Cloud Based Meta Data Driven Product Model

Arun Singh, B. Gurumoorthy, Latha Christie

Cloud Based Meta Data Driven Product Model

Arun Kumar Singh[1, 2], B. Gurumoorthy[2], Latha Christie[1]

[1]Microwave Tube Research and Development Centre (DRDO), Bangalore- 560013, India

arunsingh_aks@yahoo.com

[2]Centre for Product Design and Manufacturing, Indian Institute of Science, Bangalore - 560012, India

bgm@cpdm.iisc.ernet.in

Abstract. Product model using Core Product Model (CPM) and its extension Open Assembly Model (OAM) has been implemented with cloud compatible open source technologies. Implementation focuses on data structure for capturing assembly hierarchy, part details, part features and feature associations. Super Constraint Tolerance Feature (SCTF) model has been used to extract information from STEP file. A case study on Cylinder cap assembly has been taken to showcase the data structure and classes. Entity Attribute Value (EAV) model has been used to manage and store the product metadata and product data. Parts and assemblies are defined as entities, product metadata as attributes and product data as values has been stored in separate tables. Methods to populate product model at different stages of design has also been discussed.

Keywords: Cloud, PLM, Core Product Model, Open Assembly Model

1 Introduction

Cloud compatible technologies are gaining popularity in recent times. Most of the traditional Product data models were developed keeping in mind single-user model based environment at single/multiple location installations. With the advent of the Internet more and more products are designed and manufactured globally in a distributed and collaborative environment. Cloud compatible platforms also ena-

ble to incorporate open source web technologies. Web based systems can be used to take advantage of social networking tools for knowledge capturing.

Product model should be able communicate with other systems in a plug and play manner by providing standard definitions for interfaces. For example system will be interoperable if Geometry is captured or translated using STEP from/to traditional CAD tools. Incorporation of already existing standards like STEP to define geometry will reduce efforts in system compatibilities and import/export of geometry data in/out of product model at different stages of product life. STEP is a mature and widely used standard for the exchange of product data. In practice, STEP tends to be invoked only late in the product development process, after all design decisions have been made and when the product is ready to be manufactured or assembled. Thus, STEP is used for the exchange of information that is the outcome of design activities, rather than for the information produced and used through the development of the design. STEP provides no support for design evolution, for the early phases of design when descriptive information is sparse.

The product model presented here may be seen as the precursor for STEP in the lifecycle of a product, capturing all the information relevant to the ongoing design process until the product design is firmed up, approved and committed to manufacturing.

2 Product Models

2.1 Core Product Model (CPM)

National Institute of Standards and Technology (NIST) proposed Core Product Model (CPM) to cater issues of interoperability and having a product model, which can cater needs of Product Lifecycle Management (PLM) from conceptual design stage to end of product life. The objective of the NIST CPM [1] was to provide a base-level product model that is: not tied to any vendor software; open; non-proprietary; simple; generic; expandable; independent of any one product development process; and capable of capturing the engineering context that is most commonly shared in product development activities. CPM focuses on artifact representation including function, form, behavior and material, physical and functional decompositions, and relationships among these concepts. The model is heavily influenced by the Entity-Relationship data model [2].

The model consists of two sets of classes, called object and relationship. The two sets of classes are equivalent to the Unified Modeling Language (UML) terms of class and association class, respectively. CPM supported the notions of form, function, and behavior.

2.2 Open Assembly Model (OAM)

An object-oriented definition of an assembly model called the Open Assembly Model (OAM) is explored, which is as an extension to the CPM. The assembly model represents the function, form, and behavior of the assembly and defines both a system level conceptual model and associated hierarchical relationships. [3] The schema incorporates information about assembly relationships and component composition; the representation of the former is by the class **AssemblyAssociation**, and the model of the latter uses part-of relationships. The class **AssemblyAssociation** represents the component assembly relationship of an assembly. It is the aggregation of one or more **ArtifactAssociation**.

An **Assembly** is a composition of its subassemblies and parts. A **Part** is the lowest level component. Each assembly component (whether a sub-assembly or part) is made up of one or more features, represented in the model by **OAMFeature**. The **Assembly** and **Part** classes are sub-classes of the CPM **Artifact** class and **OAMFeature** is a subclass of the CPM **Feature** class. In CPM, **Geometry** and **Material** aggregate into **Form**. **Form** and **Function** aggregate into CPM **Feature** class.

The class **AssemblyFeatureAssociation** represents the association between mating assembly features through which relevant artifacts are associated. The class **ArtifactAssociation** is the aggregation of **AssemblyFeatureAssociation**.

3 Cloud Computing

Cloud-based Product model refers to a service-oriented networked product development model in which service consumers are enabled to configure, select, and utilize customized product realization resources and services ranging from CAE software to reconfigurable manufacturing systems. This is accomplished through a synergetic integration of the four key cloud computing service models: Infrastructure-as-a-Service (IaaS), Platform-as-a-Service (PaaS), Hardware-as-a-Service (HaaS), and Software-as-a-Service (SaaS) [4].

Cloud-based Product model refers to a networked design model that leverages cloud computing, service-oriented architecture (SOA), Web 2.0 (e.g., social networking tools), and semantic web technologies to support cloud-based engineering design services in distributed and collaborative environments [5].

The key issue in improving design communication is the extent to which design engineers fully understand a complex design process, in particular, design tasks that need to be finished or identification of individuals from whom specific information can be accessed. In traditional collaborative design systems, communication can be seen as a one-way process with a linear sequence of design. Because of the use of social media in cloud based system, design communication can be im-

proved through multiple information channels (e.g., social networking tools and product review tools) in which information flow can take place in multiple directions [5]. In traditional systems, computer-aided application tools are standalone systems and are designed for single user without communicating and collaborating with others [6]. In Cloud based system, engineering design requires more communication and collaboration within and across organizations on the modeling, analysis, and optimization of a design.

Designers can access manufacturing information like process sheets, inspection reports, issues associated in manufacturing product components and all the related information of a product while designing new products or modifying the existing products. Manufacturing personnel can communicate with the designer and all the communication data can be associated with the product form (artefact attribute) and can be saved for future references.

4 Attribute sets (Meta data) Driven Product Model

A cloud based/compatible data structure of CPM/OAM has been created, which can be used by designers to capture, store and retrieve product data. In early stages of product development when detailed geometry information is not available, it becomes difficult to associate function and behavior information to the product. There can be scenario where product is to be designed from sketch and scenario, where some reference/existing product was designed in the same environment, is available and only needs to be modified for better performance, cost reduction or new added features. Product model should be able to adapt and work in both the situations and provide methods which can be used to capture, store and retrieve design rational, assembly and tolerance information from the conceptual design stage up to the timeline where designer deals with the function and performance of product.

In situations where reference product is available and products only need feature modifications, old product design repository should be able to create new instances of whole project either with class structure skeleton model (without form, function, behavior data) or with full product history including data. A data driven attribute repository has been proposed which can be a platform to add new features for existing product designs. This repository can act as base model to define artefact. Any artefact definition would need attributes to define the form, function or behavior. Artefact may have one set of attributes for form (Geometry and topology) and others to define artefact's function and behavior. Data driven approach is adopted having EAV (Entity Attribute Value) model to capture the product information. Flow chart for attribute set creation and selection has been shown in figure 1.

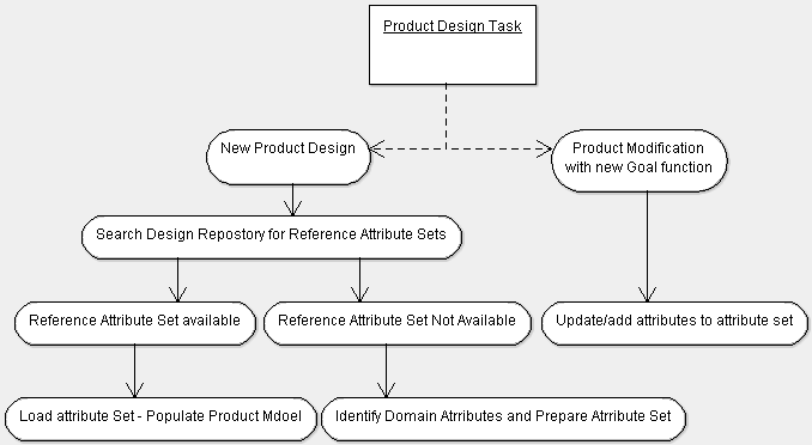

Fig. 1. Flow Chart for preparation of Product Model Attribute set (Meta data)

Semantic development is fully domain driven approach and it cannot be pre-built in Product model, which will make it domain specific product model. Main focus for developing the product Meta data in form of attribute set comes on designer. In multi domain design scenario, designers from different domain expertise can prepare their respective domain attribute sets, which in turn forms a collective attribute set definition.

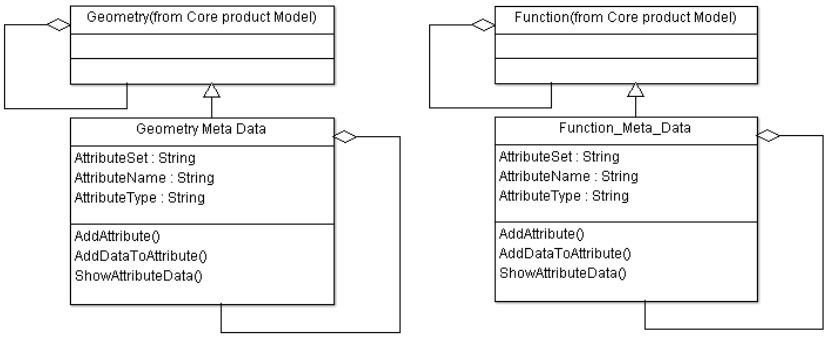

Fig. 2. Extended Geometry and Function Class having methods for attribute sets and data holders

Geometry_Meta_Data class has been inherited from CPM's geometry class. Geomtery_Meta_data contains variables to define attribute sets, attributes name and types. It contains functions *AddAttribute(), AddDataToAttribue(),* and *ShowAttributeData()* to store/retrieve attribute sets and attribute data to database. *AssemblyFeature* and *CompositeFeature*, which are inherited from OAM's Feature class, have been further extended to AssemblyFeature_Meta_data and CompositeFeature_Meta_data. Class structure and its functions are shown in figure 2.

These methods include functionality to make a connection to database and run required database queries like INSERT/UPDATE/SELECT or DELETE.

In Entity Attribute Value (EAV) model Entity definitions, attribute definitions and attribute values are stored in separate tables of database. This makes attribute set (meta-data) definition flexible as entities and attributes can be added or deleted on the fly whenever need arises. The product model need not be changed for addition or deletion of domain specific attributes. Product model, Meta Data and product data exists separately. Product models can grow from conceptual design level having very few entities & attributes into fully developed attribute set at the advanced stages of Product Design.

5 Implementation

5.1 Programming Platform and Data Structure

Meta-Data Driven Product model is implemented on Cloud compatible platform. Open source resources are used to implement product model. Programming language PHP has been chosen on Apache web server. PHP is capable of implementing Object Oriented Class structure / instances from UML defined diagrams. UML diagram has been generated using open source code ArgoUML. MySQL is the database server to store product metadata and product data.

SCTF model has been taken to extract information from neutral file format STEP and to suffix constraints and features in addition to assembly hierarchy [7]. The model content includes nominal geometry (features), constraints (including dimensions, mating conditions, and assembly constraints), tolerances (including datum reference frames), degrees of freedom (DOFs), and assembly hierarchy. The model is named *Super-Constraint-Tolerance-Feature-Graph-Based Model (or the SCTF Model for short*). The whole model is created from top down, and lower level data is gradually populated when the higher-level data is available. The order is "The general tree, parts, features, constraints, tolerances, DOFs".

5.2 MySQL Table Structure for Assembly Hierarchy

The first step to implement data structure for SCTF is assembly hierarchy definition. Assembly hierarchy is implemented using General Tree. Data Structure for general tree needs definition of Tree node class and general tree class. General tree node needs data member variable (data) and three pointers – its parent, its child and its siblings. Classes need to be defined to traverse, modify and retrieve data from the general tree.

General tree represents a hierarchical data. MySQL is a relational database. Tables of relational database are not hierarchical (like XML), but are simply a flat list. Hierarchical data has a parent-child relationship that is not naturally represented in relational database table. Hierarchical data is a collection of data, where each item has a single parent and zero or more children (With the exception of the

root item, which has no parent). There are two models for dealing with hierarchical data in relational database management systems (RDBMS) – adjacency list model and the nested set model.

5.2.1 Adjacency List Model

In the adjacency list model, each item in the table contains a pointer to its parent. The topmost element has a null value for its parent. The adjacency list model has the advantage of being simple.

5.2.2 Nested Set Model

In the nested set model, we can look at hierarchy in new way, not as nodes and lines but as nested containers. Hierarchy is still maintained, as parents envelop their children. These can be represented in a table through left and right values to represent the nesting of nodes. We start numbering at the leftmost side of the outer node (Main assembly) and continue to the right. This approach is called *modified preorder tree traversal algorithm.* The full tree can be retrieved through the use of *a self-join* query that links parents with nodes on the basis that a node's left value will always appear between its parent's left and right values. Functions for finding all the leaf nodes, retrieving a single path, finding the depth of nodes, depth of a sub-tree, finding the immediate subordinate of a node, adding new nodes, deleting nodes are easy to implement compared to adjacency list model.

Next Layer of SCTF Graph consists CTF graph, which is a nested *doubly-linked list* data structures. At the first layer, it is a *list* of "parts" contained in the assembly / subassembly node. At the second layer, each "part" is composed of a *list* of geometric "features" .At the third layer, each geometric "feature" contains its basic geometric data, a *list* of geometric constraints, a *list* of tolerances, a *list* of DoFs, and a *list* of associated points.

6 Case Study - Cylinder Cap Assembly

A cylinder-cap assembly has been taken to implement product model. 3D modeling software SolidWorks is used to model the assembly. A STEP translated model has been taken. Assembly consisting of 4 unique parts (subartifacts) namely cylinder, cap, key and bolts, is shown in figure 3.

Assembly hierarchy is implemented using nested set model shown in Table 1. Nested set model makes it easy to query the parts contained in an assembly. Like any part which is having its Comp_left and Comp_right values in-between 1 and 18 will be a part of Assem_0.

ArtifactEntities in Table 2 is listing all the assemblies and parts associated with Cylinder Cap Assembly. In CPM, to store artifact data information instance has been mentioned which can hold the values of Properties. EAV (Entity Attribute Value) model is used to store these values. For example Table 3 is defining the ArtefactFeature Entities. Table 4 is holding attributes for these entities and Table 5 contains the attribute values (Product Specific data). Table 3 & 4 are containing

the Meta data for product, whereas the real product data is in table 5. When designing a new Cylinder Cap Assembly (Conceptual), it can be started with the Meta data i.e. entities and attributes. These tables can be built based on the information available with the designer at different stages of Product Lifecycle. Designers have flexibility to add new form or function feature attributes.

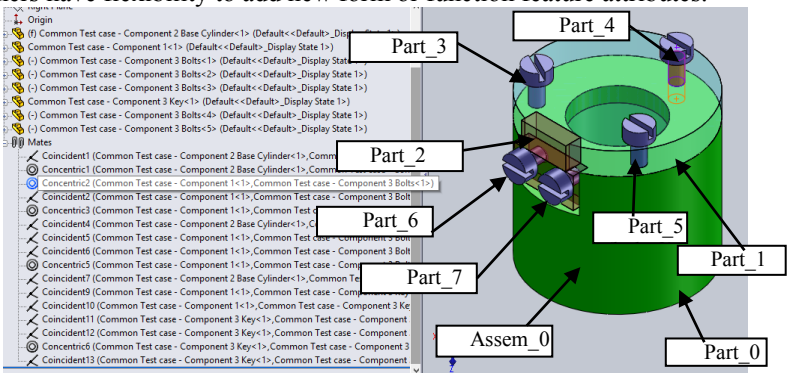

Fig. 3. Cylinder Cap assembly with Part Details and Constraints (AssemblyFeatureAssociations)

Table 1. Nested Set Model Table for assembly hierarchy

Artefact_id	Artefact_name	Comp_left	Comp_right
1	Assem_0	1	18
2	Part_0	2	3
3	Part_1	4	5
4	Part_2	6	7
5	Part_3	8	9
6	Part_4	10	11
7	Part_5	12	13
8	Part_6	14	15
9	Part_7	16	17

Table 2. ArtefactEntities

Artefact_id	Artefact_name	Artefact_Type (Part/Assembly)	Artefact_Label
1	Assem_0	Assembly	Cylinder Cap Assembly
2	Part_0	Part	Cylinder
3	Part_1	Part	Cap
4	Part_2	Part	Key

5	Part_3	Part	Bolt_0
6	Part_4	Part	Bolt_1
7	Part_5	Part	Bolt_2
8	Part_6	Part	Bolt_3
9	Part_7	Part	Bolt_4

Table 3. ArtefactFeature_entity

ArtefactFeature_id	Artefact_id	ArtefactFeature_name	ArtefactFeature_type
1	2	Centre_hole	Hole
2	2	Fastening_hole_0	Tapped Hole
3	2	Fastening_hole_1	Tapped Hole
4	2	Fastening_hole_2	Tapped Hole
5	2	Fastening_hole_3	Tapped Hole
6	2	Fastening_hole_4	Tapped Hole
7	5	Threaded_bolt_0	Threaded Pin
8	6	Threaded_bolt_1	Threaded Pin

Table 4. ArtefactFeature_attributes (For Fastening_hole_1)

ArtefactFeature_attribute_id	ArtefactFeature_id	ArtefactFeature_attribute_name
1	3	Centre
2	3	Axis
3	3	Radius
4	3	Height
5	3	thread_size

Table 5. ArtefactFeature_value (For Fastening_hole_1)

ArtefactFeature_value_id	ArtefactFeature_Attribute_id	ArtefactFeature_value
1	1	22.6873
2	2	0,0,1
3	3	5
4	4	15
5	5	M5

The AssemblyFeatureAssociation information is shown in Table 6. It lists Part feature's association with other part's feature. These can be implemented by doubly linked list.

Table 6. AssemblyFeatureAssociation

ArtefactFeature_id1	ArtefactFeature_id2	Association

3	7	LocationConstraint_concentric
4	8	LocationConstraint_concentric
9	13	LocationConstraint_coincident
10	14	LocationConstraint_coincident

7 Conclusion

The data structure has been created using CPM and its extension OAM. STEP File containing geometric data is used to extract the assembly hierarchy and part details. Case Study implementation shows that CPM / OAM can be implemented using open source cloud compatible technologies. Data structure containing Meta data and Product data for assembly hierarchies, parts, features and its associations (Constraints) has been implemented and demonstrated in Cylinder Cap assembly case study. The subsequent steps to SCTF graph are data structure for tolerances and artefact feature's DOFs, which can assist in Assembly planning and Goal based Tolerances allocation.

8 References

1. Steve J. Fenves, Sebti Foufou, Conrad Bock, Ram D. Sriram (2008) CPM2: A Core Model for Product Data, Journal of Computing And Information Science In Engineering., ASME, 8 (1), 2008.
2. Sebti Foufou, Steve J. Fenves, Conrad Bock, Rachuri Sudarsan, Ram Sriram (2005) A Core Product Model for PLM with an illustrative XML implementation, Proceedings of the PLM International Conference on Product Lifecycle Management, Inderscience Publishers, Lyon, France, pp. 21-32, July 2005.
3. Sudarsan Rachuri, Young-Hyun Han, Sebti Foufou, Shaw C. Feng, Utpal Roy, Fujun Wang, Ram D. Sriram, Kevin W. Lyons (2006) A Model for Capturing Product Assembly Information, Journal of Computing and Information Science in Engineering, *6*(1), p. 11-21, 2006.
4. Wu D, Rosen DW, Schaefer D. Cloud-based design and manufacturing: status and promise. In: Schaefer D, editor. (2014) Cloud-based design and manufacturing: a service-oriented product development paradigm for the 21st century. London, UK: Springer; 2014. p. 282.
5. Wu D, Schaefer D, Rosen DW. (2013) Cloud-based design and manufacturing systems: a social network analysis. In: International conference on engineering design (ICED13), Seoul, Korea, 2013.
6. Red E, French D, Jensen G, Walker SS, Madsen P. (2008) Emerging design methods and tools in collaborative product development, J. Comput. Inf. Sci. Eng. 2013; 13(3):031001.
7. Shen, Z., Shah, J., and Davidson, J., Analysis Neutral Data Structure for GD&T," J. Intell. Manuf., 19(4), pp. 455–472, 2008.
8. Steve J. Fenves, Sebti Foufou, Conrad Bock, Rachuri Sudarshan, Ram D. Sriram (2006) CPM2: A Core Model for PLM Support, Frontiers Presentation, 2006.