OBJECT-LEVEL INFORMATION MANAGEMENT FOR DYNAMIC DESIGN PROCESS

Hua Sun and W. F. Lu

Abstract

Information management in product development lifecycle has been identified as a critical technology to help companies to reduce design lead-time through the efficient and effective information exchange among designers. However, compared to most of object-level enterprise information management systems characterized by using individual data as information exchange unit, the design information management is currently still at preliminary document level characterized by using a document as basic information unit. This is due to the difficulties from the complexity and dynamic nature of design processes. This paper proposes a design modeling methodology specifically for the development of an object-level information management system for complex and dynamic design processes. It captures and formalizes process infrastructure and related information and knowledge, facilitate fast re-configuration of the information management system, and support the integration between information management function and knowledge support application. Based on this modeling methodology, system architecture for object-level information management is also presented.

Keywords: design process modeling, information management, knowledge support, fast-reconfiguration, dynamic design processes

1. Introduction

In the intensive competitive environment today, frequent change of customer preferences, demands of faster new product introduction, and more speedy technology generation have led to very short product life cycles. This requires firms to have fast product development to meet the pressure of time-to-market. During product development, design phase usually takes the longest time and is most difficult to control. Consequently, how to reduce product design lead-time while keeping product costs low and quality high are becoming key business factors for manufacturing enterprise to obtain competitive advantages.

Information management in product development lifecycle has been identified as a critical technology to help companies to reduce design lead-time through efficient and effective information exchange among designers. Currently, typical information management tools for product development are PDM tools. They allow users to define abstract level workflow, and electronic documents that hold product data could be distributed along the defined workflow. One of the values of such tools is that they change paper design documents into electronic documents and facilitate search and exchange of such documents. This definitely saves designers lots of time spent on finding product documents and hence improves design efficiency. However, as product development lifecycle becomes shorter and shorter under the pressure of time-to-market with increasingly complex product and process, just exchanging electronic documents is no longer sufficient in new design environment. For instance, the lack of a formal product representation that includes function, behavior, and structure has

been identified as a shortcoming of existing PDM systems [1]. Furthermore, the existing PDM tools could not actively provide designers with the exact useful individual information they need. When designers need certain specific data/information, they still are forced to wade through large amount of mostly irrelevant information in the documents to find it. This is due to the intrinsic limitation of document-level information method. In the document-level information management method, lots of information would be captured in one document and the document is the basic unit for the querying, retrieving, and indexing mechanisms of the system. Consequently the individual information/data generated along design processes could not be distributed separately to the right place. This also leads to data redundancy problem of PDM tools. In product design, it is very common that different documents could use certain portion of common information/data. However, since such information/data could not be distributed individually to downstream processes, they must be entered many times into different documents, which lead to data redundancy. This kind of document-level method is actually quite preliminary compared to the object-level information management tools used for enterprise processes such as ERP tools. Most of ERP tools model information as sets of classes related in some manner. Each information object represents an instance of a class, which includes attributes and methods. The information object is the basic unit for query, and even each attribute of the information object could be handled individually based on the object-oriented information model. This enables information to be distributed and streamlined along enterprise processes. All information/data once entered at upstream processes could be automatically distributed to downstream processes as needed. For example, as shown in Fig 1, the sales order information captured at the sales order preparation process, would be distributed to many downstream enterprise processes, such as creating job order, delivery order preparation, billing invoice generation etc. For the information objects of these downstream processes, the system could automatically infuse their attribute value that are corresponding to the attributes in sales order object.

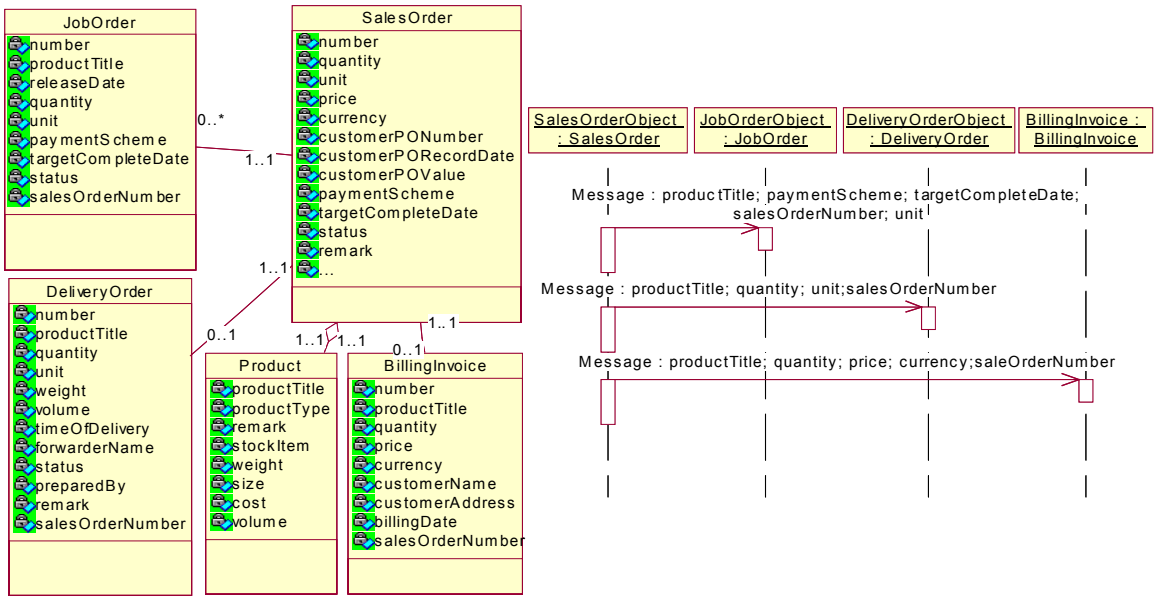

Figure 1. Example of object-level information management for enterprise processes

In this way, redundancy of information could be minimized and all information could be streamlined according to the enterprise process flow. However, for product design area, no existing tools are available for such kind of object-level information management, focusing on individual information/data generated along design processes. The difficulty firstly comes

from the dynamic nature of design process. The design process may be different for different types of products. Even for the design of same type of product, the structure and sequence of design processes may be different so as to satisfy different goals or strategies when collaborating with different partners in design chain environment. Although computers could assist in information management for the design process, they operate only with formal, repeatable, and rigorously defined processes [2]. To solve this conflict, it requires that the information system for design process must have fast, flexible reconfiguration capability. Another challenge comes from the complexity of design processes. As products are more and more technologically complicated, a complete design procedure of it may involve thousands of tasks and all the tasks may be coupled in some manner [3]. To realize the object-level information management for design processes, it requires a formalized design process model to represent the complex design procedures and a standardized information model to capture comprehensive information embedded in design processes. One thing worthy of mention is that object-level information management is different from object-oriented PDM tools. The object-level information management in this paper indicates that the information object, which holds individual data in different attributes, is the basic information unit. On the other hand, some PDM tools are developed based on object-oriented techniques. In such systems, the software design, coding, and even the database are based on the object-oriented method. The information unit is an object called document object, which in essence transforms hardcopy document into electronic one with attributes such as the document number, document owner, document release time, and document content etc. The document content attribute corresponds to the main body text of hardcopy document. Actually the information management is at document level, because the information unit for query is still the document object and the lots of information held in the document content attribute could not be queried and distributed individually.

To address the above issues, this paper proposes a design modeling methodology specifically for the development of an object-level information management system for complex and dynamic design processes. It captures and formalizes process infrastructure and related information and knowledge, facilitate fast re-configuration of the information management system, and support the integration between information management function and knowledge support application. Based on this modeling methodology, system architecture for object-level information management is also presented.

2. Design modeling

In the past decades, many researches have been conducted on design process modeling. For example, Eppinger [3,4] from MIT have developed the DSM method for sequencing design task flow. Lu and Cai [5] from the University of Southern California have developed a generic collaborative design process representation model based on the Socio-Technical framework. Melo and Clarkson [6] from Cambridge Engineering Design Center developed a design process model that allows dynamic planning and flexibility in task definition. Process model has been developed by Pahl & Beitz[7,8] for conceptual design. However, the structure and content of the above models could not satisfy the requirements of information management for design process. For instance, the design processes were modeled at very high level of abstraction and hence could not facilitate the system to distribute information to design tasks at detial level. Some models only foucsed on design task sequence but could not capture and generalize comprehensive process properties and characteristics. Further, design processes of the above models coupled tightly with product and hence could not satisfy the

fast reconfiguration requirements of dynamic design environment. Therefore, a new design modeling method is needed specifically for imformation management for design processes.

2.1 Modeling strategy

Firstly, we recognize that design process is related to products but not all depends on the specific product to be designed. Different products sometimes may have certain similar portion of design processes. However in some circumstance even same products may partially have different set of design processes due to different business strategies or collaboration with different partners. Since it is hoped that the information management mechanisms are generic enough to handle design processes of various products, the design process itself should be modeled in such a way that its content would not be tightly coupled with any specific product to achieve certain independence. This could be solved by decoupling process information and product information in the model. The process information represents the properties of individual design task itself, such as the process's objectives, functions, strategy, rational, constraints, technology used, etc. On the other hand, the product information shows the product properties such as geometry, functions, behaviors, cost, materials etc.

Further, we recognize that along design processes, certain design tasks may need to use specific domain knowledge as part of pre-requisite input information. It is desirable to separate such kind of input information from those information generated by internal design tasks, because the reconfiguration mechanisms of information management system for these two types of information would be different in terms of complexity. As for the domain knowledge, it is not distributed to anywhere but only used by its owner process to generate certain process deliverables; and therefore, it would not have any dependence on process sequence and structure. On the other hand, the internal generated information would be distributed to downstream processes to generate final product, and hence the information flow would need to be re-organized when design process infrastructure is changed.

2.2 Content and structure of design modeling

According to the above modeling strategy, the design modeling would be introduced from the following three different aspects, based on object-oriented approach.

(1) Process model:

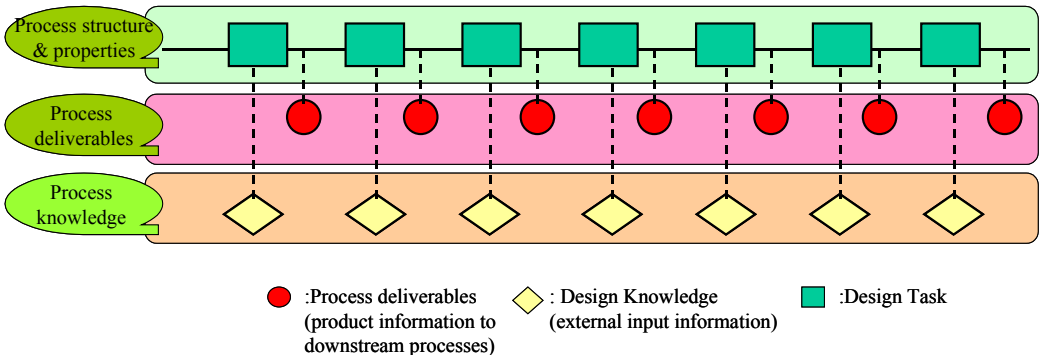

Figure 2. Process model architecture

The design process model has a three-level architecture, as shown in Fig 2. The first level is process infrastructure, which would be made up of process objects that hold process properties. The process objects would be organized systematically to represent the structure

and sequence of design processes. As shown in Fig 3, the process infrastructure comprises a set of processes represented in workflow manner at multiple abstraction levels, from most generic level to most specific level. The process object at higher degree of abstraction is super process object, and could be further broken down to a set of related sub-process objects, according to the hierarchical nature of the design processes. No matter at what level of abstraction, the process infrastructure would be basically represented by two basic classes, Process class and ProcessToProcessLink class. There are five types of relationships among design processes, including iteration type, precedence type, couple type, composition type, and inherence type. The iteration type of relationship indicates that there is an information loop within the process flow between the two linked process objects. The precedence type means one process depends on the other process for input information. The couple type is used for the two processes that require information from each other. The composition type indicates one process is the sub-task of the other process. The inherence type is only applicable for the processes that need special additional attributes and methods in order to proceed certain particular applications such as simulation and optimization.

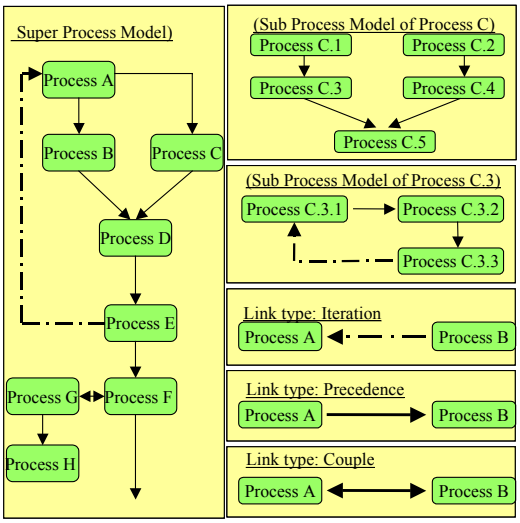

Figure 3. Example of the design process structure

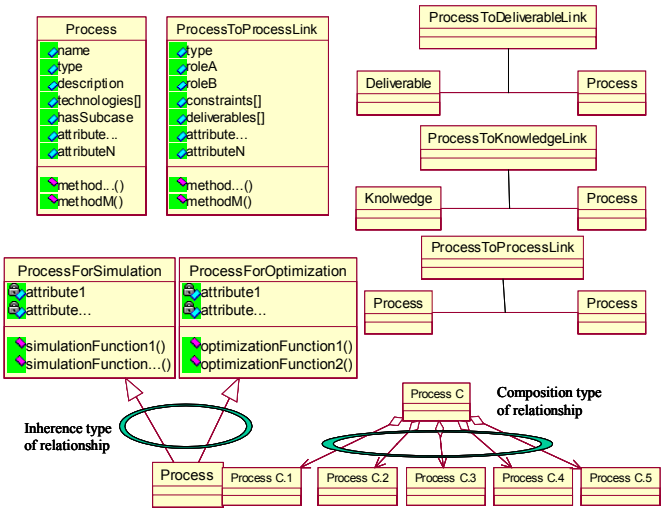

Figure 4. Classes in process model

The second level of design process model defines process deliverables, which finally put together to become the product satisfying specific customer requirements. Design processes are a set of interrelated activities aimed to generate a certain process output. The second level of the process model would identify and captures properties of such process output and their distributing flow from upstream processes to downstream processes. The properties of each process deliverable are in essence specific product information of each design instance. They could be the selected material, value of product parameters, and even some drawings for specific product, etc. Their instance values are usually different for each design instance and would be distributed according to process infrastructure from upstream to downstream, based on the links between process objects and information objects.

The third level of design process model focuses on process knowledge. As mentioned before, the process knowledge would be used by certain design tasks in order to generate appropriate process deliverables. Such knowledge is usually static domain knowledge. Since it is identified for specific process and only used by its owner process, it has no dependence on each other. The knowledge object would be linked to process object through ProcessToKnowledgeLink object, as shown in Fig 4.

As a summary, the pre-defined classes for process model are simply: Process, Deliverable, Knowledge, ProcessToProcessLink, ProcessToInformationLink, and ProcessToKnowledgeLink. The relationship between process, information (process deliverables), and knowledge would be captured through relationship objects. In such way, design process could be decoupled from specific product information / knowledge. This characteristic of independence would make process model more generic and also improve its scalability. Meanwhile, the relationship objects would facilitate the reconstruction of the information flow and knowledge flow according to the change of process sequence and process structures when doing information system reconfiguration.

(2) Information model:

As shown in Fig. 5, the information model has a two-level architecture. The first level is the relationship model to capture and represent the inter-relationship between the process deliverables that have been identified and linked to process objects in process model. It has basically two pre-defined classes: Deliverable (same as the class used in process model), and DeliverableToDeliveabelLink. The link class would define relationship properties such as the relationship type, constraints of relationship, and the pointers to information objects etc.

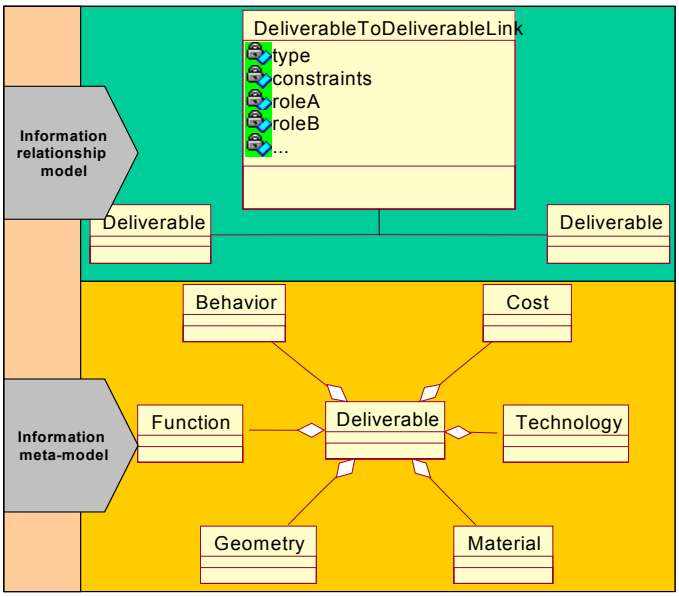

Figure 5. Structure and classes of information model

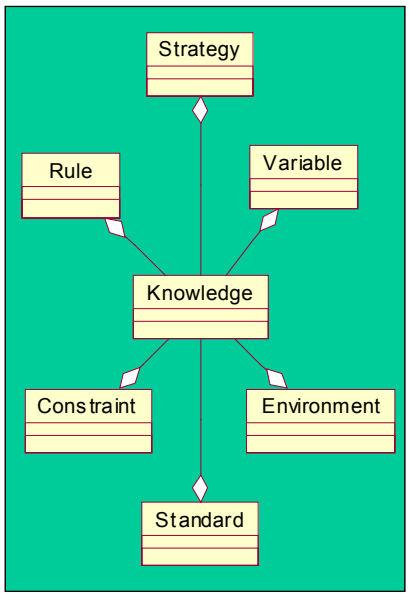

Figure 6. Structure and c lasses of Knowledge model

To avoid that the deliverable object holds too many attributes, the second level of information model would provide meta-model for each process deliverable object. As mentioned before, the process deliverables, distributed from upstream tasks to downstream tasks, are part of final product solution. Since product has many different aspects of properties such as material, behavior, function, structure, technology, cost, geometry etc [8], each process deliverable could also be interpreted from at least one or more of such perspectives. As shown in Fig. 5, the information meta-model would have following pre-defined classes: Structure, Behavior, Function, Geometry, Cost, Technology, and Material. Each class would have its specific attributes that captures particular category of product related data. In this way, the comprehensive properties of process deliverables could be classified and represented by these six basic classes. Each object of the information meta-model could be indexed, queried, and retrieved individually and more selectively for specific purpose.

(3) Knowledge model:

Many knowledge models have been developed to support knowledge-based engineering applications such as MOKA knowledge models [9] and function-behavior-form model [10]. They usually comprise of comprehensive and complicated knowledge objects to provide working environment for various knowledge application methods. However the knowledge model in this paper is developed to classify and provide a formalized meta-representation specifically for the knowledge objects that have been identified and linked to process objects in process model. As shown in Fig.6, the knowledge model has simply six basic classes: Rule, Constraint, Strategy, Guideline, Criteria, Standard, Environment, and Variable. These six classes actually specify the type, structure, and properties of the knowledge requested by individual design process. Since the knowledge objects are part of pre-requisite input information of design tasks, the information management mechanism would handle the query, retrieval, and distribution of such knowledge objects just like it deals with information objects, but not evaluate, select, analyze, or calculate the value for any properties of these knowledge objects. Actually the intelligent work is usually taken care by knowledge-based applications such as knowledge support in design. On the other hand, although the knowledge model in this paper is not developed for knowledge-based application, it could act as the interface to integrate information management with knowledge-based application. Take knowledge support in design for instance, it could extract the knowledge support requirements from the knowledge objects that are defined in the knowledge model, and then find the useful knowledge for the specific requirement through its algorithms and methods, and finally exports its result to infuse the attributes of the knowledge objects of the information management system. In this way, the knowledge model of the information management system could facilitate the knowledge support application to select right method and mechanism of knowledge processing for specific design process, while the knowledge support application could provide the instance value for the attributes of knowledge objects of information management system for each design instance.

2.3 Reconfiguration

In product design environment, design process structure, sequence, and even content of certain design task may need to change as the product changed or the design partner changed. Since the information management system is aimed to manage the information embedded in design processes, the system reconfiguration would be based on the design process model reconfiguration. The design process modeling methodology introduced above would facilitate fast system reconfiguration from following aspects.

The process object in process model does not hold any product related information or knowledge. Therefore the first layer of process model, the process infrastructure and properties, will not be affected much by the change of product within the same product family. Only little amendment in task sequence and process structure may need to satisfy different design specifications. This could help to achieve greater external product variety with less internal system reconfiguration.

Further, the product related information (process deliverables) are captured at the second layer of process model and are linked to process objects through relationship objects. The information objects would be distributed to the downstream processes that directly linked to their owner process. They would move with the change of position of their owner process objects and thereby the information flow could be automatically re-organized along with the reconfiguration of process infrastructure. For example, as shown in Fig.7, in the initial design scenario the deliverable A would be passed to process B and process C, which are

downstream processes of process A. When the sequence among process A, B, C is changed to the second scenario, the information flow would become that the deliverable A and deliverable B would be passed to process C.

In addition, the process knowledge objects modeled at the third level of process model are linked to process objects through relationship objects, and these knowledge objects are only used directly by their owner process and will not be distributed to anywhere. Therefore, the reconfiguration of process infrastructure and properties won't affect the querying, retrieving, and distributing mechanisms for such knowledge objects at all. The knowledge objects could always be provided to designers at their owner processes as long as the relationship objects exist. Of course, the instance value of these knowledge objects could be different for different product design episode, but how to select, analysis, and provide right instance value to these knowledge objects of information management system is the responsibility of knowledge support application.

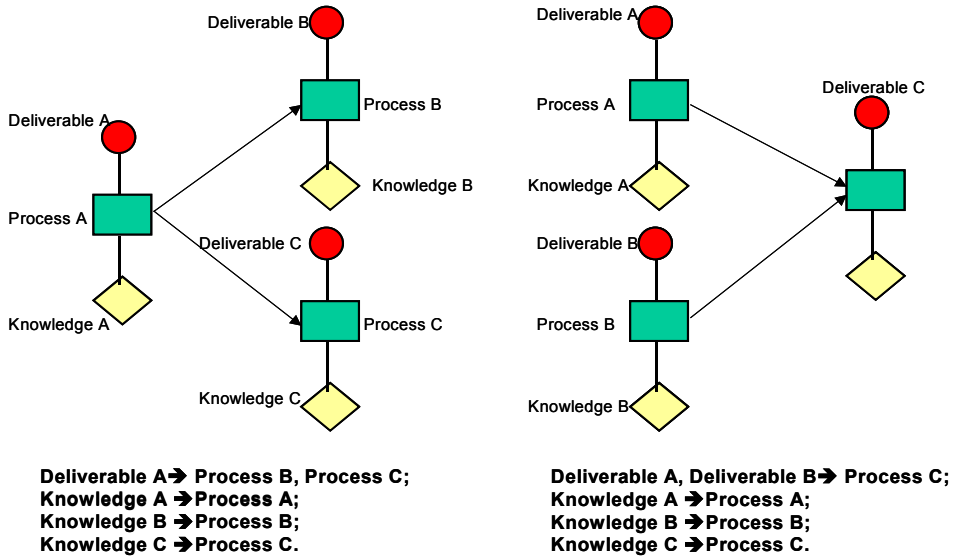

Figure 7. Example of design process model re-configuration

3. System architecture for object-level information management

The above design modeling methodology would build up a design environment specifically for the object-level information management for design processes. Based on this environment, architecture of object-oriented information management system is developed, with consideration for future integration with knowledge support applications. A shown in Fig.8, the system is made up of three layers, which are design modeling tool, design repository, and information manager. The design-modeling tool provides users with a means to create, delete, and re-configurate design process models, information models, and knowledge models. The design repository includes a database and a knowledge base. The database stores process, information, knowledge, and link objects used by generic information management function. The knowledge base would store the knowledge models and knowledge processing rules and algorithms, which required by the knowledge support application. The information manager has two basic functions, the generic information management function and the knowledge support function. The latter is actually a stand-alone component of information management system. The generic information management

function would provide mechanism that handle generic "information" which actually includes all the objects stored in database. The knowledge support function would focus on the methodologies and mechanisms of how to provide useful knowledge for design processes. The mechanisms of generic information management function would capture and make information widely available and enable users to ready access to them along the design process, while the mechanisms of knowledge support function would provide users with particular knowledge that address the current specific needs.

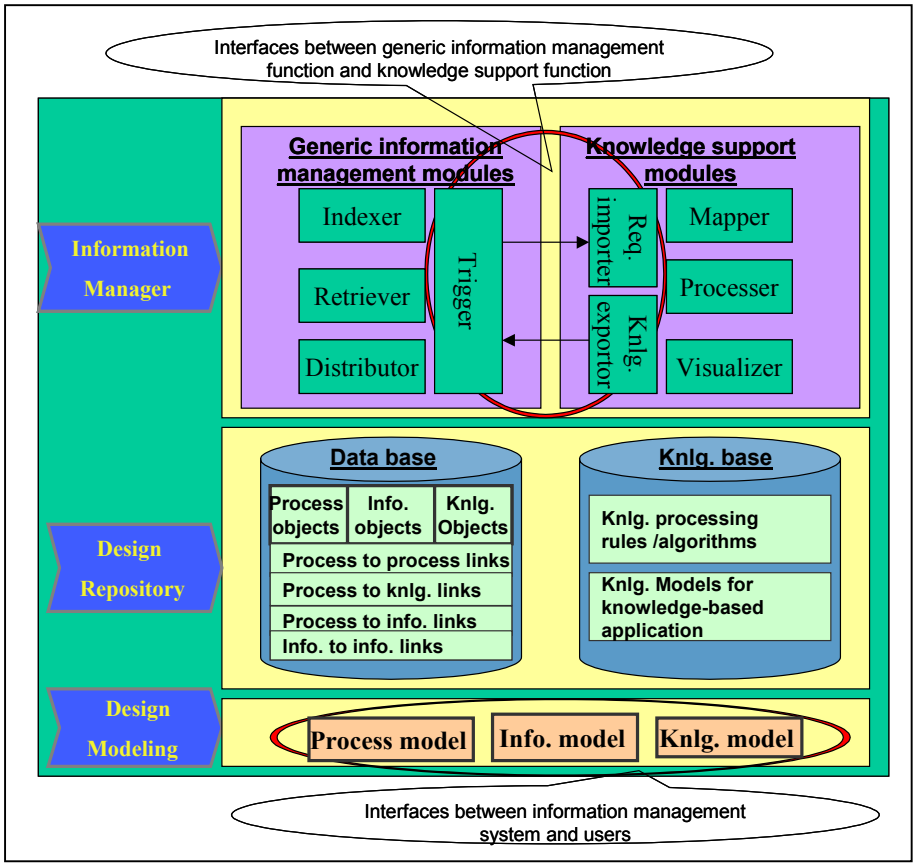

Figure 8. Object-level information management system architecture

To realize the generic information management function, the system would need to address the information indexing, retrieving, and distributing mechanisms as any other information system does. Similarly, to achieve the knowledge support function, the system would provide knowledge mapping, processing, and visualizing mechanisms that requested by most of knowledge-based applications. To integrate these two functions, the information management function side further needs a trigger module to trig the knowledge support function to work. The knowledge support function side would need two additional modules, requirement importer and knowledge exporter. The requirement importer would import specific knowledge support requirements from the knowledge objects that linked to design processes. The knowledge exporter would transform the final result of knowledge support function to information management module and infuse the attributes of knowledge objects as instance value for each design instance. These three additional modules would act as the interfaces to integrate the generic information management function and knowledge support function.

4. Conclusion

In this paper, a design modeling methodology was proposed specifically for the development of an object-level information management system for complex and dynamic design processes. By decoupling process from product and information from knowledge, the design process model could be more generic and facilitate the fast re-configuration of information management system in design. The information model provide meta-model for information objects (process deliverables) identified by process model. This meta-model classifies the properties of process deliverable into different perspective, which facilitate the information to be queried and distributed more selectively. The knowledge model provides the knowledge interface for the integration between information management and knowledge support application. And finally, architecture of an object-level information management system was presented based on the environment built up with these modeling methods.

Reference:

[1] Simon Szykman, Ram D. Sriram, William C. Regli, "The Role of Knowledge in Next-generation Product Development Systems", Journal of Computing and Information Science in Engineering, March 2001, Vol 1/3, ASME.

[2] Mary Lou Maher, M. Bala Balachandran, Dong Mei Zhang, "Case-based Reasoning in Design", Lawrence Erlbaum Associates Publishers, 1995.

[3] Steven D. Eppinger, Daniel E. Whitney, Robert P. Smith and David A. Gebala, "A Model-Based Method for Organizing Tasks in Product Development", Research in Engineering Design, Vol.6, 1994, pp.1-13.

[4] Robert P.Smith, Steven D. Eppinger, "Identifying Controlling Features of Engineering Design Iteration", Management Science, Vol.43, No.3, 1997.

[5] Lu, S.C.-Y. and Cai, J., "A Generic Collaborative Engineering Design Process Model for Conflict Management", Department of Aerospace and Mechanical Engineering, University of Southern California, Impact Working Paper, No. WP00-03, 2000.

[6] Andres F Melo, P John Clarkson, "A State-Action Model for Design Process Planning", Cambridge Engineering Design Center, June 1999.

[7] Pahl, G and Beitz. P., "Engineering Design", Sringer-Verlag, Berlin, 1984.

[8] D Brady, N P Juster, "A Computerized Tool To Create Concept Variants From Function Structures", AI System For Conceptual Design, Proceedings of the 1995 Lancaster, International Workshop on Engineering Design, Springer.

[9] Melogy Stokes, "Managing Engineering Knowledge, MOKA : Methodology for Knowledge Based Engineering Applications", Professional Engineering Publishing Limited, 2001.

[10] Henson, B., Juster, N., and Pennington, A., "Towards an Integrated Representation of Function, Behavior and Form", Computer Aided Conceptual Design, Proceedings of the 1994 Lancaster International Workshop on Engineering Design, Lancaster University EDC, Lancaster, pp. 95-111.

If you need more information please contact: Hua Sun and W. F. Lu*, Singapore Institute of Manufacturing Technology, 71 Nanyang Drive, Singapore 638075; Phone: (65) 67938297; Email: hsun@SIMTech.a-star.edu.sg, wflu@SIMTech.a-star.edu.sg.

* Singapore-MIT Alliance Fellow.