Proceedings of the TMCE 2004, April 13-17, 2004, Lausanne, Switzerland, Horváth & Xirouchakis (eds.)

A FLEXIBLE ELECTRO-MECHANICAL DESIGN INFORMATION SYSTEM

Anton H Basson
Department of Mechanical Engineering
Stellenbosch University
South Africa
ahb@sun.ac.za

G Maarten Bonnema
Department of Engineering Technology
University of Twente
Netherlands
g.m.bonnema@utwente.nl

Yang Liu
Department of Mechanical Engineering
Stellenbosch University
South Africa
yliu@sun.ac.za

ABSTRACT

Design information systems are of increasing importance due to globalization of product development, increasing complexity of products (in terms of product variants and range of technologies employed) and increasing emphasis on improving product development efficiency (shortening times, reducing cost, increasing performance). Current design information systems are typically applicable in a limited range of contexts. To be flexible, i.e. applicable over a wide range of contexts, design information systems have to accommodate the diversity of the design process in terms of scope, level of abstraction, level of detail, conflicting use of terminology, wide ranging procedures, multi-disciplinary character, information types, etc. A flexible design information system is proposed that is based on the use of ontologies and conceptual graphs, with the result that the main data structures and, to a lesser extent, user interfaces are not affected by the introduction of new terminology, procedures and tools.

KEYWORDS

Design information, collaboration, distributed design, ontology, conceptual graph

1. INTRODUCTION

Design information systems are receiving much attention currently in industry and the research community, as outlined in the review by Wang et al. (2002). Much research has been directed at supporting collaboration and distributed team scenario's (e.g. Toye et al., 1993, Jin and Lu, 1998, Huang and Mak, 1999, Szykman et al., 2000, Peña-Mora et al., 2000, Abselsalam and Bao, 2000, Schueller and Basson, 2001, Giannini et al., 2002 and Wang et al., 2002). Other research looked at integration of the information systems over various phases of product realisation (Lutters, 2001).

Most design information systems reflect a specific engineering design context and cannot easily be adapted to other contexts (e.g. Veeramani and Mehendale, 1999 and McKay et al., 2003). The use of object-orientated approaches (e.g. Szykman et al., 2000, Abdelsalam and Bao, 2000) introduce a measure of flexibility, but may require software development or database structure changes for handling new objects.

This paper proposes a flexible design information system derived from previous research aimed at integrating information used in all product realisation phases (Lutters, 2001). In the system presented here integration is not an objective, but the focus is on flexibility, i.e. being adaptable to a wide range of design contexts with minimal effort.

This paper considers information related to the engineering design of electro-mechanical systems. Some of the proposed concepts are applicable to a wider scope of engineering design, but not necessarily to artistic design, industrial design (e.g. aesthetics-related aspects of consumer products) or architectural design.

For the purpose of this paper, design information is taken to include the requirements derived from the whole product life cycle (which obviously includes clients' requirements), a detailed product definition, and all the information the design team used to move from the requirements to the definition.

An aspect of some design information systems not considered here, is the inclusion of artificial intelligence in the design information system (in contrast to e.g. Stacey et al., 2002). The approach taken here is that the system only aims to present appropriate information to the designers, who then apply their knowledge.

Another distinction between the information system considered here and some other commercially available and research systems, is that the system should not only manage documents, such as reports, but should ideally handle design information as it is entered by the designer. This ideal is currently not feasible for some information in the later design phases, e.g. the information in CAD and FEA files. However, the design information system considered here should capture as much information as is practically possible and assist the designer in creating coherent documentation.

The following sections in the paper show that the inherent diversity in the design process implies that design information systems that aim to be generally applicable, have to be flexible, i.e. highly adaptable. A strategy to achieve such flexibility is described thereafter.

2. THE OBJECTIVES OF DESIGN INFORMATION SYSTEMS

Before considering how information management systems should handle diversity in engineering design, this section considers what the design information in the system will be used for.

Significant reductions in the time a product is on the market, globalisation of commerce combined with increasing product variety, etc. have lead to renewed interest in making product realisation (which includes product design) more efficient. Efficiency in this context can be expressed in terms of product performance, life cycle cost and time to market in relation to the design cost and duration. Since information exchange is a central activity in design (as in all product realisation phases), more efficient management of this information should contribute to making product realisation more efficient.

In practical terms, some of the objectives for design information systems include enabling distributed design, helping in backtracking from dead ends (when a design selection later proves to be infeasible), tracing the effects of changes in design decisions, archiving design information for re-use in future designs, documenting design information for project and quality management purposes, etc.

Within the design activity, much information handling occurs, e.g. exchanging information between design team members to clarify requirements, generate concept solutions, stimulate each other's thoughts, etc. Design information systems must therefore handle data from various sources within the design process, but since product design is influenced by and has an impact on the whole product life cycle, design information systems must also interface with information systems supporting other life cycle phases. An example of this is the close interaction between product design and manufacturing process planning. Design information may also form part of an enterprise information system, which introduces further links and interfaces to contend with.

In many companies much of design information is currently not captured in structured or systematic ways. The final product definition is nearly always captured in a detailed and structured form since this is the main output from the design activity and is an input for downstream product realisation processes. Many tools are therefore available to manage the latter type of information (such as in current leading CAD systems). However, as the cost and performance of the final product is largely determined in the early phases of the design process, the capture and management of information from these earlier phases is necessary to achieve the objectives for design information systems listed above. The diversity of design information in the early phases presents challenges to design information systems, and is addressed in this paper.

3. DESIGN PROCESS DIVERSITY

The diversity of design processes has significant implications for design information systems aimed at handling a wide range of design scenarios.

3.1. Design Scope

One of the main sources of diversity in design processes is the range of scopes that can be addressed: engineering design activities can range

from designing the proverbial mousetrap, to designing the means of getting a human being to Mars and back. Certain activities will be common to all scopes, e.g. synthesis, analysis and evaluation on subsections of the design task (Blanchard and Fabrycky, 1998). Even specification development can be cast in this form (using the terminology of Ullman, 2003): engineering requirements must be synthesised, target values must be attributed to the requirements through analysis and evaluation must show whether the user requirements are completely reflected in the engineering requirements.

Another view of design scope relates to the level of abstraction of the design artefact. For example, creation of product concepts in "industrial design" terms, technology selection in systems engineering, selection of physical principles that can realise a particular function, sizing of components through strength calculations, selection of tolerances, and producing rapid prototypes can be considered as design activities arranged in approximately reducing levels of abstraction.

Level of "detail" is closely related to this, since more abstract levels tend to apply to less detailed information. Ullman's (2003) distinction between selection design (selecting catalogue parts), configuration or layout design, and parametric design conveys similar differences in level of detail of the design activity. The different ways of "zooming" in and out can be represented by the three dimensional space, introduced by Krumhauer (1974), shown in Figure 1. Concreteness and Complexity have been dealt with above. The third axis, Realisation, shows the progress from idea to product.

Since design activities should involve inputs from the whole product life cycle (particularly manufacturing, support, use, disposal), ideally in a concurrent engineering approach, the scope of the design activities may extend even beyond generating a detailed product definition. An example of this is the interaction with the "design" of the manufacturing processes required to produce the product or logistic support system required to keep the product operating at the required level of reliability. In, for instance, the car industry, a large part of the total design effort is spent on these activities.

Although it is unlikely that one design information system will ever be suitable for all scopes, design activities related to even marginally complex products would include elements of subdivision and different levels of abstraction. This is because complexity is often handled in the design process by subdivision: a big unmanageable task is subdivided into smaller manageable tasks. Another view of this is that complexity is handled by working at higher levels of abstraction, e.g. using functional descriptions to make early design decisions.

Section 4 considers the implications for design information systems of the range of design scopes.

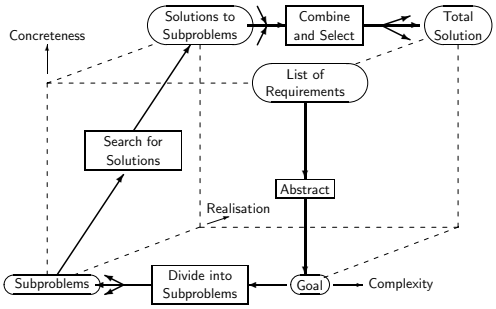

Figure 1 The space of the conceptual design phase (Krumhauer, 1974)

3.2. Models, Methods and Procedures

Engineering design is a complex process due to strong coupling and interdependence between the many sub-processes, the wide spectrum of expertise and technology that may be involved, the non-uniqueness of the "best" process and the "best" end-result, and the iterative nature of design. The complexity is reflected in the bewildering array of design procedures and methods that have been proposed or are being used. Each procedure can be related to a particular design model. Cross (1994) gives a useful summary of many methods. In this section, the distinctions between methods, procedures and models and the variety of each of these will not be considered in detail. Only a few aspects of the variety of procedures will be discussed here to illustrate the diversity in engineering design that flexible design information systems have to contend with.

One distinction between different design processes is the extent to which a top-down approach (e.g.

"function before form") is promoted, as opposed to a "cut-and-try" approach (choosing a concept or building a prototype early and then gradually improving it). This aspect is related to Cross's (1994) distinction between *prescriptive* and *descriptive* design models. Prescriptive models try to persuade designers to follow a more-or-less algorithmic, systematic procedure as a better means of working. The descriptive models, on the other hand, simply describe the typical sequence of activities, generally reflecting a solution-focussed approach where an initial solution is formulated early in the design process and then gradually improved.

An aspect closely related to the previous, which is common between all design procedures, is the role of feedback and inevitable revisions. Prescriptive procedures attempt to minimise the revisions (to save time and cost) through structuring the process, but no process claims to eliminate revisions. Suggestions for minimizing their cost are, for instance, to put more effort in the early phases of design and to analyze the possible concepts more thoroughly (e.g. Ullman, 2003, and Bonnema, 2003).

Another aspect of significant diversity is in the overall structure assigned to the design process. To illustrate this, a few well-known references can be compared. Ullman (2003) identifies the following main steps in the design process: Identify need, plan for the design process, develop engineering specifications, develop concepts, and develop product. These steps employ various methods such as functional decomposition. Suh (1990) has a much simpler, but more abstract structure. He considers the design to be a mapping between the "functional requirements" in the functional domain and the "design parameters" of the physical domain, which occurs through the proper selection of design parameters that satisfy functional requirements. Blanchard and Fabrycky (1998) consider the design process for large engineering systems to comprise conceptual design, preliminary design, detail design and development. Each of these parts is divided into five to eight sub-elements, with a common thread being synthesis, analysis and evaluation applied at increasing levels of detail.

The differences between the design process structures of the various approaches are in some respects related to differences in scope (level of abstraction and size of the design project), but in many respects they just show the variety of design processes structures in general use. The commonalities are usually on a more abstract level, and are therefore not obvious to the design practitioners.

The confusion in the terminology used in design process literature is another indication of the diversity in the design process, with the same term used for different aspects in different contexts. To illustrate this, the use of the term "function" can be considered:

Ullman (2003) defines a "function" to be the logical flow of energy (including static forces), material, or information between objects or the change of state of an object caused by one or more of the flows, and "functional modelling" as a decomposition of the design problem in terms of the flow of energy, material and information. Suh (1990) makes extensive use of the concept of "functional requirements", since it is a core feature of his design process, as pointed out above. In his approach, the functional requirements form a statement of the design's objectives in terms of specific requirements. This is in contrast to Ullman's (2003) approach where a distinction between requirements and functions is maintained. In Suh's approach "constraints" are distinct from "requirements", but there is no such distinction in Ullman's approach. Szykman et al. (2000) even makes a distinction between "function" and "behavior". Akiyama (1991) divides functions into "external" and "internal" functions, in which the latter more or less coincides with Suh's and Ullman's use of functions, but the former is closer to Blanchard and Fabrycky's (1998) use of the term, which is outlined below.

Two disparate contexts both using the notion of functions are systems engineering and industrial design. Blanchard and Fabrycky (1998) define "functional analysis", in the context of systems engineering, as the process of translating system requirements into detailed design criteria, along with the identification of specific resource requirements at the subsystem level and below. As with Suh's approach, functions and requirements are not clearly distinguishable in Blanchard and Fabrycky's approach, although they define a function as a "specific or discrete action that is necessary to achieve a given objective". Eekels and Poelman (1995), working in industrial design, even use the notion of emotional functions and semantic functions, as shown in Figure 2.

The preceding paragraphs have outlined some aspects of diversity in design procedures. No single design process can be proven to be "the best" in all applications, as the project and the design team influence what procedures are appropriate in a particular circumstance (Bender et al., 2001). The implications of this for design information systems are considered in section 4.

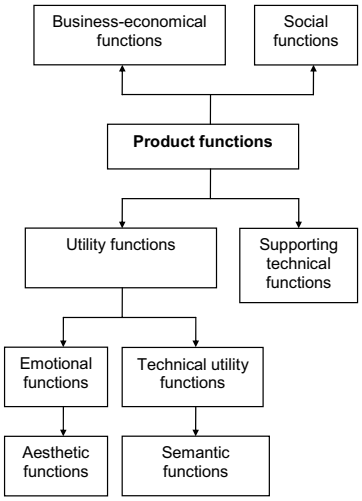

Figure 2 A view of functions in industrial design (Eekels and Poelman, 1995)

3.3. Other Aspects

Some other aspects of design process diversity that have bearing on design information systems will be outlined in this section.

Most engineering design activities are multi-disciplinary. Expertise and experience from various knowledge areas normally have to be integrated to produce a successful design, and the disciplines involved may change from project to project. Each discipline may have some peculiarities in its design process and design terminology.

Engineering design uses a wide variety of tools. Some are integral parts of the design process, such as axiomatic design (Suh, 1990), TRIZ (Altshuller, 1999), and QFD (QFD Institute, 2003). Others are used to analyse various models (e.g. FEM structural models, dynamics, CAD geometry, etc).

The formats of information employed in engineering design are diverse. Some examples are text, calculations with associated sketches, graphic designer sketches and drawings, CAD models (including drawings, tolerancing information, etc), data sheets from catalogues, analysis models and results (structural analysis, fluid mechanics, dynamics), etc. Even though they may contain equivalent information, many information types or formats associated with design tools are incompatible, which leads to barriers to information transfer. The result is that the same information is handled and stored in different formats, possibly in different locations, which violates basic principles of database design since it allows room for inconsistency in the information set as a whole.

Human beings play a central role in design. The variety in human aspects, such as language, culture, educational background, professional experience, personal preferences, etc. also adds to the diversity of engineering design.

4. IMPLICATIONS FOR DESIGN INFORMATION SYSTEMS

4.1. Flexible Design Information Systems

The requirements associated with flexibility are considered here, before considering the more general implications in the next section.

Given the need for and roles of design information systems and the diversity inherent in the engineering design process, the authors contend that research should be directed at developing information systems that handle the diversity to the greatest extent possible. The objective should be for the information system to adapt to the application context, particularly the design team, the peculiarities of the particular project and the design process chosen by the team for that project. Information systems that do not succeed in doing this will be limited to application in specific contexts, will find resistance from the users and will impede the design process.

The following main requirements for a flexible or agile design information system can be deduced from the previous section.

The system must be scalable and support subdivision of the design tasks: It must be able to handle a range of design scopes, levels of

abstraction or detail, and life cycle phases. The common thread should be repeated synthesis, analysis, evaluation and decision, on progressively greater levels of detail. Provision must be made for subdividing the design problem into subtasks, which can be handled concurrently by distinct teams if need be. To ensure that the overall objectives are met, each subtask must depart from a "baseline" common between all subtasks and the evaluation activity should determine to what extent the solution synthesised for each subtask meets the requirements set in the baseline. In addition to reference to the baseline, some subtasks will have to interact with other subtasks. The design information system should therefore support the management of the baselines, as well as the interfaces between a subtask, the baseline and other subtasks. It is important to note that a subtask does not necessarily coincide with a subsystem or module. Some aspects of a product can only be achieved by a system-wide approach (e.g. cost of a product, “overlay performance” of a wafer-stepper and safety of an airplane).

Design information systems that aim to be generally applicable cannot enforce a prescriptive design model or any particular overall structure. It must allow the team to use the design procedure and terminology most appropriate to the particular context, whether top-down, bottom-up, or a combination (INCOSE, 2000: "one of the Systems Engineers' first jobs on a project is to establish nomenclature and terminology"). It must facilitate frequent revisions of previous design decisions, but in a controlled and traceable fashion. For example, parts of a design that have been "frozen" must be protected from inadvertent revision.

The information required and produced by a wide variety of design tools should be handled by the information system, even though the information will be of a wide variety of formats. In multi-disciplinary teams, this requirement becomes even more important.

4.2. Design Information Systems in General

Irrespective of the objectives of the design information system, the source of the data is in the design process. Since the information captured by the system will normally be recorded by the design team members, the system must aim to place the minimum additional workload on them and not impede their creative processes. The system should therefore provide the designers with an intuitive user interface, which is easy to relate to their preferred way of working.

Since a design information system is not a goal unto itself, it is not only influenced by the sources of the information, but also the objectives for a particular system. Some requirements that are introduced by the information system's goal, are outlined in the following paragraphs.

Some requirements common to many users of the information already captured by a design information system are facilities to extract information selectively, to improve and increase the information by recording comments on the information and to communicate these comments to the appropriate team members.

If a design information system is aimed at making the information exchange during the design process more effective, the ease-of-use requirements listed above are particularly important. It is also imperative that the system responds virtually immediately to user request otherwise creative processes will be disrupted. All design team members, even if they are in different locations, should further have concurrent access over data networks, like the Internet, to the information relevant to their activities. The information system must ensure consistency of the information (e.g. it must prevent that two users edit the same information at the same time or that users inadvertently use outdated information).

Since performance, cost and time are equally important in engineering design, design information systems should assist design teams in managing information about all three these aspects.

The objective of a design information system will influence the extent of the information that must be captured, e.g. systems aimed at design reuse (also called a design repository by Szykman et al., 2000) would require more extensive documentation of the context of the design task, and would have to provide tools for searching for candidates for reuse.

Many design information systems must also work with legacy data. One of the side effects of globalisation of product realisation is that design information systems from various companies that have been amalgamated or are collaborating, have to be made accessible through a design information system.

5. PROPOSED APPROACH

As pointed out in the introduction, many design information systems have been or are being developed. In the authors' view, however, they do not meet the requirements outlined above for flexible systems. A new approach is called for, which is outlined in this section.

5.1. Overview

The key notions for the flexible design information system presented here are the use of an ontology approach, coupled with a database implementation structured as a conceptual graph. In the present context, Corazzon's (2003) definition for ontology is appropriate: "Ontology is the theory of objects and their ties. The unfolding of ontology provides criteria for distinguishing various types of objects (concrete and abstract, existent and non-existent, real and ideal, independent and dependent) and their ties (relations, dependences and predication)." Another formulation particularly relevant in the context of information exchange, by Toye et al. (1993), is that a common ontology is "a shared vocabulary of terms and definitions".

The objects under consideration in design information systems can be classified as design information elements, and their ties as relationships between the elements. The classification in terms of elements and relations is found in conceptual graphs (Sowa, undated). Another aspect of conceptual graphs of importance for flexible design information systems is that elements only interact with each other via relations, and relations only interact with elements. The notion of attributes, which is sometimes used with conceptual graphs, is not used here since it makes the database structure dependent on the number of attributes associated with each element type. The information that would have been contained in attributes is rather handled by introducing further element types and relation types.

Such use of ontology and conceptual graphs was identified by Lutters (2001) when seeking ways of integrating information systems over all product realisation phases. The diversity of information in that situation is even greater than in design alone. The main difference between Lutter's proposals and what is proposed here, is that the flexible design information systems will not try to get all designers to agree a priori on the ontology and then restrict them to this ontology, but will allow designers to adapt the ontology used in any particular project even though this will impede data exchange outside the design context.

The differences between the approach proposed here and object orientated design information systems (such as that by Szykman et al., 2000) should be considered. Object orientated information systems also use an abstraction of design information (in the form of objects), but such objects are substantially more complex than conceptual graphs, even though any object can, in turn be described in terms of elements and relations. Working with objects more complex than merely elements and relations certainly hold intuitive advantages for the user, but any changes or additions to the objects handled in the design information system, may require significant database changes and therefore software code changes. Further, object orientation loses some of its appeal if the data cannot be structured hierarchically (e.g. in terms of parent and child objects), such as when a particular information element has relations with elements that would naturally be in unrelated objects.

Figure 3 shows the information flows for the main components of the proposed flexible design information system. The components are described in the following sections.

5.2. Ontology Database and Project Database

The Ontology Database and the Project Database are the core of the design information system. They are illustrated by the simple examples in Tables 1, 2, 3 and 4.

The Ontology Database (Tables 1 and 2) contains a description of the available types of elements and relations. Each type has a unique ID, which is referred to by the Project Database. The "Description" field explains the role of the element or relation for the particular design team.

The Relation Types table (Table 2) also shows the types of elements that may be associated with each relation (ID_{ET1}, ID_{ET2}). The element types in the Relation Types table are redundant information as the same information can be derived from the Project Database. Even though this permits potential inconsistency in the databases, this risk is accepted because the availability of the element type information in the Relation Types table makes the creation of generic user interface code (described in

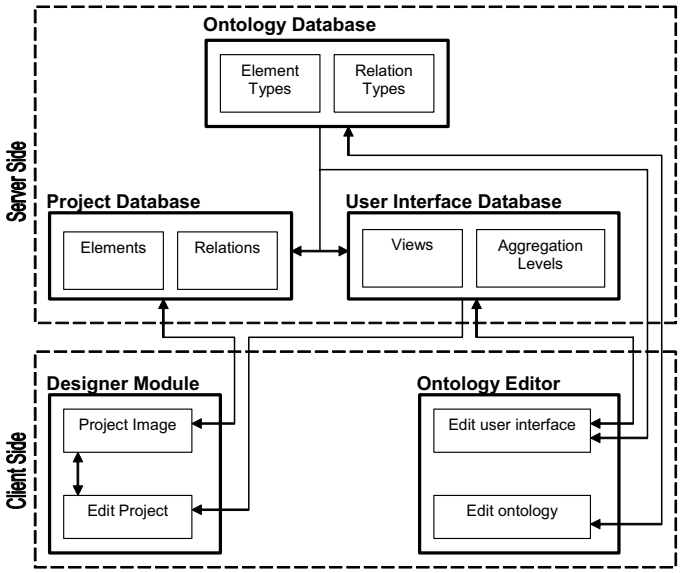

Figure 3 Main Information Flows in the Flexible Design Information System

section 5.3) possible and makes the meaning of each relation clearer to the user. The risk of inconsistency can be reduced by taking appropriate measures in the Ontology Editor and by not allowing changes to the type of an element in the Project Database if there are any relations associated with that element.

The Project Database contains the actual design information associated with one or more projects. The element table (Table 3) contains cross-references to the element types in the ontology (ID_{ET}), while the relation table (Table 4) cross-references the type of relation in the ontology (ID_{RT}), as well as the two elements associated with the relation (ID_{E1}, ID_{E2}).

The flexibility of this approach becomes evident when the designers want to handle new types of information or when new associations between existing types must be recorded. In such cases, the Ontology Database can be expanded and the new information can be added to the Project Database. The database structures and the data already entered into the databases are not affected by expanding the Ontology Database (the impact of such changes on the user interface is considered in a following section).

To allow designers the freedom to brainstorm, without having to worry about classifying information in terms of the ontology at the same time, "unclassified" element and relationship types can be provided. The information system can automatically record the design team member who entered an unclassified element and later present the designer with a "to-do" list of elements that still have to be classified (Szykman et al., 2000). The flexibility of the proposed approach extends to not fixing a specific sequence in which data has to be entered, except that a relation normally has to be added after at least one of the elements to which it refers. "Incomplete" relations can also be added to a designer's "to-do" list.

In the discussion above, it is implied that only one project's data is contained in the Project Database. This is however only done to simplify the discussion, since there is no inherent restriction on the number of projects handled in the same information system. The project's identity is merely another element type and adding a new relation type to associate an element with a project is straightforward. An element can even be associated with more than one project, if the user so wishes.

Table 1 Example Element Type Table (Ontology Database)

ID_{ET}*	Name	Description
0	Unclassified	Rough ideas just jotted down; to be classified later
1	Customer requirement	Customer's requests in own words
2	Design specification	Measurable design specification
3	Target value	Target value associated with an engineering requirement
4	Finalization status	Suggestion, Challenged, Accepted, Rejected, Frozen, Deleted
5	Design team member	User ID of design team

* cross-referenced in Table 2 and in Table 3

Table 2 Example Relation Type Table (Ontology Database)

ID_{RT}+	Name	Description	ID_{ET1}*	ID_{ET2}*
0	Unclas-sified	Used for rough ideas; team member must classify later	0	5
1	Derived from	Relates a customer requirement to a specification	1	2
2	Has property	Relates target value to an engineering requirement	2	3
3	Has status	Relates status flag to element	all	4

\+ cross-referenced in Table 4
* cross-reference to Table 1

5.3. Ontology Editor, User Interface Database and Designer Modules

Due to the simplicity of the Ontology Database, an editor to maintain the database should be simple. The selection of element or relation types that must be added to the ontology is the very challenging part, but is not addressed in this paper. Case studies using a trial implementation of a flexible design information system currently under development will use published design procedures and observations of design teams in practice, to compile one or more ontologies.

One of the major challenges in producing a flexible design information system is to provide user interfaces that can easily be adapted as the ontology grows. Even though the extent to which this can be realized remains an open question, it is conceivable that a "rapid interface development" tool can be developed similar to the user interface of Borland's C++ Builder, Delphi and Kylix RAD environment.

Table 3 Example Element Table (Project Database)

ID_{E}**	ID_{ET}*	Value
1	1	Must not be heavy
2	4	Frozen
3	2	Total mass
4	4	Accepted
5	3	10 kg

** cross-referenced in Table 4
* cross-reference to Table 1

Table 4 Example Relation Table (Project Database)

ID_{R}	ID_{RT}+	ID_{E1}**	ID_{E2}**
1	3	1	2
2	1	1	3
3	3	3	4
4	2	3	5

\+ cross-reference to Table 2
** cross-reference to Table 3

Another approach to user interface development is to develop a range of generic user interface applets that each can handle a certain type of relation. For example, a user interface to handle a one-to-many relation type can build a tree-type display of elements, independently of the specific type of element or relation. Such a user interface can be used, for example, for a hierarchical functional decomposition, but also when user requirements are classified into categories such as life cycle phases. Another generic user interface could, for example display any many-to-many relationship with a value associated with each relationship. This user interface can be used to display discrete parts of a QFD house of quality, e.g. the relationship between the customer requirements and the engineering requirements, with an indication of the strength of each relation. The same basic user interface can also handle a comparison matrix of requirements and concepts.

Under the assumption that most of the user interface requirements for a flexible design information system can be satisfied by the above type of generic user interfaces, Figure 3 includes a table of "Views" in the User Interface Database. Each view would comprise the type of generic user interface and a set of relations that is displayed in that user interface. If a new element type or relation type is added to the ontology, the "Edit user interface" function in the "Ontology Editor" can be used to create a new "View" with which the new ontology entries can be viewed and edited.

One of the outflows of the generic user interface approach suggested above, where the user interface

is tied to a relation, is that each new element type added to the ontology should have at least one relationship type that is associated with it. This requirement is, however, quite reasonable, since design information not related to any other design information is hard to conceive.

The requirement that the design information system allows users to select the level of detail of the information displayed, is handled by provision of "Aggregation Levels" (Lutters, 2001) in the user interface. Aggregation levels can be controlled by the type of elements or relations displayed or the number of levels displayed.

Once the user interface database has been populated, the designer can view and edit project information. The user interface settings from that database will contain the appropriate ontology information that the "Designer Module" in Figure 3 needs to select the information to be displayed. The system outlined in Figure 3 assumes that users will communicate with the databases via the Internet, in a client-server configuration. Since this type of communication can be too slow to allow effective interactive work, the "Designer Module" would have to maintain an image of the particular project's information on the client side. This image would typically be maintained by one software thread, while another thread is used for interacting with the user. The "Project Image" thread would send the changes due to the user's edits through to the Project Database, and would periodically ask for updates on the project information from the database, in case other designers have added or edited the project information.

Since the database structures are not affected by ontology changes, users are not affected by such changes, other than accessing the new user interface options if they need it. Extensions to the ontology or new user interface options can therefore be implemented without any interruption of design team members, while a menu in the "Designer Module" would make users aware of the available views in the User Interface Database.

5.4. Evaluation Against Requirements

The preceding sections outline the main aspects of the proposed flexible design information system. This section considers more specifically how they satisfy the requirements given in section 4.

Scalability and support for subdivision of design tasks are important requirements. In the proposed design information system, they can be reduced to extensions of the ontology, which are easily handled. The notions of "Views" and "Aggregation Levels" in the User Interface Database (Figure 3) provide, in principle, the means to select the information that a particular designer needs to work with. The data structures further assume no specific procedure, thus allowing the design team the maximum freedom in the procedure they use.

The use of the ontology approach also allows design teams the maximum freedom to adapt their terminology to the particular design context. Even a facility for team members to attach comments to someone else's information (redlining) can be handled by appropriate provision in the ontology. If there is a significant requirement to exchange the design information with other information systems while the design information is developed, more care will be required in formulating the ontology to minimise future ontology changes, since ontology changes will propagate outside the design information system.

A limitation of the proposed design information system is that its data structures are inherently text based. Simple graphical information, such as sketches generated during conceptual design of mechanisms, can also be handled as information elements. Complex non-textual information (e.g. FEM models, CAD files, etc. where complex relationships exist within the information) cannot be handled completely within the data structures of the proposed system. Element types that contain links to other data structures, documents, files or URL's will have to be used for non-textual information, similarly to what present design information systems do. The impact of handling different applications' data on the user can be reduced by making use of services provided by current computer operating systems, e.g. OLE (Object Linking and Embedding), for the apparent seamless integration of different application's data in one user interface. This will, however, result in restricting the design information system to a particular operating system, thereby reducing its flexibility.

Another limitation in the proposed flexible design information system is in its ability to incorporate design tools into the user interface and Project Database. Tools such as QFD (QFD Institute, 2003), that are primarily a specific procedure in

which specific design information is generated, can comfortably be handled. Tools such as TRIZ (Altshuller, 1999), where design information has to be generated, the appropriate information identified from reference sources, and that information then extracted, will require links to external information or that the user enter the extracted information into the design information system. Since all the reference sources used in the design process (e.g. part catalogues) will not be known a priori, any design information system will have to contend with this limitation.

The importance of rapid response during interactive design sessions was pointed out in section 4.2. The "Project Image" in the "Designer Module" in Figure 3 was introduced to ensure quick response while working. Nonetheless, the ontology approach does require substantially more searching in the data structures since the relationship information is not "hard coded" into the data structures. The searching overhead can be reduced somewhat by departing from the purely conceptual graph based database structure and including some relationship information in the elements table. This will only be advisable where all elements have a relation with a specific type of element, for example creation date, finalisation status and aggregation level. These inclusions should, however, be done after careful consideration whether the loss in generality is warranted by the performance gains.

Aspects such as concurrent access, ensuring data consistency and distributed access will be readily realisable within the system outlined in Figure 3.

The requirement that cost and time aspects be managed, along with performance, can be handled by including these aspects as elements in the ontology. It is expected that in this way, budgets (for cost, weight, power consumption etc.) are easily devised and handled as integral aspects of the system design. The cost, time and other budget elements, and their associated relations, in the Project Database will provide all the information required, but specific algorithms for computation, display and editing will be required.

6. CONCLUSIONS

Design information systems are of increasing importance due to globalisation of product development, increasing complexity of product (in terms of product variants and range of technologies employed) and increasing emphasis on improving product development efficiency (shortening times, reducing cost, increasing performance). Although much research is being done to develop design information systems, the current systems are only applicable in a limited range of contexts.

To be flexible, i.e. applicable over a wide range of contexts, design information systems have to acknowledge the diversity of the design process. Design process diversity includes aspects of scope, level of abstraction, level of detail, conflicting use of terminology, wide ranging procedures, multi-disciplinary character, information types, etc.

A design information system is proposed that is based on the use of ontologies and conceptual graphs. This approach enables a great degree of flexibility with the result that main data structures and, to a lesser extent, user interfaces are not affected by the introduction of new terminology, procedures and tools. Design teams can therefore adapt the system's ontology to the preferences of the teams and to what is appropriate to the particular design problem.

The approach can naturally handle all information that can be expressed in terms of information elements and their interrelationships. Textual and simple graphical information can therefore be handled within the design information system, but only links to complex non-textual information, such as CAD files and analysis data files, and external information sources, such as catalogues, will be kept in the system.

A trial implementation of the proposed system and case studies to evaluate its capabilities and limitations are underway. Particular aspects that need to be clarified are whether "generic" user interface code can be developed (so that ontology extensions do not necessarily require software development) and whether the additional database searching cost is acceptable.

ACKNOWLEDGEMENTS

The financial support of the National Research Foundation (NRF grant GUN 2034084) and the Stellenbosch University Research Committee is gratefully acknowledged.

REFERENCES

Abdelsalam, H.M., and Bao, H.P., (2000), "Towards a Collaborative Engineering-Computation Environment: An Application of an Object-Orientated

Database to Project Management", paper DETC2000/CIE-14611, Proceedings of ASME 2000 Design Engineering Technical Conferences, Baltimore, USA.

Akiyama, K., (1991), "Function Analysis: Systematic Improvement of Quality Performance", Productivity Press, Cambridge.

Altshuller, G., (1999), "The Innovation Algorithm", Technical Innovation Center Inc.

Bender, B., Bender, B., and Blessing, L.T.M., (2001), "How Missions Determine the Characteristics of Product Development Methodologies", in Design Management – Process and Information Issues, Proceedings of the International Conference on Engineering Design, ICED01, Glasgow, UK, pp. 313-320.

Blanchard, B.S., and Fabrycky, W.J., (1998), "Systems Engineering and Analysis", Prentice Hall.

Bonnema, G.M., and Van Houten, F.J.A.M., (2003), "Conceptual Design in a High-Tech Environment", Proceedings of 2003 CIRP Design Seminar, Grenoble France.

Corazzon, R., (2000), "Descriptive and Formal Ontology", http://www.formalontology.it/, [20 August 2003].

Cross, N., (1994), "Engineering Design Methods: Strategies for Product Design", 2nd ed., Wiley.

Eekels, J., and Poelman, W.A., (1995), "Industriële Productontwikkeling, Deel 2: Methodologie", Lemma, Utrecht.

Giannini, F., Monti, M., Biondi, D., Bonfatti, F., and Monari, P.D., (2002), "A Modelling Tool for the Management of Product Data in a Co-design Environment". Computer Aided Design, Vol. 34, pp. 1093-1073.

Huang, G.Q., and Mak, K.L., (1999), "Web-based Collaborative Conceptual Design", Journal of Engineering Design, Vol. 10, No. 2, pp. 183-194.

INCOSE, (2000), "Systems Engineering Handbook", version 2, International Council on Systems Engineering.

Jin, Y. and Lu, S.C-Y., (1998), "An Agent-Supported Approach to Collaborative Design", Annals of the CIRP, Vol. 47, No. 1, pp. 107-110.

Krumhauer, P., (1974), "Rechnerunterstützung für die Konzeptphase der Konstruktion", PhD Dissertation, TU Berlin.

Lutters, E., (2001), "Manufacturing integration based on information management", PhD Dissertation, University of Twente, Netherlands.

McKay, A., Thomas, P.J., Ramirez, B., Leech, S.J., de Pennington, A. and Rait, J., (2003), "Supporting Global Design Teams through Virtual Design Offices", Proceedings of 2003 International CIRP Design Seminar, Grenoble, France.

Peña-Mora, F., Hussein, K., Vadhavkar, S., and Benjamin, K., (2000), "CAIRO: A Concurrent Engineering Meeting Environment for Virtual Design Teams", Artificial Intelligence in Engineering, Vol. 14, pp. 203-219.

QFD Institute, (2003), http://www.qfdi.org/, [20 Aug 2003].

Schueller, A., and Basson, A.H., (2001), "A Framework for Distributed Conceptual Design", Proceedings of the 13th International Conference on Engineering Design ICED01, Glasgow, UK, pp. 385-391.

Sowa, J., (undated), "A Brief Introduction to Conceptual Graphs", http://www.cs.uah.edu/~delugach/CG/Sowa-intro.html, [20 Aug 2003].

Stacey, M., Clarkson, P.J., and Eckert, C., (2000), "Signposting: an AI Approach to Supporting Human Decision Making in Design", paper DIE-14617, Proceedings of the ASME Design Engineering Technical Conferences 2000, Baltimore, USA.

Suh, N.P., (1990), "The Principles of Design", Oxford University Press.

Szykman, S., Racz, J., Bochenek, C., and Sriram, R.D. (2000), "A Web-Based System for Design Artifact Modeling", Design Studies, Vol. 21, pp145-165.

Toye, G., Cutkosky, M.R., Leifer, L.J., Tenenbaum, J.M., and Glicksman, J., (1993), "SHARE: A Methodology and Environment for Collaborative Product Development", Post-Proceedings of the IEEE Infrastructure for Collaborative Enterprises, CDR-TR #19930507.

Ullman, D.G., (2003), "The Mechanical Design Process", 3rd ed., McGraw-Hill.

Veeranmani, R. and Mehendale, T., (1999), "Online Design and Price Quotations for Complex Product Assemblies: The Case of Overhead Cranes", Paper DETC/CIE-9079, Proceedings of ASME 1999 Design Engineering Technical Conferences, Las Vegas, USA.

Wang, L., Shen, W., Xie, H., Neelamkavil, J., and Pardasani, A., (2002), "Collaborative Conceptual Design – State of the Art and Future Trends", Computer-Aided Design, Vol. 34, pp. 981-996.