id* and *domain* as its attribute. A feature also contains a list of referenced entities, a list of constraints and a list of parameters. Parameters can be uniquely identified by a *parameter_id* from parameter table. *Referenced_entity* of feature includes entities (e.g. solid, faces, edges and vertices, etc.) or other features. *Entity_id* can uniquely identify the referenced entities stored in the 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. Maintenance of Feature Validity

Feature validity must be checked during feature modelling operations in order to maintain the feature semantics. A feature is valid as long as the feature satisfies all the relevant constraints and the feature geometry is valid. After each feature modelling operation, the solid modeller 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. The constraint manager maintains all constraints in a constraint graph for EPM, which contains sub-graphs for specific application views. Constraint manager solves constraints by calling the corresponding solvers according to different constraint types. For example, SkyBlue algorithm [13] can be used to solve local algebraic constraints in design domain. If a conflict of intra-application constraints occurs, local constraints solver can determine automatically which constraint should be satisfied 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*. Inter-application constraints can also be solved under the control of constraint manager according to the value of *domain_strength*. Its value, which regulates priority sequence of different domains, is predefined. Any conflict of inter-application constraints will be detected by

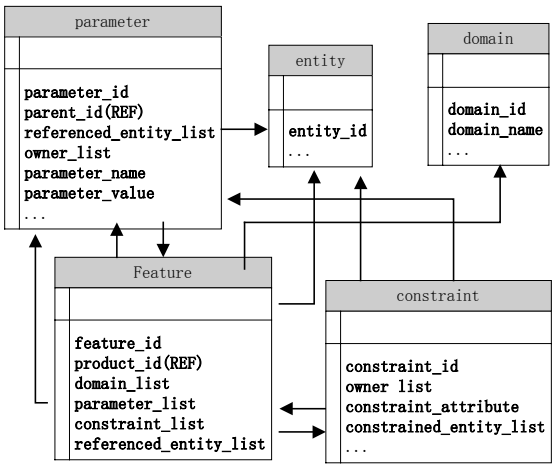

Figure 4. Generic feature representation in database

constraint manager, which can trigger the relevant applications and the constraint solver to re-evaluate the product model according to *domain_strength*. Only when all constraints are checked and feature geometry is validated, does feature validation finish.

6. Case Study

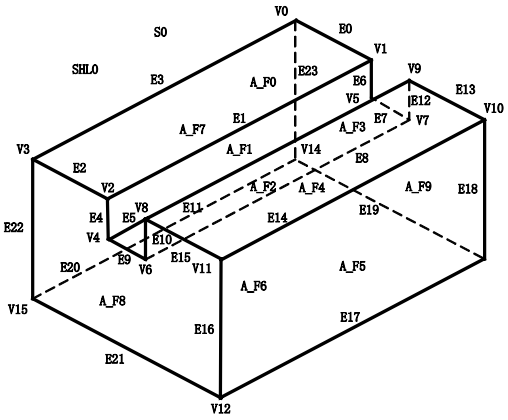

Figure 5 Geometrical representation of example part

A case study is carried out to testify whether product and process information can be well managed using the proposed database schema. The geometrical entities (e.g. shell, advanced_face, etc.) of an example part (block with through_slot feature) are explained in figure 5. Adopting the proposed feature representation schema, the example part can be express in a database as shown in Figure 6.

Functions for managing product information, such as *save*, *restore*, and *validate*, have to be developed. These functions are used to organize information for different application views according to users' requirements. Here, we only briefly explain the *save* and *restore* algorithms. Part information, which includes geometrical data, features and others, is represented as ENTITIES. ENTITY is a virtual class; it represents common data and functionality that is mandatory in all classes that represent permanent objects. *Save* algorithm can be expressed in step as follows: (a) Create an empty entity list and add the part to be saved to the list; (b) Get all entities such as solid, shell and so on from the part and add them to a graph map so that object pointers can be fixed as unique database object IDs; (c) Use such object pointers to call *save*

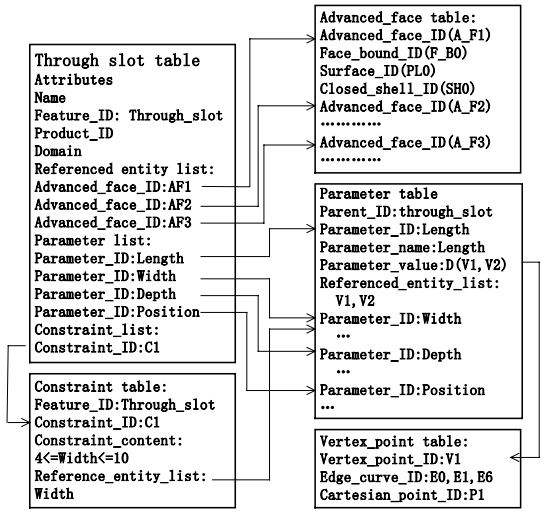

Figure 6 Through_slot feature in database

functions of the specific class (e.g. *point.save* (), *vertex.save* ()) to save part data to the database.

Restore algorithm has the following steps: (a) All the entity of a part are retrieved from the database by searching their linked Object IDs; (b) Reconstruct new objects; (c) Add all the entities to a newly generated object graph map; (d) Convert these IDs to genuine pointers; (e) Create an entity list and add all the entities to the list to form the part.

7. Conclusion

In this paper, a framework is proposed to enable information sharing among CAx applications. The proposed four-layer information model can integrate different applications with EPM, and allow the manipulation of application-specific information within sub-models. STEP information model has been extended; product and process information can be organized for multiple applications with great flexibility. A generic feature representation schema and a geometrical data representation schema in databases are given. Mechanism to maintain feature validation is described.

Reference

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

[2] H. J. Helpenstein, *CAD Geometry Data Exchange Using STEP,* ECSC-EEC-EAEC, Brussels-Luxembourg, 1993.

[3] Y.P. Zhang, *"An Internet based STEP Data Exchange Framework for Virtual Enterprises",* Computers in Industry Vol. 41, pp. 51-63, 2000.

[4] T. Dereli and H. Filiz, *"A note on the use of STEP for interfacing design to process planning"*, Computer-Aided Design, Vol. 34, pp. 1075-1085, 2002.

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

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

[7] *ACS software*. http://www.acssoftware.com/.

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

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

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

[11] Tang S. –H., Ma Y. –S. and Chen G., *"A Feature-oriented Database Framework for Web-based CAx Applications"*, accepted by conference of CAD04, 2004.

[12] *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.

[13] M. Sannella. *"The SkyBlue constraint solver"*, Technical Report 92-07-02, Department of Computer Science and Engineering, University of Washington, 1993.