Knowledge Discovery and Data Mining within a Design Environment

Mark Haffey
Alex Duffy

CAD Centre
University of Strathclyde
75 Montrose Street
Glasgow
UK
G1 1XJ

Tel. +44 141 548 2374
Fax. +44 141 552 3148

mark@cad.strath.ac.uk

Knowledge Discovery and Data Mining within a Design Environment

M K D Haffey and A H B Duffy
CAD Centre, University of Strathclyde, Glasgow, Scotland, UK, G1 1XJ.

Key words:
Data manipulation, Knowledge transformers, Design (knowledge) Re-use, Knowledge Data Discovery, Data Mining

ABSTRACT

Designers, in the process of satisfying design requirements, generally encounter difficulties in, firstly, understanding the problem and secondly, finding a solution [Cross 1998]. Often the process of understanding the problem and developing a feasible solution are developed simultaneously by proposing a solution to gauge the extent to which the solution satisfies the specific requirements. Support for future design activities has long been recognised to exist in the form of past design cases, however the varying degrees of similarity and dissimilarity found between previous and current design requirements and solutions has restrained the effectiveness of utilising past design solutions. The knowledge embedded within past designs provides a source of experience with the potential to be utilised in future developments provided that the ability to structure and manipulate that knowledge can be made a reality. The importance of providing the ability to manipulate past design knowledge, allows the ranging viewpoints experienced by a designer, during a design process, to be reflected and supported. Data Mining systems are gaining acceptance in several domains but to date remain largely unrecognised in terms of the potential to support design activities. It is the focus of this paper to introduce the functionality possessed within the realm of Data Mining tools, and to evaluate the level of support that may be achieved in manipulating and utilising experiential knowledge to satisfy designers' ranging perspectives throughout a product's development.

1 INTRODUCTION

Knowledge is a valuable commodity which designers both utilise in order to progress the development of a design artefact and generate as a result of the design activity. The process of designing draws knowledge from various sources and accumulates and modifies the available knowledge to reflect in, and progress, a design's development.

The utilisation of traditional Computer Aided Design (CAD) tools has allowed designers to edit and manipulate past designs to the point where it is possible to modify successfully developed previous designs to satisfy design requirements. However, the simple regurgitation of past designs and their associated knowledge is limited in its application as it necessitates the previous requirements of a solution to mirror the current design requirements [Oxman 1990]. Experiential knowledge refers to design knowledge associated with the product(s) within a domain and the specific processes used to develop the design(s). However, the focus of this paper will be on the utility of specific and domain design knowledge. Past experiences possess the potential to be utilised to support future design activities. Past design solutions involves research into applicable domains to ensure the conformity of a design to domain specific heuristics or conditions. Such heuristics or conditions can detail domain specific principles that may have been developed through experienced domain experts recognising trends or patterns within a domain. However, a restraining factor involves limitations imposed by a human's inability to reason with a significant amount of data. Ways in which past design cases may be manipulated to reflect current situations is a key aspect in maximising the effective utilisation of past design knowledge to support designers during decision making processes. Not only can domain specific past design cases be used, but also within specific design domains, more generic domain knowledge may exist. However, the ability to manipulate experiential knowledge is limited by a designer or domain expert, through an inability to fully analyse a domain and extract implicit knowledge. Until recently, computer systems were unable to support domain experts in effectively analysing large repositories of data. In order to discuss how experiential knowledge can be

manipulated to reflect present requirements, it remains pertinent to discuss firstly the nature of the design activity, and secondly the practical implications of utilising experiential knowledge.

2 THE NATURE OF DESIGN

The activity of designing has been described as one that is an *ill-structured* problem solving process [Simon 1973] requiring a *pragmatic* approach to be taken in overcoming immediate problems [Li 1994]. Gero distinguishes technological design from other similarly titled activities through not only the domain but also from the very nature of *exploration* involving goal variables, decision variables and exploring appropriate attribute variables [Gero 1990]. As Li cites, those aspects that distinguish design from general problem solving processes, namely the ill structured, explorative and pragmatic nature of design [Li 1994], have restrained the development of computational design support systems.

2.1 Ill structured

At the initiation of a design task, designers are often presented with incomplete and sometimes ambiguous design specifications (requirement descriptions) thus restraining a designer from recognising the immediate direction(s) necessary to satisfy the requirement(s). As the attributes reflected in a design are interrelated and dependent on values allocated to other associated attributes, the tendencies of attributes to interact exemplifies the problem of missing information. In addition to missing attributes and constraints, at the outset of a design activity, Li also attributes the ill-structured nature of design to the goal of the activity that is ill-defined [Li 1994]. As both Dasgupta and Gero state, the goal itself is the product of the design activity which is poorly understood to start with [Dasgupta 1989, Gero 1990].

2.2 Explorative

The initial need to explore a domain, derives from the often ambiguous nature of a requirement description being weakly stated, incomplete, inconsistent and possessing dependant constraints [Smithers et al. 1989]. The exploration within design begins with the known attributes or specific requirements being used to define and refine an evolving solution. Through the exploration of known solution spaces, designers may begin to understand how the attributes are interrelated by exploring past cases. Designers use past design cases to guide and prompt their solution direction. This allows designers to initially classify their problem specification into similar solution spaces that reflect, at some level, previous cases with similar attributes or requirements. Thus a design space is considered as a space of possible designs which possess the knowledge applicable to support the evolution from a vague requirement description to a product description. In order to effectively utilise experiential knowledge, designers try to look beyond specific design cases in order to identify trends or relationships that may exist between attributes. Termed *Domain Exploration* [Duffy et al. 1995], this process enables the utility of experiential knowledge from within a domain, through identification and structuring, in order to re-use domain specific knowledge as opposed to specific design knowledge, to support a design activity.

2.3 Pragmatic

Li outlines three aspects that demand a pragmatic approach in design [Li 1994], detailing how predictions and associations may be used by a designer in making more grounded decisions. Firstly, the use of design experience allows designers to substitute and gauge values of various, initially unknown, attributes allowing the progression of the design to be maintained. Secondly, a major task in design is the satisfaction of many, often interrelated, objectives. This involves a designer balancing and manipulating these objectives, often sacrificing and compromising some to allow for a strongly desired objective to obtain its desired level. Thirdly, in most cases, several concepts will be generated, each providing strengths and weaknesses in certain aspects of the design and therefore, the designer must assess each from a practical perspective considering differing viewpoints such as, manufacturing, cost, customer expectations, etc.

The varying aspects involved in design require a designer to attain an understanding of the design's intended context in order to make decisions which will be considered acceptable by a majority (those

involved in the design's development and it's identified market). This requires a pragmatic approach of weighting benefits against drawbacks in the quest for success.
A designer's concern is thus in satisfying the objectives of a product specification, through trade off decisions between product entities in order to progress the concept with the greatest potential of succeeding in the market place. The pragmatic nature of designing has substantial gains to be realised through the use of experiential knowledge, enabling designers to predict decision outcomes, based on domain experiential knowledge. The specific issues identified as being inherent in the design activity reflect the strength of support that may be achieved by utilising experiential knowledge. The ability however to utilise experiential knowledge in an efficient and effective manner remains a concern as repositories of data, information and knowledge expand far beyond human cognitive abilities.

3 REQUIREMENTS OF UTILISING EXPERIENTIAL KNOWLEDGE

The ability to utilise experiential knowledge remains conditional on among others, the ability to explore a domain, locating, comprehending and applying the relevant knowledge to support designers as and when required. The structuring of knowledge allows a designer's cognitive abilities to support exploratory thinking, promoting economical search and adaptation [Oxman 1990], through the abstraction and generalisation of knowledge [Akin 1986], thus supporting the mechanisms of knowledge retrieval and utilisation. Experiential knowledge must be presented in a formalism which enables a designer to access, identify and manipulate knowledge that is specific and applicable to their needs. Thus the effective utility of knowledge requires the structuring of knowledge while maintaining a degree of flexibility in the formalism.

3.1 Representation of knowledge

The representation of knowledge serves several purposes in the utility of experiential knowledge. The representation enables the storage, indexing, accessing, retrieving, etc. of specific design knowledge. Several considerations must be addressed if a representation is to enable the manipulation of experiential knowledge and thus improve the effectiveness of reusing experiential knowledge.

- **Represent design cases and their abstractions.** Initial design descriptions represent an abstract concept of the product to be developed. Through the evolution of a design's development, a design description evolves from the abstract to the specific. Thus, in order to utilise experiential knowledge, the knowledge formalism must reflect the need for experiential knowledge to be abstracted to various degrees of abstraction in order to reflect and support the evolution of a design. Through abstraction, the process of generating a succinct design description with the omission of product specific features, designers begin to identify implicit knowledge held within the explicit knowledge of past designs in order to utilise a more widely applicable chunk of knowledge. As Oxman describes, designers assess and classify prior knowledge by abstracting the specifics of a design in order to match them to the situation of the present [Oxman 1990]. The process of generalising design cases, allows designers to explore a potential solution space more efficiently through the analysis of domain specific generalisations.
- **Represent qualitative and quantitative attribute value pairs.** The nature of design draws many ranging forms of knowledge into a product design description, reflecting knowledge of function, behaviour, geometry, process and relations among others [Zhang et al. 1996]. The varying forms of knowledge can be categorised into two distinct forms - qualitative and quantitative. A system intended to support a design process must therefore represent and manipulate both forms of design knowledge.
- **Represent incomplete or missing data**. When a designer is presented with a set of requirements that a proposed product should attain, the requirements often purvey minimal detail, with missing and incomplete information. The missing information is often concerned with the context of the design from two standpoints. Firstly, the constraints imposed due to the context of the product development process, and secondly, the specific context produced by the product during it's operation in the physical world [Finger & Rinderle 1990, Gero 1990]. Thus the designer is not only concerned with missing information but must also visualise and predict expected constraints that are imposed on the design during development and use.

- **Represent relations between design attributes.** At each decision point encountered by a designer during a design process, the designer is relied upon to make judgements that best reflect the needs of the developing product. The relationships that exist between attributes, entities, abstractions and designs provide a valuable source of knowledge. The relationships may be used to support future developments by recognising how, based on experience, the allocation of one attribute's value propagates and determines related attribute value pairs.

3.2 Utilisation of knowledge

The recognition of the relationship between supporting the design process through supporting the utilisation of experiential knowledge has long been recognised [Smithers et al. 1989]. Some identified issues are:

- **Recall experiential knowledge.** The effectiveness of informal approaches to identify and recall experiential knowledge to support a newly encountered design problem is determined by the number of design cases in the repository and the thoroughness of the indexing scheme [Maher & Garza 1997]. The recall of specific design cases and their abstractions may be computationally supported in various ways. Specific designs may be recalled based on attribute value pairs being matched from the current design requirement to those in the knowledge base, detailing all designs which are comparable to the requirement. In addition, based on a lack of known attributes, values and relations existing between attributes in the initial stages of the design process, the need to recall experiential knowledge based on the degree of commonality of attributes between present requirements and stored cases is considered fundamental in maintaining the progression of a design activity.
- **Reflecting the designers viewpoint.** In order for experiential knowledge to be utilised to support a designer, the knowledge must be sufficiently flexible to be customised (in terms of restructuring) to reflect the viewpoint of the designer. The Customised Viewpoint (CV) approach [Duffy & Kerr 1993], recognises the need for designers to be supported with experiential knowledge that reflects the varying perspectives encountered during a design development activity. Through the ability to manipulate domain knowledge, a designer may be supported in making decisions based on past experiences of the domain, thus reducing the pragmatic nature of the design process.

4 MAKING IMPLICIT KNOWLEDGE EXPLICIT

The existence of implicit knowledge embedded within the explicit knowledge of past design cases, provides the opportunity to extract generalised knowledge of a domain. The process of identifying and extracting implicit knowledge, explicates the knowledge into domain generalisations. As a designer's experience evolves within a domain, a designer may begin to identify trends and relationships within and across designs, based on a degree of similarity, which may be more generally applicable and representative of a domain. Certain knowledge transformations that designers carry out in order to make knowledge of past designs more flexible and more widely applicable are discussed [Duffy & Kerr 1993, Sim & Duffy 2000]. For example, designers not only utilise specific past designs but also learn and understand the salient features in order to:

- generalise their knowledge;
- develop heuristics;
- make generalisations to varying levels of abstraction; and,
- evolve and develop relationships between entities and solutions in order to improve the flexibility of the application of knowledge and to support prediction.

A designer with experience of a particular domain has capabilities in recognising how a past design can be manipulated to aid another design activity. This can be seen from the distinction made between novice and expert designers, where a novice generally relies on rule-based reasoning to solve problems while the expert can base judgements from higher level similarities between examples [Hampton 1993]. The aim of supporting designers with the utility of experiential knowledge, necessitates the need to assist designers in bringing the relevant past to the present in a format that

reflects a designer's various viewpoints. The following methods of knowledge transformation show how the modification and generation of knowledge [Persidis & Duffy 1991] may be achieved to explicate implicit knowledge increasing the efficiency of utilising domain experiential knowledge.

Abstraction/detailing. Abstraction is a necessary step in generalisation. An abstraction is involved with the generation of a new version of a concept with less detail than the original. The abstraction of a concept dispenses the knowledge that is superfluous to the requirements while maintaining what is relevant. Detailing promotes the inclusion of specific knowledge.

Association/disassociation. Association is the discovery of relationships or correlations among given entities or descriptions based on logical, casual or statistical relationships. The relation may be expressed as a taxonomic relationship (kind-of) or a compositional relationship (part-of). The relationships can also be expressed by rules and equations showing attribute value conditions that occur frequently together in a group of similar designs. Disassociation reflects a lack of dependency between entities.

Classification/unification. Classification determines a specific index description that may be used to classify a design or associated entity into one of several predefined classes. Each group member is based on a level of similarity in some predetermined perspective, while remaining distinct from other groups. Unification groups all data without the use of a description or criteria.

Clustering (Group rationalisation)/ungroup (decomposition). Group rationalisation involves the grouping or clustering of similar past designs based on the similarity of some criteria or perspective [Duffy & Kerr 1993]. Decomposition or ungrouping removes the grouping.

Derivation(reformulation)/randomisation. Derivation is the process of deriving a piece of knowledge that is based on another piece(s) of knowledge, through a level of dependency. Randomisation transfers one knowledge segment into another by making random changes.

Generalisation/specialisation. Generalisation provides a concise and succinct description or model of a collection of data within a set of designs. Thus the description characterises all of the designs based on the specialisation of the concepts. Specialisation increases the specificity of the description.

Transforming knowledge on such basis, aids in *Domain Exploration* and in turn, brings experiential knowledge to support designers in making more grounded pragmatic decisions. In order for a design to be classified into a cluster, the level of similarity/dissimilarity is assessed to determine the appropriate cluster which is representative of a design. A domain expert can effectively reason and identify underlying similarities or structure within a limited amount of data and information. However, while design information and data repositories continue to expand, techniques to improve the utility of experiential knowledge continue to evolve, attempting to decrease the gap between quantity and analysis capabilities. The simple regurgitation of knowledge limits the applicability of utilising past design knowledge and approaches which store and to a degree manipulate its data, information and knowledge content, are briefly discussed.

5 COMPUTER BASED MODELLING

The introduction of computer systems into the design process enables design data and information to be stored electronically as opposed to the paper format of design documents. What transpired from its introduction was the ability to store data and information in a format which enables the indexing and retrieval, to various degrees of ability, of information and data. The next sections discuss some approaches which recognise the need to store, extract and manipulate data, information and knowledge, discussing their ability to support utilising experiential knowledge.

5.1 Databases

Database systems offer a rigid method of structuring and indexing data and information. Such systems require data to be structured and follow a pattern of consistency in their structure. Their ability to support a design process is limited due to the evolutionary nature of the design process, where, at the outset, the structure and quantity of attributes and entities remain unknown, only being known at the

completion of the design activity. The lack of ability of databases to manipulate and reason with its content, restrains its applicability in supporting the design process [Duffy et al. 1996].

5.2 Knowledge Based Systems

Knowledge Based Systems are computer systems which can enact the role of a human expert, having domain knowledge which it can apply to solve problems, advise and communicate knowledge to others. Knowledge Based Systems provide a domain with a structured source of knowledge, representing both specific design cases and their associated knowledge [Coyne et al. 1987]. However, based on their inability to learn from new experiences and the very nature of their development, several limitations are evident [Kerr & Duffy 1992].

- Their structure is predetermined based on the perception and understanding of the system developer or knowledge engineer.
- The rigidity of their structure, being pre-determined during development, disables them from manipulating and evolving their content to reflect the differing viewpoints experienced by designers.
- Abstractions of specific design cases are not possible. The knowledge, i.e. heuristics, associated with each design may only be inherited from parent classes.

5.3 Case Based Reasoning

Based on analogical reasoning, Case-Based Reasoning utilises abstract forms of past design cases to maintain and support the evolution of a design process [Maher & Garza 1997]. Recognising some level of similarity between abstractions or specific designs, a designer may be supported with abstracted, experiential knowledge which maps between the known attributes of a stored design or abstraction and the current design problem. The utilisation of analogical techniques remains restrictive in their ability to manipulate specific design cases to suit a newly encountered one. Only specific design cases and abstractions are manipulated based on a degree of similarity between design attributes. No induction techniques exist to identify and extract implicit knowledge from within a domain.

5.4 Knowledge Data Discovery

Knowledge Data Discovery (KDD) involves the analysis of repositories of data and information for the purpose of identifying and extracting implicit or unseen knowledge which may be submersed in a data set. KDD represents a process of explicating implicit knowledge, through data preparation, data transformation, data analysis and data interpretation among others [Fayyad et al. 1996]. Data Mining, the analysis step in the KDD process involves the identification of patterns or implicit knowledge from within a design data set. The ability to explicate implicit knowledge may be provided through various techniques, such as those mentioned in section 4, enabling the system to present concise domain generalisations. The techniques presented in section 4, are utilised to transform specific design cases into domain generalisations through a data mining system's ability to structure and restructure domain knowledge.

The ability of KDD, in particular data mining, to extract and present generalised domain knowledge from within specific design cases will be discussed further, considering the goals of KDD and data mining, the methods of knowledge transformation and the specific knowledge representations utilised.

6 DATA MINING IN A DESIGN CONTEXT

Despite receiving relatively little attention in design to date, data mining has been successfully utilised in areas such as fraud detection, identifying consumer trends and maximising the efficiency of production processes. Drawing from distinct research areas, such as machine learning, knowledge representation and statistics, such data mining systems can be used to identify, modify and extract knowledge. The complex nature of the design process, has to date restrained the development of computer programs to support and aid a designer during product development. Data mining tools currently provide the core functionalities required to support designers through: the identification and exploration of relevant design spaces, supporting the use of experiential knowledge to reduce the

pragmatic extent of decision making; the structuring of experiential knowledge through their ability to generalise, abstract, cluster and associate within design domains; and, possessing the ability to learn from new experiences. Such tools may be utilised to structure and manipulate experiential knowledge in order to map between previous design solution principles and current design perspectives.

6.1 Data Mining Goals

Data mining is the process of discovering new knowledge or knowledge of interest such as patterns, associations, changes, anomalies, structures, principles, etc., from data or information repositories. An integral part of the Knowledge Data Discovery (KDD) process [Adriaans & Zantinge 1996, Fayyad et al. 1996], data mining capabilities realise the achievement of the knowledge discovery goals of KDD. Such primary goals may be distinguished based on the intended use of the system as:

- *verification* - systems used to verify or discount an hypothesis;
- *discovery* - systems autonomously search for new patterns and present discoveries to the users. Discovery can be further broken down to:
  - *prediction* - where the system discovers patterns for use in predicting the future behaviour of some entities; and,
  - *description* - system finds patterns for the purpose of presenting them to the user in an interpretable form.

The distinction between the prediction and description goals is useful for understanding the overall discovery goal, but the boundaries between the two often overlap (description models may be representative of the domain while providing prediction, and vice versa).

6.2 Knowledge Transformation Techniques

Data mining tools have a wide range of techniques which, depending on the goal or reason for implementation, determine the specific or suite of techniques to be utilised. Many such techniques have been in use for more than a decade, being utilised in specialised analysis tools, whose only prominent constraint was their inability to analyse large volumes of data. A range of specific techniques [Fayyad et al. 1996, Shapiro & Frawley 1991] used in data mining systems are presented in order to illustrate how the goals of a KDD process may be realised and introduce the functionality that can now be harnessed from such computational techniques.

Abstraction - generation of an entity's description with less detail and specificity.

Association - discovery of associative relations among a given set of entities which satisfy a given set of criteria.

Classification - determines a specific criteria that may be used to classify a design or associated entity into one of several predefined classes.

Clustering - segregate through (dis)similarity of a data set into a finite set of categories.

Derivation - generates new knowledge as derived from another piece of knowledge.

Generalisation - determines a domain description that is representative of a complete domain or grouping of design cases.

6.3 Knowledge Representation Techniques

The utilisation of various knowledge formalisms, provides several benefits to various concerns in design. The ability to represent the same knowledge through various representation techniques supports designers at differing times in a design process. For example, through the use of domain specific equations a designer may determine attribute values based on known values or the use of a dependency network to support the design development process. In addition, various formalisms support user's in understanding the results and in turn applying the knowledge to their design viewpoint. Data Mining systems utilise specific knowledge formalisms to support the structuring of data, information and knowledge within a domain's ranging viewpoints.

Dependency Networks - enables the generation of models based on identified significant dependencies between variables. Figure 1 details a basic dependency network presenting the relational strengths between entities. The figure details how the independent attribute 'CUBIC' possesses a strong relationship with 'CARGO DWT'. The information displayed in the network provides designers with the ability to make decisions while determining and recognising those related 'trade-off' attributes.

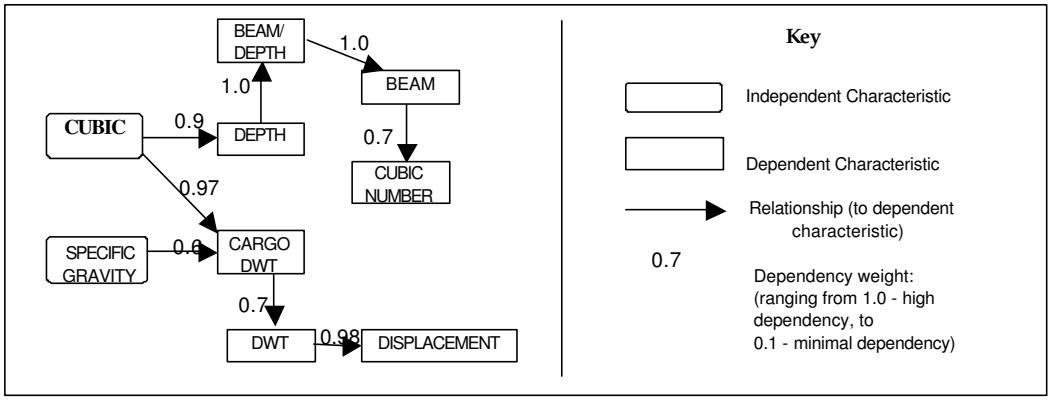

Figure 1: Dependency Network showing the Utilisation of Weighted Relationships

Decision Trees - enable an activity or process to be modelled in terms of specific decision points. The tree like structure, as shown in Figure 2, represents a set of decision points with all possible outcomes represented. Such a representation of knowledge is similar, in knowledge content, to the rule-based formalism. The generation of a decision tree based on a user's contextual needs enables designers to procedularise an activity, based on past experiences, to ensure the satisfaction of all considerations.

Equations - Another formalism for the dependencies between attributes is in the form of an equation (1). The quantified dependencies of attributes extracted from a source of past design examples may be represented as:

$$Cb = 0.968 - (0.269 \times Vs / (\sqrt{L} / 0.3048)) \quad (1)$$

where the attribute Block Coefficient (Cb) may be determined based on known values for the attributes Length (L) and Speed (Vs).

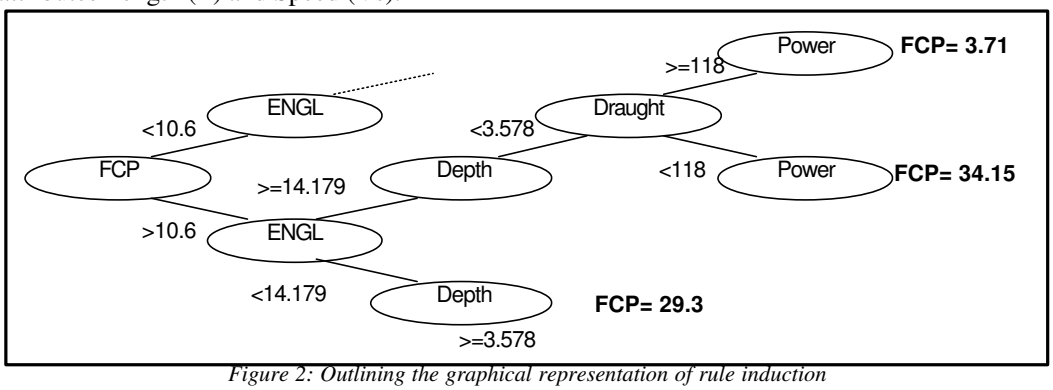

Figure 2: Outlining the graphical representation of rule induction

Neural Network's (NN's) - most prominent feature is their ability to learn. An NN may be trained to map a set of input patterns onto a corresponding set of output patterns, through the exposure of past examples. Training is performed through the gradual adoption of the internal weights of the network, to reduce the difference between actual network outputs and the desired network outputs (based on a given set of inputs). A major drawback of an NN, is its 'black box' approach to detailing their rationale used, upon which it has based decisions [Lu et al. 1996].

Production rules - involve the extraction of IF condition THEN action rules from data sets. Thus, if a condition is satisfied then a logical conclusion may be drawn. For example:

IF (attribute) draught < (value) 12.92
AND (attribute) lob > (value) 6.18
AND (attribute) engwt > (value) 350.5
THEN (conclusion) ENGL = (result) 12.13

This rule derives the attribute 'ENGL' based on statistics that draught, lob and engwt all have a strong positive correlation with ENGL. If…. Then rules are particularly helpful in the development of

traditional rule based expert systems [Winston 1984].

The knowledge transformation and representation techniques, discussed in sections 6.2 and 6.3 respectively, are not a comprehensive list but serve to show, not only a range of tasks that data mining can accomplish but in aiding the process of KDD to ensure that the goals are identified and matched onto the above approaches. However, it is considered here that the above forms of knowledge representations and techniques, all have specific roles to play in a design process, depending on the stage and viewpoint of the designer.

In order to evaluate the degree of support that may be gained through the utilisation of data mining techniques, an evaluation was initiated to assess the functionality that data mining tools can provide to a technological design domain. Based on the differing considerations discussed in sections 2 and 3, five knowledge discovery/data mining systems were obtained and utilised to assess their potential ability to address the ill structured, explorative and pragmatic nature of a design activity. The selection of the five systems from the range of commercial and research systems, though not all encompassing, the intent was in utilising and evaluating a functionally representative cross section of systems. Brief overviews of five systems are presented, followed by a discussion that encapsulates how the functionality of the systems may be used to support design activities.

7 DATA MINING SYSTEMS

Data mining systems may be initially categorised as either mono-strategy systems presenting a single technique such as clustering, or multi-strategy systems which incorporate a suite of techniques. Of the five systems utilised two systems, BRUTE and CN2, reflected the mono-strategy approach, and two systems, CLEMENTINE and MOBAL, reflected the multi-strategy approach. An additional Customised Viewpoint tool, PERSPECT is introduced.

Mono-Strategy Sysems

BRUTE, an experimental research system, was developed by Segal, to exhaustively search a data set with the goal of inducing rules. The system is directed mainly for use in fields were qualitative attributes are common, but through the process of making values discrete, the system may manipulate quantitative data [Segal 1992].

CN2, developed by Clark and Niblett, is a rule induction program that takes a set of examples, in the form of attribute values pairs, and generates a set of rules from which to generalise examples. [Clark and Matwin 1993]. As in BRUTE, the CN2 system can, unsupervised (autonomously), generate rules using quantitative attributes, but remains restrictive in its inability to predict the value of a quantitative attribute, unless previously made discrete

Both BRUTE and CN2 are distinctly restrained in terms of their functionality and graphical representation of knowledge within a domain, both present easily interpretable results, in the form of production rules.

Multi-Strategy Systems

Two multi-strategy systems follow in the discussion, namely CLEMENTINE and MOBAL, that realise the needs of designers requiring different functions, manipulations and representations to allow experiential knowledge to be effectively utilised.

CLEMENTINE [SPSS] allows users to extract and manipulate data, and visualise trends and relationships. This system provides neural network and decision tree techniques through classification and discovery methods. The very nature of the manipulation of data, allows the user to not only control rule generation but the system also displays the information and knowledge in such a way that the user is supported in identifying patterns and relationships, thus reflecting both supervised and unsupervised learning. This functionality is brought to the system through a user interface which represents the flow of data through a serious of transformational and representational processes.

MOBAL [ILP ESPRIT] was developed for the purpose of developing, validating and maintaining domain knowledge models of an application domain. MOBAL allows the incremental evolution of a domain model to be generated through both supervised and unsupervised learning. Being iterative in nature, the system allows the user to incrementally develop a model drawing on the user's knowledge of the domain to guide and develop models. The output or knowledge representation is presented to

the user in the form of logical facts and rules. At any point in the generation of a model, the user can control, direct and inspect the knowledge input, transformation and output through graphical (i.e. graphs) or textual (i.e. rules,) representations.

The PERSPECT system [Duffy & Duffy 1996] goes beyond the previous systems discussed, by providing users with an interactive design support system. Developed to assess the utility of the Customised Viewpoint (CV) approach, the system provides a suite of software which not only draws knowledge directly from past designs held within CAD software, manipulating, generating and modelling knowledge specific to requirements, but closes the cycle by drawing the new knowledge back into design systems to continue the design activity with supporting knowledge. The PERSPECT system was designed to, during preliminary design activities, retrieve and present knowledge to support a designer in making decisions without delaying the design's progress. PERSPECT supports, at the highest level, the design process by supporting designers in the following design tasks.

- Preparation of a domain model – supports designers in creating a new or checking and evolving existing models to reflect the implicit knowledge within a domain.
- Initiation of a design model – supports designers in utilising the model with the intent of progressing an existing design solution through (a) predicting unknown values through utilising specific design cases or identifying relevant (sub)equations which reflect the designers circumstances or viewpoint, or (b) generalising existing domain models to reduce the complexity or number of unknown attributes which may be required, reflecting the knowledge available and required within the designer's viewpoint.

Of the systems discussed, the ability to generate, modify and evolve models and equations were discussed. PERSPECT however, extended the functionalities by facilitating the initiation and utilisation of the models. PERSPECT's ability to extend the utilisation of generated models stems from the systems architecture and interfacing subsystems.

8 EVALUATION OF DATA MINING IN DESIGN

The evaluation of the five data mining/knowledge discovery systems is not presented as an exhaustive critique of the systems, but rather one that is indicative of the nature of a technological design environment. Sections 2, 3 and 4 will each be discussed in context of the five systems as follows. Firstly, the aspects which address the nature of the design activity will be discussed, namely: provide structure; support exploration; and, support the pragmatic nature of design. Secondly, the systems' abilities to represent knowledge from various considerations are discussed followed by an analysis of the level of knowledge utility obtained by the systems. Lastly, the systems will be discussed in terms of the knowledge transformation techniques which enable the explication of implicit knowledge to be achieved. Table 1 presents an evaluation of the knowledge discovery and data mining systems within a design environment. In deriving the table, the following observations could be reached.

Provide Structure - Through the structuring of domain and design knowledge all of the systems possess the abilities to identify trends and patterns and present explicated implicit domain knowledge as models. The restraining ability of the systems to support the recognition of the relations within and between design cases and abstracted design concepts, is considered to restrain the user's understanding of a domain.

Support Exploration - The importance of identifying and utilising applicable design spaces ensures that any explicated knowledge reflects the requirements of a designer and not just a representation of a domain (you get what you inspect and not what you expect). CLEMENTINE can determine relevant design spaces based on the attributes in a designer's viewpoint. The determination of a design space in the MOBAL system is presented in the form of positive and negative examples with the intent of disclosing the past cases which support and provide parameters respectively to the user. PERSPECT however, effectively supports the exploration of a domain while maintaining the relationship of supervised learning, as the user interactively refines and directs the focus or viewpoint of the system. Both BRUTE and CN2 provide limited support in the exploration of a domain, being limited to explorations based on attribute value pairs.

Support Pragmatic Nature of Design - In the determination of attribute value pairs, both the MOBAL and PERSPECT systems may use a case based approach, thus presenting designers with specific design case values. The CLEMENTINE system predicts attribute values based on designers' needs and the explicated relations between attributes and entities within the domain, in addition detailing the maximum and minimum attribute values found within the domain. Through predicting and presenting attribute values and the underlying relations, the MOBAL and PERSPECT systems enable the progression of the design activity to be maintained by generating domain specific dependency networks to aid in the assessment of trade-off decisions between attributes. However, what appears to reduce a dependency network's ability to support a design activity, is the lack of knowledge concerning relative weights between dependants, reflected only in the PERSPECT system. In addition, a parametric type network which details maximum and minimum ranges, may present boundary conditions, beyond which a designer will recognise to act with awareness.

Representation of Knowledge - The ability of the systems, CN2 and BRUTE, to represent explicated knowledge in various formalisms was a restraining factor in terms of diversity of the systems and also in the understanding of the generated knowledge. The multi-strategy systems, through a range of knowledge representations, supported the process of instilling a level of understanding of the explicated knowledge in the user. Considered a major strength, the ability of the systems to represent the knowledge in differing formalisms allows the knowledge to reflect the specific nature of support required, such as predicting an attribute's value through a production rule or decision list, or the effect of the allocation of an attribute on the attributes within a design description through a trained Neural Network or equation.

Utilisation of Knowledge - The incapability of the systems to utilise both qualitative and quantitative attribute value pairs represents a deficiency in terms of supporting the nature of design. In addition, the inability to provide information on specific design cases or abstracted concepts inhibits the recognition of utilising case based values as a means to progress an activity. While possessing the ability to utilise specific values, actual cases are not recovered by the systems. The analysis of a concept during preliminary design stages presents a valuable commodity and reflects an effective utility of available knowledge. As a designer makes judgements and decisions concerning a concept, the allocation of an attribute value may be ascertained by experiential knowledge to identify and highlight non-conforming values or those values which are considered to be beyond the experiences of past domain conforming designs.

Manipulate Experiential Knowledge - The manipulation of knowledge reflects the goal of the KDD process. With the varying requirements related to the nature of the design activity, the requirement of a system to reflect a multi-strategy approach, incorporating a suite of knowledge transformation techniques, is considered vital.

EVALUATION CRITIQUE \ SYSTEMS	Mono		Multi		
	BRUTE	CN2	CLEMENTINE	MOBAL	PERSPECT
Provide Structure					
Reflect relationships between - Designs			◆	◆	◆
Reflect relationships between - Abstractions				◆	◆
Reflect relationships between - Attributes	◆	◆	◆	◆	◆
Support Exploration					
Refine explorations to identify solution spaces			◆		◆
Explore designs based on attribute value relations	◆	◆	◆	◆	◆
Support Pragmatic Nature of Design					
Predict or derive attribute values	◆	◆	◆	◆	◆
Detail relational strengths (between attributes, concepts, etc.)				◆	
Representation of Knowledge					
Represent specific design cases	◆	◆	◆	◆	◆
Represent design abstractions				◆	◆
Represent specific design knowledge (models)			◆	◆	◆
Utilise multiple knowledge representations			◆	◆	◆
Incorporate qualitative attribute value pairs	◆	◆	◆	◆	
Incorporate quantitative attribute value pairs			◆	◆	◆
Representation of incomplete design data	◆	◆	◆	◆	◆
Relations between designs/abstractions/attributes presented	◆	◆	◆	◆	◆
Utilisation of Knowledge					
Utilise and manipulate qualitative data	◆	◆	◆	◆	
Utilise and manipulate quantitative data			◆	◆	◆
Recall/utilise specific design cases				◆	◆
Recall/utilise design/abstractions based on attribute value pairs					◆
Recall/utilise design/abstractions based on common attributes				◆	◆
Making Implicit Knowledge Explicit					
Abstraction	◆	◆	◆	◆	◆
Association	◆	◆	◆	◆	◆
Classification			◆	◆	◆
Clustering			◆		◆
Derivation			◆	◆	◆
Generalisation	◆	◆	◆	◆	◆

Table 1:Evaluation of Knowledge Discovery and Data Mining systems (◆ represents a degree of support)

As experiential knowledge is manipulated and condensed into a domain, the degree of contextual knowledge needs to be assessed. Being presented with a model or heuristic without the detail knowledge of whether a model was extracted autonomously, generated by a domain expert (or novice), or pre-compiled as a recognised domain heuristic, may reduce the confidence of the user in its appropriateness to their viewpoint. In order to promote the user's confidence in the process, adequate contextual knowledge must be presented through detailing: firstly, the quality of obtained results, such as number of missing fields encountered during analysis, number of cases used (disclosing concept or abstraction), the ratio between actual cases used and number of cases present, statistical results and obtained accuracy; and secondly, the recognition of attribute value pairs as being maximum/ minimum values, average values, predicted values or case based values.

Within the process of knowledge discovery, a lack of integration between CAD systems and data mining systems, highlights an issue which introduces error susceptible activities such as information loss and disjointed fields between donor and knowledge discovery systems.

A foreseeable issue in utilising data mining techniques within a technological design domain is with the vocabulary used to represent a product or entity. The very 'mismatch' between one designer's interpretation and the allocation of an attribute name, may reflect a misrepresentation of knowledge within a domain, jeopardising the accuracy of results.

9 CONCLUSION

Past design cases possess the potential to provide support not only in the form of specific design knowledge, but also in generating design domain principles based on implicit knowledge. As design repositories expand, the activities of manually identifying, representing and utilising implicit knowledge from within domains becomes increasingly difficult. Therefore, these three quite distinct

activities must be sufficiently and effectively supported if the activity of supporting future design developments, with implicit knowledge of past designs (e.g. in the form of design rules or principles) is to become a reality. The functionality delivered by data mining can realise these three activities. This paper has presented KDD as an approach in analysing large design repositories and interrogating data, enabling a controlled level of implicit knowledge to be *identified*. The ability of data mining systems to possess varying techniques of *representing* and *utilising* experiential knowledge through various knowledge representations has been discussed. In addition, the ability of a system to display the results through a range of techniques, enabling users to analyse generated patterns or models more effectively and also in assessing their relevancy to a specific domain, thus promoting a factor of confidence in the results, has been presented. This paper has shown that the functionality of data mining has a significant contribution to provide within technological design. However, due to the developments focusing upon practical applications, aside from the domain of technological design, the systems provide some of the core functionalities necessary but individually are functionally deficient regarding the effective utilisation of experiential design knowledge. The integration of knowledge discovery and data mining techniques into a design environment can provide a design support tool capable of not only creating new design knowledge, but that can evolve and learn from that knowledge to support future design scenarios.

References

Adriaans, P. and Zantinge, D., (1996), Data Mining, Addison-Wesley.

Akin, O., (1986), Psychology of Architectural Design, Pion, London.

Clark, P. and Matwin, S., (1993), Using qualitative models to guide inductive learning. In P. Utgoff, editor, Proc. Tenth Int. Machine Learning Conference (ML-93), pages 49-56, CA. Kaufmann. The CN2 system:(http://www.cs.utexas.edu/users/pclark/software.html).

Coyne, R.D., Rosenman, M.A., Radford, A.D. and Gero, J.S., (1987), Innovation and Creativity in Knowledge-Based CAD. In Expert Systems in Computer Aided Design, J.S. Gero (ed.), pp 435-465, Amsterdam: North-Holland.

Cross, N., (1998), Engineering Design Methods: Strategies for Product Development, Second Edition, Wiley and Sons.

Dasgupta, S., (1989),The structure of design process. In M. Yovits (ed.), Advances in Computers, Vol. 28, Academic Press, New York, pp 1-67.

Duffy, A.H.B. and Kerr, S.M., (1993), Customised Perspectives of Past Designs from Automated Group Rationalisation. Artificial Intelligence in Engineering, Vol. 8 (3) pp 183-200.

Duffy, S.M., Duffy, A.H.B. and MacCallum, K.J., (1995), A Design Reuse Model, Proc. Tenth Int. Conf. Eng. Design, 490-495.

Duffy, S.M. and Duffy, A.H.B., (1996), Sharing the Learning Activity using Intelligent CAD, Artificial Intelligence in Engineering Design, Analysis and manufacturing, Vol. 10 pp 83-100.

Duffy, A.H.B., Persidis, A. and MacCallum, K.J., (1996), NODES: a numerical and object based modelling system for conceptual engineering design, Knowledge-Based Systems, Vol. 9, pp 183-206.

Duffy, A.H.B., (1997), The "What" and "How" of Learning in Design, IEEE Expert, May/June, pp71-76.

Fayyad, U.M., Shapiro, P., Smyth, P. and Uthurusamy, R., (1996), Advances in Knowledge Discovery and Data Mining, AAAI, MIT Press.

Fayyad, U.M., Houssler, D. and Stolorz, P., (1996), KDD for Science Data Analysis: Issues and Examples, AAAI, MIT Press.

Finger, S. and Rinderle, J.R., (1990), Transforming behavioural and physical representations of mechanical designs. In Proceedings of the First International Workshop on Formal Methods in Engineering Design, Manufacturing and Assembly Colorado pp133-151.

Gero, J.S., (1990), Design Prototypes: A Knowledge Representation Schema for Design, A.I. Magazine, Vol. 11 (4) pp 26-36, Winter.

Gero, J.S., (1998), Design Tools that Learn, Advances in Engineering Software, Vol. 29 (10) pp.755-

761, Elsevier.
Hampton, J., (1993), Prototype Models of Concept Representation. In Van Mechelen, I., Hampton, J., Michalski, R.S. and Theuris, P. (eds.), Categories and Concepts. Theorertical Views and Inductive Data Analysis, Cognitive Science Series, Academic Press, London, pp 67-95.
ILP ESPRIT basic research project. The Mobal system(http://nathan.gmd.de/projects/ml/mobal/mobal.html).
Kerr, S.M. and Duffy, A.H.B., (1992), Machine Learning in Design Workshop, 2nd International Conference on Artificial Intelligence in Design, Pittsburgh, Pennsylvania, USA.
Li, H., (1994), Machine Learning of Design Concepts, Computational Mechanics Publications, Southampton, UK, and Boston, USA.
Lu, H., Setiono, R. and Liu, H., (1996), Effective Data Mining using Neural Networks, IEEE Transactions on Knowledge and Data Engineering, Vol. 8 (6), pp 957-961.
Maher, M.L. and Garza, de Silva, A.G., (1997), Case-Based Reasoning in Design, IEEE Expert, March/April, pp 34-41.
Oxman, R., (1990), Prior Knowledge in Design: A Dynamic Knowledge-Based Model of Design and Creativity, Design Studies Vol. 11(1), pp 17-28, January.
Persidis, A. and Duffy, A.H.B., (1991), Learning in Design, Intelligent CAD III, Yoshikawa, H., Arbab, F. and Tomigama, T. (eds.) pp 251-272, Elsevier, North-Holland.
Segal, R., The Brute system: (http://www.cs.washington.edu/homes/segal/brute.html).
Shapiro, P. and Frawley, (1991), W.J., Knowledge Discovery in Databases, AAAI, MIT Press.
Simon, H.A., (1973), The Structure of Ill-Structured Problems, Artificial Intelligence 4 (3-4),pp 181-201.
Sim, S. K. and Duffy, A. H. B., (2000), Evaluating a Model of Learning in Design Using Protocol Analysis, 6th International Conference on Artificial Intelligence in Design'00, 26-29 June.
Smithers, T., Conkie, A., Doheny, J., Logan, B. and Millington, K., (1989), Design as Intelligent Behaviour: An AI in Design Research Programme, Fourth International Conference on Applications of Artificial Intelligence in Engineering, 11-14 July.
SPSS Inc. Headquarters, 233 S. Wacker Drive, 11th floor, Chicago, Illinois 60606. The Clementine system: (http://www.isl.co.uk/clem.html).
Winston, P.H., (1984), Artificial Intelligence, Reading, Mass, Addison Wesley.
Zhang, Y., MacCallum, K.J. and Duffy, A.H.B., (1996), Product Knowledge Modelling and Management. Workshop proceedings: 2nd WDK Workshop on Product Strategy, 3rd and 4th June, Delft, Netherlands.