THE USE OF KNOWLEDGE BASED COMPONENTS FOR AUTOMATED TASK SUPPORT IN THE PROCESS INDUSTRY

Arnold, J. A. and Teicholz, P.

ABSTRACT: This in-process research project investigates a knowledge based component' information model that is capable of supporting distributed software services for design and engineering in the process industry. The test case component information model for a control valve is based on the results of previous research that defines the information requirements [Arnold, Teicholz 1996]. It integrates an explicit description of product data (form, function, and behavior} and an engineering process to offer partial automation for the task of control valve selection in an intelligent design environment. The research also investigates the business and technical issues related to the deployment of knowledge based component information libraries on the Internet. This work seeks to understand how such software services can be realized, how they can integrate with work process, and how they would benefit A/E/C business practice.

KEYWORDS: productt, process model, integration, knowledge based component, component library

1. INTRODUCTION

Timely and accurate exchange of information about the components that go into process plants is an important business problem. Vendor knowledge, contingent upon the quality of engineering requirements provided by the customer, often determines final component specification for facility design and engineering, and the procedures for component installation and maintenance. The specification process for components requires extensive information exchange between product vendors, plant designers and engineers. This information, in turn, requires considerable effort to access, interpret, validate and transform so that the relevant data can be used for project documentation and the necessary fabrication, installation, operations and maintenance activities. Failures in the timely and accurate delivery of component information causes delays and uncertainty in the performance of tasks which, in turn, result in poor component selection, incorrect specifications, time delays and re-work, increased change orders, etc. These coordination problems increase the transaction costs between component vendors and customers, and between the other parties who need the information in the course of the facility life-cycle. Thus, improved methods for accessing, storing and using component information can have a significant impact on design and construction cost, time and quality.

The following points highlight the business issues:

- Vendor knowledge, contingent on the customer relationship, often drives final component specification, installation and maintenance procedures;
- Considerable non value-added effort is required to access, interpret, validate, and transform vendor information into project documentation;
- The vendor bears considerable marketing cost to distribute information and maintain customer relationships;
- The added effort increases the transaction cost of data exchange;

'A component is defined as a simple (single part) or complex (assembled part) item that is normally manufactured and sold by a vendor and becomes incorporated into the facility. The term "component model" is used to differentiate this information model from "product model" which is used to model the facility itself. The terms "part" and "component" are used interchangeably, though we prefer the latter because it includes complex assemblies of simple parts. In a process plant, the components consist of such items as pipes. valves, pumps, fittings, etc. Clearly, there is a strong overlap in the technologies used for modeling components and products. and there are times when this differentiation is not appropriate. The business issues for vendors (who sell components) and E/C contractors (who design and build plants) are, however, significantly different. This, and the need to identify the special information modeling issues for components, requires that a distinction be drawn between component and product models.

- Current information products and services do not improve business process nor provide added value beyond decreased search time. In particular, they do not add value to the owner of the project for use of the facility.

1.1 Research Goals

This research attempts to understand the problem described above for the process industry. The project:

- Studies the business issues of information exchange and the life-cycle information requirements for the components that are installed in process plant facilities;
- Identifies engineering tasks and coordination requirements that an information model should support for one component type, the control valve;
- Investigates standards development for component and product information models;
- Implements a test case that explores integration of a component information model with a task based process model to provide decision support for a control valve.

2. WORK TO DATE

2.1 Information Requirements Study

A facility life cycle information requirements study focuses on a representative component type, the control valve. This case study identifies the product information for control valves that is needed during the various processes of the facility life cycle.

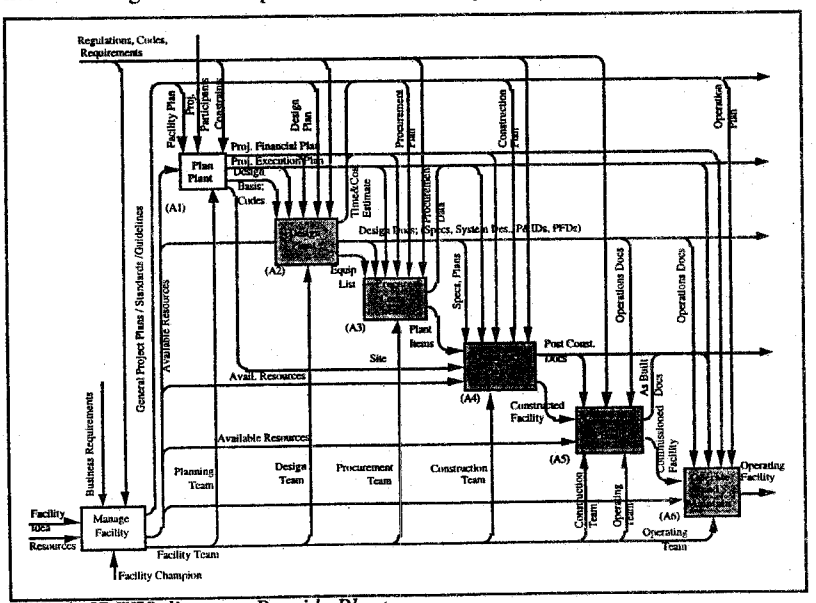

FIG. 1: IDEF0 diagram, Provide Plant.

These phases include the key requirements for business and technical processes during design and engineering, equipment procurement and construction, plant commissioning, and operations and maintenance. The overall process description of the facility life-cycle shown above is decomposed to lower level tasks that focus on a functional description of control valve sizing, material selection, procurement, installation and maintenance. The study lists 150 life cycle information requirements for control valves grouped amongst 15 major tasks. This information is categorized according to life cycle

tasks and a set of terms that are useful for generally describing the product information requirements for components. Each category follows:

Properties; Assigned, Design Requirement, Design Standard.

Behaviors; Computed, Measured.

Administration; Information that supports the flow (coordination and control) of information and resources for transactions (procurement, etc.).

Task Coordination; Design Contingency, Change Notification, Task responsibility.

Other Factors; Time contingency and resource availability are additional critical factors in the determination of coordination requirements.

2.2 Analysis

2.2.1 Task Based Views

Partition of the requirements by task suggests one of many possible views of the information. No individual project participant needs or uses all the information that is listed. Most participants work with a subset that is relevant to the task at hand.

From the decision support perspective, a component model should be capable of representing multiple views of underlying information for various work requirements. The work requirements vary depending upon the purposes for use and the time at which it is required or available. Similarly, the product model in which the component is referenced should also be capable of representing these views and of providing a design context for the characteristics of the component model that are functionally dependent on the design.

Some argue that support for engineering reasoning and decision support should be provided within software applications that access and manipulate an underlying information model. It is a significant research issue whether it is possible to develop a domain independent conceptual framework that supports view based knowledge and reasoning for engineering processes.

2.2.2 Dynamic schema generation

The detailed requirements identified in the case study represent the union of what was learned through the interviews, interaction with experts working on STEP information models, and a literature search. Despite these efforts that resulted in the identification of 150 separate information items, the enumeration of the requirements is assuredly incomplete. For example, most of the information is collected from E/C firms, information integrators, facility owners, and valve vendor representatives who participate in technical marketing. The information requirements study does not include detailed requirements from the perspective of the component manufacturer for component production and test procedures.

It has already been noted that each participant involved in information exchange about components has a specific perspective or view of the requirements dependent upon many factors, including the usage requirements and time at which the information is needed. Furthermore, no one participant has all the knowledge and expertise to develop a complete model. These factors point to the need for an information model that supports design schema evolution and enables the dynamic addition of information throughout the facility life cycle. This point has been expounded by [Eastman 1995] in development of EDM2. Whereas EDM2 focuses on an underlying data model that supports, possibly unique, design compositions, this research focuses on the representation of building products that can be used in compositions.

One should not lose sight of the relationship between design schema evolution and its importance for supporting business process. To achieve business goals, individuals and organizations access, manipulate, and transform information for a business purpose. This often results in new, derived information about a product. This study shows, to some degree, the broad range of information requirements for one component across the facility life cycle. An information model that has a general mechanism for schema evolution can support a complete representation of engineering design objects and

their interrelationships [Phan, Howard 1993] without attempting, a priori, to fix and define a domain for which the information requirements inevitably change.

2.2.3 Limits of current State of the Art in Product Modeling

The study also investigates current efforts to create standard information models for the process industries; STandard for the Exchange of Product data (STEP) Application Protocols 221 and 227, and ISO 13584 Part Libraries (PLIB). They satisfy some of the requirements identified by the study and highlighted by the test case while other requirements remain out of scope for current efforts.

The STEP framework does not yet provide a description of component behavior which is necessary to support automated reasoning. Nor does it explicitly incorporate a product description (properties and behavior) with related engineering process. Such a model would provide support for automated component evaluation and analysis software services that add business value, For a thorough description of how each modeling method satisfies the requirements, see [Arnold, 1996].

2.3 Test Case

As mentioned, the information requirements study identifies several engineering processes that relate to control valves during the facility life cycle. Some of the processes are engineering tasks that require the application of engineering knowledge and reasoning to compute performance characteristics for a control valve. This reasoning is representative of design and engineering knowledge work, and particularly of tasks that involve analysis and evaluation of designed elements. Such knowledge translates project requirements and constraints into the designed product model. Automation of knowledge work is common in many software applications (e.g., finite element analysis programs). However, information models developed for standard product data exchange do not yet explicitly support this important task related dimension of design and engineering.

Modeling behavior in addition to the functional and structural properties of components makes it possible to provide analysis and evaluation software services for component selection and usage. These software services should formally describe and integrate a description of component data and related engineering process. In so doing, they would offer value-added functionality through the automation of existing process. Product, process information model integration can result in increased reuse of data, better component selection, improved decisions for operations and maintenance, and improved coordination between project participants.

The test case explores this concept. It demonstrates an integrated product and process description in support of the control valve sizing task by providing control valve sizing analysis and evaluation in an intelligent engineering application. It also demonstrates the usefulness of a software service that interacts directly with the design model.

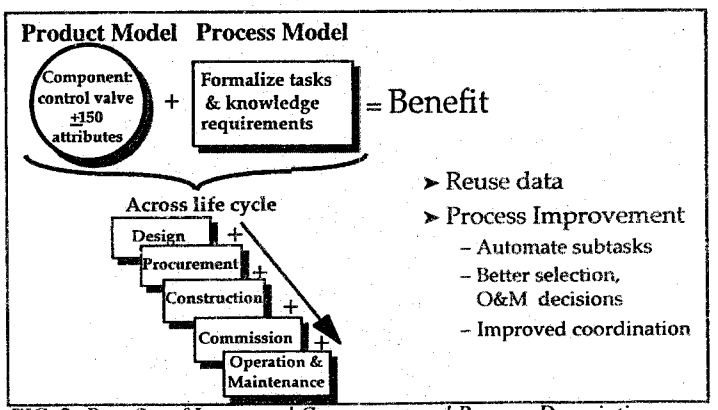

FIG. 2: Benefits of Integrated Component and Process Description

2.3.1 Product Description

The test case is an information model for a component and a piping sub-system. It includes a behavioral representation that supports one engineering task, preliminary valve sizing and selection. It uses the Symbolic Modeling Extension (SME) [Clayton, Kunz, Fischer, Teicholz 1994] modeling methodology that builds upon Form, Function, Behavior (FFB) [Kunz 1988], [Clayton, Fischer, Kunz, Fruchter 1995] concepts to model the control valve component and its placement in a piping sub-system of a P&ID product model.

2.3.2 Process Description

Through reification[2] of FFB, symbolic models enable the explicit representation of the internal states of a system. [Gero 1995] elaborates a causal relationship between FFB elements **(he uses the terms structure and function- check this)** and incorporates them into a design process framework in which the interaction of form and function constraints determine behavior. The comparison of predicted behaviors with intended functions can necessitate 'reformulation' of form and function characteristics to satisfy behavior constraints. Through the activities of designing[3] the designer reconciles form, function and behavior to provide an acceptable solution for a design problem. Gero maps the form, function, behavior relationships to the following activities of designing:

- formulation;
- synthesis;
- analysis;
- evaluation;
- reformulation;
- production;
- design description.

These activities define the steps for the application of design knowledge and reasoning to the definition and refinement of a product model. They represent the value added service provided by design and engineering professionals, yet the knowledge content of these activities is not fully or explicitly represented in current information models. The full value of the design and engineering service is not captured at each stage of the design project nor is it passed to the facility owner.

The creative element of design formulation and synthesis is beyond the capability of current computing technology, however model based reasoning for the analysis and evaluation phase of designing is possible using symbolic models that explicitly represent the constraint relationships between object form, function and behavior [Luth, Krawinkler, Law 1991], [Clayton, Fischer, Kunz, Fruchter 1995].

2.3.3 Views

The content of a model should be parsimonious to the extent that it contains only the information necessary to fulfill a modeling purpose.

In relational databases, views limit the visible fields of a relation. [Law 1992] explores object definition and management for topological views of structural elements based on underlying relations. View development for the design context and designing activities in an A/E/C product model is challenging due to the complexity and diversity of information perspectives for the various disciplines. Yet, product model views are necessary to provide a context for model based analysis and evaluation.

2.3.4 Integration of Product Description, Process and View

[Clayton, Fruchter, Krawinkler, Kunz, Teicholz 1993] propose a method, called Interpretation, for the real time assignment of views to a product model. They emphasize designing as a visual process in which practitioners informally draw ideas before semantically interpreting them into symbols that are subject to analysis and evaluation. Relating this perspective to Gero's work, interpretation further

---

[2]Reification is defined in Webster as the process of regarding an abstract concept as a material thing.

[3]Gero uses the word 'designing' to signify the process of creating an artifact and 'design' to signify the description of the designed artifact.

clarifies the difference between 'formulation' and 'analysis and evaluation'. To model the work process, they argue for interactive interpretation as another activity of designing. [Clayton, Fischer, Kunz, Fruchter 1995] reifies the interpretation of design shapes (walls, doors, etc.) through Interpretation Objects. An Interpretation Object relates design forms with a functional issue (e.g., cost estimation, energy, egress and spatial requirements analysis) in the SME prototype. Interpretation objects then may be 'critiqued' according to interpretation specific constraints. A Critique generates predicted behaviors for a functional issue that is associated with a set of forms. It then evaluates the predicted behaviors in terms of the functional requirements and furnishes an analysis to the user. Each Interpretation is a view of a product model[4]. Clayton calls the run-time association of Interpretations to a CAD representation a "Virtual Product Model".

SME demonstrates dynamic annotation of CAD graphics with interpretation (view based) product model information. Equally important, through the integration of *critique* objects into the product model, it shows that model based analysis and evaluation, and thus automated engineering process, is possible for architectural design.

2.3.5 Implementation

Detailed Description

The valve sizing task is properly described as the analysis and evaluation phase for designing a control function of a process stream. The diagram below, adapted from [Kunz 1996] depicts a form, function, behavior view of the valve sizing task in light of Gero's design process framework and Clayton's interpretation objects. In step 1 the user manually 'interprets' a sub-system of the P&ID diagram. Through Interpretation, a symbolic representation of the component model links with a product model for which it provides the control function. The design decision support of the component model consists of its ability to predict and assess the behavior of a control condition for a given piping sub-system and set of functional requirements defined by a process stream. Step 2 shows the 'feature constraints for Behavior' and the 'Requirement constraints for behavior' relations between the relevant line and steam form and function objects and the predicted

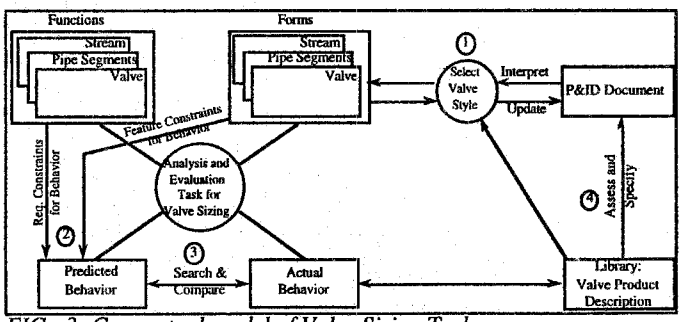

FIG. 3: Conceptual model of Valve Sizing Task

valve behavior object. The predicted behavior represents the required performance condition, given the form and function constraints that are provided. Step 3 occurs during the 'Critique' phase. The predicted valve behaviors are computed and compared with the actual behavior of available valve products that are available in a component library. In step 4 the user assesses the products that match or exceed the requirements and selects a valve for the design.

To perform this service, the model represents the valve's thermodynamic properties, the thermodynamic characteristics of the piping system in which it is installed and the process engineering knowledge for

---

[4] Interestingly, the application of set operations (intersetion, union, difference) to a collection of Interpretations leads to insights about shared information between them. Clayton's SME implementation begins to explore this through user interface tools that allow the user to perform set operations on available interpretations.

calculating a correct valve size. To support preliminary control valve sizing computation and analysis, the component information model incorporates [ISA 75.01 1991] standard sizing algorithms.

The test case performs this task for incompressible fluids. It does not represent a geometric valve description or its spatial configuration within a process plant. This modeling work has been performed within STEP [ISO CD 10303-227 1995]. In addition, the model does not yet represent valve trim, bonnet and actuator materials or other accessories. These elements are not necessary to perform preliminary sizing. However, as per the information requirements identified in the requirements study, it would be necessary to represent these items if the reasoning functionality were extended to support valve material selection, specification, operations, and maintenance activities.

As mentioned above, many commercial software applications provide design decision support. Several programs assist users to specify control valves [Fisher 1993]. The difference is that the test case is model based. It reifies the valve behavior within the design model and captures behavioral state conditions of the valve under different functional load conditions. Conceptually, any project participant could query the product model to perform this analysis. Process and behavior information is incorporated into the product model and can be reused easily. Conventional software applications do not represent this information explicitly.

Application Description

Figure 4 shows the valve sizing *Interpretation Inspector* and the P&ID CAD representation. Each piping sub-system item in the list box named *Items* is associated with an element in the CAD drawing. When the user **Inspects** an item, an *Item* dialogue box opens (not shown) in which the user assigns properties and functional requirements. Once the annotation is complete, the user presses the **Size** button to invoke the *Interpretation Critique*. The user then views the *Critique* results by pressing the **Results** button. From the *Results* dialogue box, the user can select an appropriate valve for the product model. The implementation uses the AutoCAD R12, Sun workstation platform. The symbolic models are developed using the Intellicorp Kappa object oriented programming environment for Sun OS, UNIX.

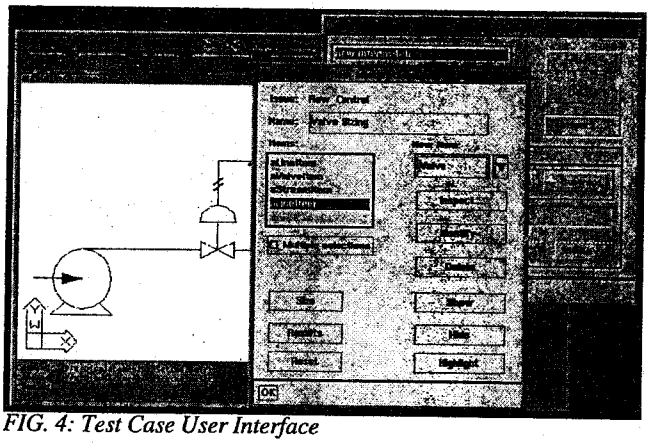

FIG. 4: Test Case User Interface

Test Case Information Model

This section describes the test case information model structure using Conceptual Dependency Diagrams (CDD) that are generated from the Kappa object model. A CDD is itself an object model generated by an application written for Kappa. The CDD application extracts class objects, class object relationships, and methods from a source Kappa application (e.g., the test case) to show the class sub-class specialization hierarchy, the relations between objects that are not in the same graph, and the methods that are associated with each class. While less expressive than general diagramming techniques (EXPRESS-G, OMT, etc.), the CDD is useful because one can generate it directly from the Kappa development environment. CDDs do not indicate class instanciation. Figure 5 indicates how to read a CDD. Refer

to Appendix D for an object model figure that shows the run time instances that are generated from a valve sizing Interpretation. Each run time instance is part of an Interpretation (explained below), and can be identified by its name suffix.

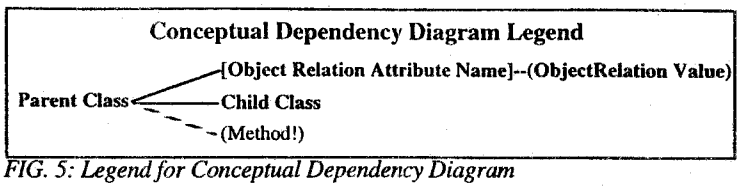

FIG. 5: Legend for Conceptual Dependency Diagram

The suffix name is derived from the CAD drawing file with which the Interpretation is associated. For the diagram in Appendix D, the suffix is <@*testdwg*>. Figure 6 (below) shows the object named **Pipe_Sys_Item** (PSI). Subclasses of PSI offer logical representations of the plant items that are necessary to perform valve sizing for a given control valve placement. Child classes of the PSI include **Line** (pipe segment), **Stream** (process media), and **Valve**. When a valve sizing Interpretation is created, instances of the Pipe, Line, and Valve objects are generated. They link to graphical elements in the CAD system. The object relation attributes: PSI_Function, PSI_Form, and PSI_Behavior link the logical PSI objects to appropriate Form, Function, Behavior objects for each logical object type. The FFB objects are instanciated during the Interpretation process. The methods bound to the PSI objects perform object management for these relations. The VPPM (Virtual Process Plant Model) object contains the methods for creating the link between the symbolic model in Kappa and the CAD representation. Figure 6 also shows the Valve Critique object. An instance of this object is generated each time the user invokes an Interpretation Critique (by clicking on the **Size** button in the Interpretation Inspector). Figure 7 (see forward) shows the Form class

objects. Instances of **Line_Form** represent different nominal pipe diameters, classification ratings and materials. These instances are predefined and are not generated at run-time. The association of the FFB object to the PSI logical object is transparent for the user. In the case of the **Line_Form** object, it occurs when the user selects a line diameter in the *Line Item Inspector*.A logical object **Line** can be associated with any one **Line_Form** instance. The methods associated with the **Line_Form** class furnish the piping geometry factor (Fp) when it is necessary to account for pipe swages. The sub classes of **Valve_Form** correspond to the different valve types that are distinguished by shape (geometry), not by function. Note that Valve Form objects provide constraints for valve behaviors through the object relation attribute *FeatureConstraintsForBehaviors*. These object relations reify Gero's assertions about the causal relationship between form, function, and behavior summarized previously. Much like instances of Line_Form, Valve_Form instances are predefined and cannot be modified by the user. The methods bound to the **Valve_Form** object are 'Get' functions for the following attributes: valve style modifier (Fd), liquid pressure recovery factor (Fl),and Laminar flow factor (Fs).Figure 3-9 lists the Function and Behavior objects that relate to the Line, Stream, and valve objects. Instances of the **Line** and **Stream** function classes are generated at run timewhen the user enters the functional criteria for each functional case. For valve sizing there are normally three functional cases to consider:

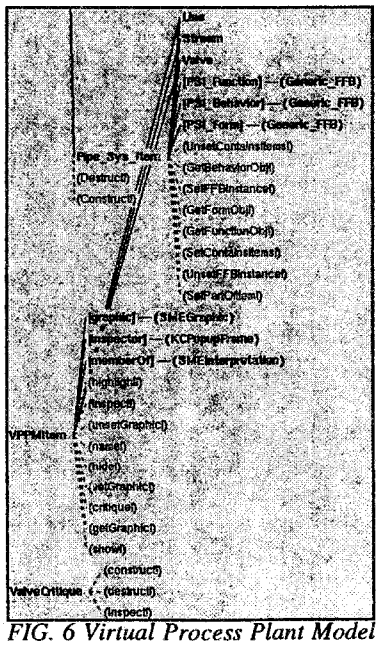

FIG. 6 Virtual Process Plant Model Item (VPPM) & Pipe System Item Objects

minimum, maximum and upset flow conditions. However, N functional stream instances can be generated for a given Interpretation. A Behavior instance is generated for each functional case when the user invokes an Interpretation Critique. Each function object relates to a corresponding behavior object through the object relation attribute *ReqConstraintsForBehavior*. The inverse object relation attribute is *DerivesFromReqConstraints*. The behavior methods bound to **Valve_Behavior** are of primary interest for valve sizing. They include:

- Reynolds number (Re!);
- Reynolds number factor (Fr!);
- Predict coefficient of flow (PredictCv!);
- Determine whether a valve will Cavitate (DpKc!).

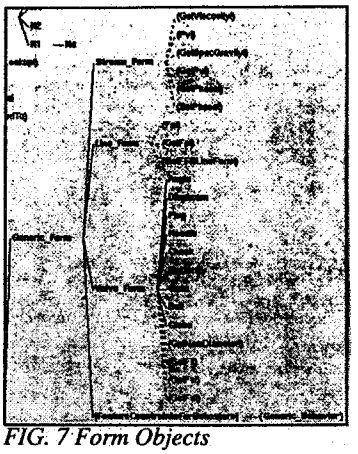

FIG. 7 Form Objects

The method PredictCv! formalizes an engineering process. It is called from the Interpretation object method Critique! (see forward). PredictCv! encapsulates the engineering reasoning (logic) for valve sizing analysis by determining the proper way to compute the sizing algorithm based upon the functional conditions (laminar, transitional, or turbulent flow).

Figure 9 shows the Interpretation object called **Flow_Control_Interpretation**. This object furnishes the method Critique! which invokes valve behavior object instanciaton and sets the object relations described previously. From within the Critique! method, a newly instanced valve behavior object invokes the PredictCv! method described previously to perform the valve sizing analysis. Then, the Critique! method searches for a valve library object type (e.g., ButterflyType) that matches the users choice for the valve style (valve form object) and the valve type Search! method is invoked. Search! compares the predicted valve behaviors determined from invocation of PredictCv! with the actual behaviors of valve type instances that are predefined in the library.

Figure 10 shows the simple library representation developed for the test case. It shows the logical object **ValveLib** with logical object sub classes for the various valve types. These valve types have instances that refer to the same FFB classes as the product model described previously. Run time generated instances of **Valve_Behavior** to compute candidate valve performance are differentiated from predefined, library supplied behaviors through a naming convention. Predefined valve behavior instance names have the prefix *<Actual>*. Note that the object relation attributes have the same name as those utilized for the PSI objects. In a more robust implementation, the FFB objects and properties may be separated for the library and product models. For the test case, this implementation was expedient.

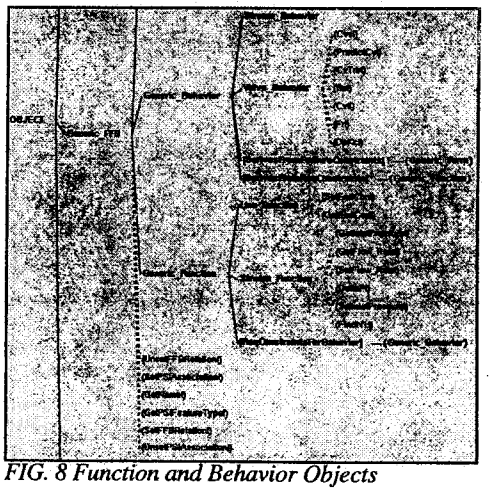

FIG. 8 Function and Behavior Objects

2.3.6 Information Model Summary

The information model for the test case reifies Form, Function, Behavior objects for each pipingsystem component type and explicitly associates them to logical objects that represent a relevant portion of a piping circuit. Generation of the instances for Form objects are predefined from standard component descriptions. The user can select size or style options for each component type through the user interface. Function objects may be generated (e.g., stream functional cases) at run time. There are two types of behavior instance: predefined or *actual* behaviors that describe tested

component performance, and *predicted* behaviors that are generated at run time during valve sizing analysis and evaluation. The valve sizing equations (ISA coefficient of flow formulas) and the comparison of actual to predicted behavior according to engineering analysis logic constitutes the 'reasoning' of the system. This behavioral reasoning is encapsulated within methods. The SME approach enables two degrees of product model definition flexibility. First, an Interpretation can be developed for any evaluation and analysis reasoning that should be performed for a design. The dynamic annotation of Interpretation objects to a CAD representation allows the user to add semantic schemas as needed. Second, the separation and explicit association of FFB objects to a logical product model enables further flexibility in the information model structure.

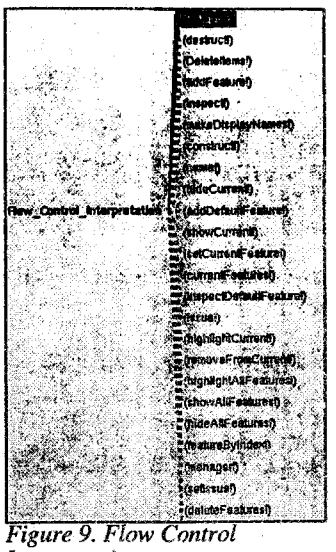

Figure 9. Flow Control Interpretation.

The dynamic linkage of FFB types to a logical representation enables a range of functional and behavioral representations to be expressed according to constraints that are specific to form, function, and behavior properties.

It is understood that whether model semantics have an a priori, static definition or are assigned at run time, the issue of standard representation for semantics persists. Without agreement on the semantics of content representation, information sharing is not possible. Through the explicit representation of behavior and process in addition to form and function, SME/FFB adds two additional dimensions to the modeling challenge and thereby significantly increases modeling complexity. The fact that there is currently no conceptual classification of Interpretation (process description) or FFB objects indicates the relative immaturity of the approach and its current lack of generality and extensibility. This is a matter for further research.

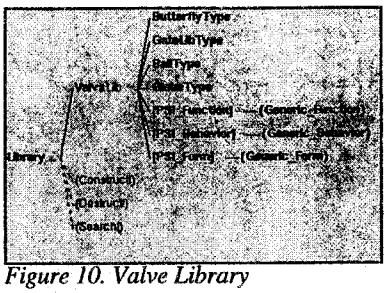

Figure 10. Valve Library

Nonetheless, SME demonstrates the integrated representation of object form, function, behavior, and task based process. The test case implementation reifies the constraint relationships that are formalized in design process theory and indicates that they may be useful concepts for the development of model based evaluation and analysis software services. Such concepts and services are not available yet in the standard product data representation initiatives.

Concept Validation

To date, validation efforts for the test case have been informal. Nonetheless, the process model was described and the implementation was demonstrated to representatives of an Engineering/Construction firm who had participated in the information requirements study. The response was positive. The formalization of process corresponded well to how engineers work. The concept of placing an analysis tool on the engineer's desktop that integrates with the design model was consistent with the business improvement goals for the company.

The engineering reasoning and valve sizing output of the test case was validated by comparing it with the output of a commercial valve sizing program for the same input parameters. This test was successful.

3. CONCLUSION

The project basis of the A/E/C industry leads to enterprise organization and information technology (IT) support systems that are fragmented. This problem manifests through communication problems, errors, delays, and increased cost for project participants. The first requirement for correcting IT support for this business problem lies in the standardization of product data representation. This is the challenge of

ISO STEP, and more recently the Industry Alliance for Interoperability (IAI). These initiatives provide a data structure and content foundation for software vendors to develop integrated applications that improve communication between enterprises.

Information integration through product data standardization solves only part of the business problem. It is also possible to formally describe processes, the things that individuals and organizations do with product data to achieve project goals. The integration of product and process description in an executable program can lead to software services that offer significant benefit through task automation, increased data and knowledge reuse, better decision making and improved coordination.

This research has attempted to understand the information content requirements for developing models that provide such services.

3.1 Challenges

3.1.1 .Component Information Model Design

Review of the information model structure shows that the implementation is relatively ad hoc and lacks generality for the definition of the Form, Function and Behavior objects. Note that the purpose of the test case was to demonstrate the potential for an information model that integrates a component (properties and behaviors) and process description, not to define a canonical information model structure. Nonetheless, the object hierarchy for FFB objects jumps directly from "Generic" objects to sub-classes specialized directly for control valves. Without a set of common FFB primitives from which more complex objects can be derived, it will be impossible to share information between models and systems [Phan, Howard 1993]. A conceptual hierarchy for form, function, behavior objects is a matter for further investigation and research. In addition, there is no conceptual hierarchy to characterize the process description that is reified within the Interpretation object. Doing this would require extensive work to classify processes and decompose them into to sharable primitives as well.

Several information models were reviewed prior to developing the prototype. The concept of the pipe_system_Iitem object to represent the logical description of the pipe, stream, and valve entities was adapted from ISO AP 227. Further parallelism with other models was not attempted. It was decided that this would be too time consuming without direct contact and interaction with the persons who developed the other models. Even attribute assignment for the various objects was subjective. It was based upon best judgment at the time the implementation was developed. While the rationale for the model structure and properties was subjective, the goal was to be internally consistent with the definition of structure and assignment of attributes and behaviors.

The ad hoc nature of the test case and the general variance of modeling methods (e.g., AP221 and AP227), indicates the need for the development of "good" modeling principles and, vitally important, a set of metrics for the evaluation of information models.

3.1.2 Conceptual View Framework

The SME work raises an important issue concerning the dynamic assignment of Interpretation objects to a product model. It is plausible to consider a library of Interpretation objects from which a user annotates the entities in a product model. However, predefinition of Interpretation information for the product description would improve the efficiency of design model generation. From the perspective of a component library, it would be necessary to develop a conceptual view framework to support multiple component representations for various design and engineering tasks. The PLIB conceptual model offers a compelling basis for this framework. It offers a mechanism for homogenous characterization of component attributes at the class level (General and functional classes). Also, it enables users to organize and access information according to domain specific practice (Semantic dictionary). Last it supports multiple representation (functional views) of components so that information access can be organized according to usage requirements. Other work that formalizes process description within the A/E/C industry includes [Levitt, Hayes-Roth 1989] who develop the OARPLAN construction planner. In addition, the computational organizational modeling literature contains references to research that investigates conceptual frameworks for process description [Malone, Crowston, Lee, Pentland 1993],

[Lee, Yost, and PIF Working Group 1994]. This research discovered no work directly related to formalization of engineering and business tasks for the process industries.

3.1.3 Directions for Future research

In the future it will be possible to offer interoperable software services for nearly any hardware platform over distributed networks. Design and engineering professionals will be able to tap information and task knowledge from a 'virtual' desktop comprised of resources available within the company and from external enterprises. They will be able to incorporate this formalized knowledge into project work directly and thereby leverage it for greater productivity.

To build towards this vision, plans for future work include further investigation into the integration of product and process description for component models. To validate and compare the findings for control valves, a second case study will be performed to either 1) understand the process requirements for another task related to control valves, or 2) understand the information requirements for design analysis and evaluation of another component type. From these findings, the research will investigate a task-based view framework for a conceptual data model that improves support for product and process model integration.

We also hope to understand the potential business impact of software services based on such models by developing a new test case that demonstrates component object transactions using the World Wide Web. The test case will show access, retrieval, and use of component information objects for design and procurement between a component supplier and a user. The objects that are accessed from the supplier will integrate seamlessly into the user's CAD system or a project database management system. The test case will be implemented in Java to enable the creation of component objects that support the behavior and process descriptions developed in this report.

Understanding the software management issues for interoperable, distributed objects will be another goal of future research. In this effort, it hoped that our work, which focuses on information content, representation, and process functionality, can be joined with CIFE work by others that investigates the network middle ware requirements for interoperation of legacy software applications that are developed using different languages and hardware platforms [Kunz, Law, Howie 1996].

4. REFERENCES

Arnold, J. A, Teicholz, P.(1996) A Study of the Life Cycle Requirements for an Information Model of the Components that are Incorporated in Process Facilities, CIFE Technical Report #107, Center for Integrated Facilities Engineering (CIFE), Stanford University, Stanford, CA. 93405-4020 USA

Clayton, M. J., Fischer, M., Kunz, J., and Fruchter, R.(1995). Behavior Follows Form Follows Function: A Theory of Design Evaluation, Second ASCE Congress on Computing in Civil Engineering, Atlanta GA. USA

Clayton, M. J., Kunz, J. C., Fischer, M. A., Teicholz, P.: 1994 First drawings, Then Semantics. *in Reconnecting, ACADIA94.* (eds.) Anton Harfmann and Michael Fraser, Association for Computer Aided Design in Architecture, 1994. pp. 13-26 .

Clayton, M. J., Fruchter, R., Krawinkler, H., Kunz, J. and Teicholz, P.: 1993, Propose-Interpret-Critique-Explain: A communication cycle for collaborative conceptual building design, in Gero, J. S. and Maher, M. L. (eds.), *AI in Collaborative Design*, AAAI, Menlo Park, CA, pp. 287-292

Eastman, C. M.: 1995, Managing Integrity in Design Information Flows *in Computer Aided Design*, IPC Press

Fisher Sizing Program, Revision 1.41. Copyright © 1989-1993. Fisher Controls, International, Marshalltown, Iowa

Gero, J.: 1995, The Role of Function-Behavior-Structure Models *in* ASCE 2nd Congress on Computing in Civil Engineering, June 5-7, 1995

Howie, C., Kunz, J. C., Law, K. (1996). A Model-Based Approach to Software and Data Interoperability for Process Plant Applications, CIFE Seed Proposal INT960, Center for Integrated Facilities Engineering (CIFE), Stanford University, Stanford, CA. 93405-4020 USA

ISA-S75.01, Standards and Recommended Practices for Instrumentation and Control, 11th edition ©1991. Instrument Society of America, 67 Alexander Dr. P.O. Box 12277, Research Triangle Park, North Carolina 27709

ISO CD 10303-227: 1994, International Standards Organization, Technical Committee ISO 184, Industrial automation systems and integration, Subcommittee SC 4, Industrial data and global manufacturing programming languages - ISO 10303 Product data representation and exchange - Part 227: Plant Spatial Configuration

Kunz, J. C., Law, K. H., Howie, C.: 1996, A Model-Based Approach to Software and Data Interoperability for Process Plant Applications, CIFE Seed Proposal, INT9601. Center for Integrated Facilities Engineering, Stanford University, Stanford, CA 94305-4020

Kunz, J. C.: 1996, Rapid Conceptual Design Evaluation Using a Virtual Product Model. Center for Integrated Facilities Engineering Technical Report No. 105, 1992

Kunz, J. C.: 1988, Model Based Reasoning in CIM. Intellicorp 1988, Mountain View, California 94040

Law, K. H.: 1992, Managing Design Information in a Shareable Relational Framework. CIFE Technical Report No. 60, 1992. Stanford University, Stanford, CA 94305-4020

Lee, J., Yost, G., and the PIF Working Group: 1994, The PIF Process Interchange Format and Framework, Version 1.0, December 22, 1994. MIT Center for Coordination Science Working Paper 180

Levitt, R. E., Hayes-Roth, B.: 1989, OARPLAN: Generating Project Plans in a Blackboard System by Reasoning about Object, Action and Resources, CIFE Technical Report No. 2, 1989. Stanford University, Stanford, CA 94305-4020

Luth, G. P., Krawinkler, H., Law, K., H.: 1991, Representation and Reasoning for Integrated Structural Design. CIFE Technical Report No. 55, June, 1991. Stanford University, Stanford, CA 94305-4020

Malone, T., Crowston, K., Lee, J., Pentland, B.: 1993, Tools for inventing organizations: Toward a handbook of organizational processes. Working Paper #141 Center for Coordination Science, Massachusetts Institute of Technology, May 1993. In Proceedings of the 2nd IEEE Workshop on Enabling Technologies Infrastructure for Collaborative Enterprises, Morgantown, WV, April 20-22, 1993.

Phan, D. H., Howard, C. H.: 1993, The Primitive-Composite (P-C) Approach - A Methodology for Developing Sharable Object-Oriented Data Representations For Facility Engineering Integration. CIFE Technical Report #85A, August 1993. Stanford University.