Jurnal Teknologi, 39(D) Dis. 2003: 27–44

INTELLIGENT OBJECT ANALYSER FOR CONCEPTUAL DATABASE DESIGN MODEL

SHAHRUL AZMAN MOHD. NOAH[1] & MICHAEL WILLIAMS[2]

Abstract. Conceptual database design is seen as the most important stage of a database design process, as the conceptual model produced at this stage is the first design model constructed with formal and detailed semantics. The stage, however, is also viewed as a difficult task for designers, and the potential for committing and correcting errors is significant. As a result, it is not surprising to see the emergence of a number of automated systems employing artificial intelligence (AI) techniques in providing assistance in the design of such a model. However, the majority of these tools have been focussed on the task of design synthesis with limited diagnostic capabilities. This paper, therefore, proposed a set of diagnostic rules meant for intelligent diagnosis of conceptual database design model. These rules have been implemented using a prototype tool called the Intelligent Object Analyser (IOA). Initial testing on a number of domains has so far produce encouraging results. However, there still remains undiagnosed cases whereby the proposed diagnostic rules incapable of diagnosing. The paper concludes with a discussion of areas for future development in this field.

Keywords: Conceptual modelling, database design, intelligent system, knowledge-based systems

Abstrak. Fasa reka bentuk konseptual pangkalan data merupakan fasa terpenting dalam proses pembangunan sebuah sistem pangkalan data memandangkan model konseptual yang terhasil melalui fasa ini adalah merupakan model semantik pertama yang dijanakan. Walau bagaimanapun, fasa ini juga digambarkan sebagai suatu fasa yang sukar dan kemungkinan untuk berlaku ralat pemodelan adalah tinggi. Sehubungan dengan itu, beberapa sistem pengautomasian yang berasaskan teknik kecerdasan buatan telah dibangunkan yang bertujuan untuk membantu mereka bentuk model konseptual tersebut. Tetapi kebanyakan dari pada sistem ini lebih tertumpu kepada aspek sintesis reka bentuk dengan kemampuan diagnosis yang sangat terhad. Melalui kertas ini, satu set petua diagnosis telah dicadangkan dan diimplementasikan di dalam prototai p sistem iaitu *Intelligent Object Analyser* (IOA). Sungguhpun pengujian awal telah menunjukkan keputusan yang menggalakkan, masih lagi terdapat ralat pemodelan yang gagal dikesan oleh petua diagnosis yang dicadangkan. Kertas ini diakhiri dengan perbincangan dan perluasan penyelidikan dalam bidang ini.

Kata kunci: Model konseptual, reka bentuk pangkalan data, sistem kecerdasan, sistem berasaskan pengetahuan

1.0 INTRODUCTION

Database design process usually proceeds into four distinguishable stages, namely:

[1] Faculty of Information Science and Technology, National University of Malaysia, 43600 UKM, Bangi, Selangor, Malaysia. Tel.: 03-89296182, Fax.: 03-89356732, E-mail: samn@ftsm.ukm.my

[2] European Business Management School, University of Wales Swansea, Singleton Park, Swansea SA2 8PP UK. Tel: 01792-295181 Fax: 01792-285626 E-mail: m.d.williams@swansea.ac.uk

requirements specification and analysis, conceptual design, logical design and physical design. Conceptual database design, however, is seen as the most important stage of a database design process [1]. This is due to the fact that a conceptual model produced at this stage is the first design model constructed with formal and detailed semantics. Although contemporary database design tools have excellent facilities for drawing and maintaining a conceptual database model, these tools are characterised as being passive and incapable of supporting the basic characteristics of conceptual database design process. Such a process is regarded as being a knowledge-intensive activity that begins with a vague requirement of the problem domain model of what is to be done and results in a highly detailed formal object namely a database model.

The conceptual database design stage is also viewed as a difficult task for designers [2,3], and the potential for committing and correcting errors is significant. Due to its importance and yet error prone, it is not surprising to see the development of a number of intelligent database design tools aiming to support the tasks of conceptual database design [4,5]. These intelligent tools employ artificial intelligence (AI) technique mainly in the form of rules in providing assistance to users during the process of database analysis and design. Such assistance can be broadly categorised as *design synthesis and design diagnosis* [6]. Design synthesis occurs where a tool is capable of generating design output, whereas design diagnosis occurs where a tool attempts to detect inconsistencies and redundancies that may exist within the design, and suggests corrections to the design.

However, the majority of these tools have been focussed on the task of design synthesis with limited diagnostic capabilities. For instance the View Creation System [7] and Object Design Assistant [8] tools contain rules which dictate the order or steps of database analysis and design tasks; and rules for extracting entities, relationships and attributes from a set of requirements. Other tools such as SECSI [9] and E-R Translator [10] provides rules for automatic transformation of an ER model into a set of relations usually in the form of 4th Normal Forms (4NF).

This paper presents the development of an intelligent database design tool called the Intelligent Object Analyser (IOA) which is meant to provide support for the conceptual stage of database design. Although IOA has the capacity of performing two aspects of design (design synthesis and design diagnosis), the main characteristic exhibited by IOA is within the aspect of conceptual database design diagnosis by using a set of diagnostic rules. This paper is organised into the following four sections. Firstly, the intelligent approach to conceptual database design by means of the IOA tool is being discussed. This section is intended to provide readers the required background information, and to illustrate the overall process currently adhered by the IOA tool. Secondly, a detail discussion of the rules used during the process of conceptual database design diagnosis is provided. Discussion and analysis of the effectiveness of the proposed rules proceed after that. Finally, the paper concludes with conclusions and future research work that may be drawn from this work.

2.0 INTELLIGENT APPROACH TO CONCEPTUAL DATABASE DESIGN

IOA was developed to provide support during the conceptual stage of database design process. In constructing a conceptual design model, IOA employs the Object Modelling Technique (OMT) methodology advocated by Rumbaugh *et al.* [11]. Therefore, the outputs produced by the IOA exhibit the key object-oriented characteristics of object classes and instances, and the three most popular forms of data abstraction; association, aggregation and generalisation (including multiple inheritance).

The processing employed by the IOA system can be generally categorised as a two-step procedure as follows:

- The first step involves creating an initial representation of the application domain (known as the problem domain model) and subsequent refinement of this model.
- The second step involves the creation of an analysis model, and subjecting this model with analysis and refinement rules to generate an object-oriented conceptual schema.

To support this two-step procedure, the IOA architecture is organised into three major components: the user interface, the inference engine and the knowledge base as illustrated in Figure 1. The *user interface* is a medium for communication between the user and the IOA. The IOA employs an interactive window system interface which includes the use of pull down menus and a natural language interface. The tool contains multiple menus for controlling a design session; viewing an evolving design model; saving and loading a design model. The specification of the information requirements provided by the user takes the form of restricted natural language description that describes the application domain. This natural language description can be submitted as a whole or on a single statement basis. Dialogue that usually occurs as a *question and answer* process also takes the form of natural language. The answer may either take the form of: *yes or no* for confirmation in any action to be taken by the tool; selection of a series of suggestions provided by the tool; or additional information concerning the application domain when requested by the tool.

The *inference engine* acts as a controller that controls the interaction between the user and the tool. It directs any part of the user input to the correct processor for processing and decides which rule or rules to trigger during the analysis process.

The IOA knowledge base is organised into rules and facts. Rules correspond to knowledge of how to perform the design task (the order in which design activities take place); detecting and resolving ambiguities, redundancies and inconsistencies within an evolving design; and handling the gradual augmentation of an evolving design as a design session progresses. The set of rules are a combination of both

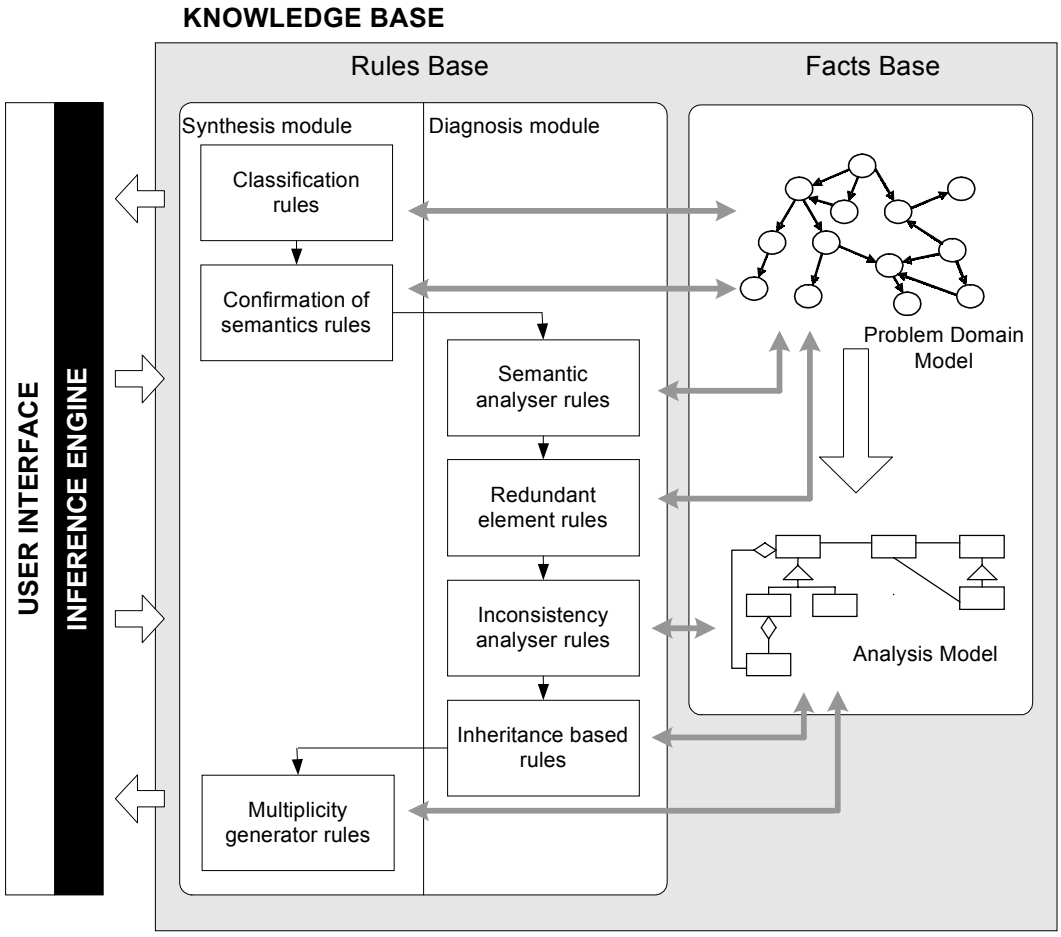

Figure 1 The architecture of the IOA tool

production and procedural rules. The production rules take the form of condition-action pairs containing knowledge of those based on database analysis and design rules; and those based on heuristics developed as a result of experience gained during the testing and evaluation of the tool. The procedural rules organised into meta rules, control the order in which various analysis and design tasks are performed; and select the appropriate rules to be applied at any given stage. The rules base can be further divided into two modules: the synthesis module and the diagnosis module. As illustrated in Figure 1, the synthesis module is mainly use to generate or synthesise the required model given the facts from the facts base or by direct questioning to the user. The diagnosis module is responsible to detect any form of inconsistencies or errors that may exists.

Facts are used to present two views of the application domain; an initial representation (the problem domain model) as provided by the user, and the object-oriented design generated from this initial representation. The problem domain model which represents the user's view of the application domain takes the form of a semantic network structured into concepts and arcs. This model is progressively refined and augmented resulting in the creation of an analysis model. This analysis model is represented as an object model and is comprised of a series of object classes and relationshi ps between these object classes. The model views the application domain by means of an object-oriented conceptual model. Facts regarding multiplicity and membershi p requirements constraints are obtained from the information held within the analysis model and by direct questioning with the user.

Using the IOA, the first stage of processing requires a set of declarative statement that described the application domain to be submitted to IOA. These statements are a variation of the method of interactive schema specification described by Baldiserra *et al.* [12] being based upon the binary model describe by Bracchi *et al*. [13]. These statements are used to construct the problem domain model representing the application domain. Once constructed, the semantics aspects of the problem domain are being classified and analysed (by using the classification rules and the confirmation of semantic rules); that is, whether each structure within the model represents generalisation, aggregation or association.

The classified problem domain model is then submitted to a series of refinement procedures in order to detect and resolve forms of semantic inconsistencies and redundant elements that may exist. These aspects of diagnosis are managed by the semantic analyser rules and redundant element rules, which can be performed both with and without the requirement of user input (sometimes referred as *external* and *internal* validation respectively). Once such inconsistencies have been resolved, IOA makes use of the problem domain model in order to generate an analysis model (a conceptual model in object oriented form). Inconsistencies in the forms of missing properties and inheritance based redundancies are being conducted at this stage of processing. The inconsistency analyser rules and the inheritance based rules are respectively responsible for these two types of inconsistencies. Information regarding the multiplicity (integrity constraints) of the model is being handled by the multi plicity generator rules. A detail discussion on the tasks of intelligent database design diagnosing is being provided in the next section of this paper.

3.0 RULES FOR CONCEPTUAL DATABASE DESIGN DIAGNOSIS

According to the previously discussed approach of IOA intelligent processing, there are four types of design inconsistencies (errors) that should be detected and resolved [14]. These are as follows:

(i) *Semantic inconsistencies*. Inconsistencies occurring as a result of missing links (i.e. no associated relationships for a particular concepts) or transitivity that may exists within the generalisation or aggregation hierarchies.
(ii) *Redundant elements*. Inconsistencies occurring as a result of synonyms such as synonymous concepts and relationships which usually lead in turn to redundancies.
(iii) *Inconsistent concepts*. Inconsistencies occurring as a result of missing properties (i.e. no associated properties for a particular concept).
(iv) *Redundant inherited properties and relationships*. Redundancy occurring within a generalisation hierarchy where a generic class (superclass) and its corresponding specific class(es) (subclass) contain the same properties or participate with the same relationships.

In the remaining of this section, we will discuss each of these types of inconsistencies and how such inconsistencies can be intelligently detected and diagnosed with the use diagnostic rules implemented in the IOA tool.

3.1 Semantic Inconsistencies

Semantic inconsistencies refer to concepts that do not participate in any relationships. For example, Figure 2 illustrates an isolated concept, "*Patient*" which does not participate in any forms of relationships.

In intelligent database design diagnosing, such a concept is detected by inspecting each node of the problem domain and the corresponding links associated to it. Once the isolated concept is found, a series of interaction with the user will then take place in order to seek the user's decision on what to do next. As for example, the following dialogue is generated by IOA when the isolated concept of "*Patient*" is being detected.

```
IOA >   It was found that the concept PATIENT has no associated
        relationships. Do you wish to ...
```

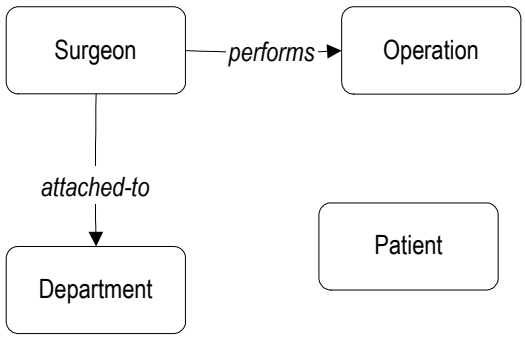

Figure 2 Semantic inconsistencies (isolated concept)

```
        a) Provide the relationships associated with PATIENT
        now.
        b) Remove the concept PATIENT from your specification.
        c) Take no action for the time being
User >  a
IOA >   Please enter relationships associated with PATIENT -
        type END to finish
User >  patient undergoes operation
User >  End
```

Another type of error considered in this category is the existence of transitivity. Transitivity holds in aggregation relationships [11,15,16], that is, if 'A is a component part of B' and 'B is a component part of C', then 'A is also a component part of C'. If all these three were to appear in the evolving design model, the relationship 'A is a component part of C' is redundant and should be removed. Transitivity also holds in generalisation relationships [16], that is, if 'A is a subclass of B' and 'B is a subclass of C', then 'A is also a subclass of C'. If all of these three relationships were to appear in the problem domain model, the relationship 'A is a subclass of C' is redundant and should be removed.

The detections of transitivity in intelligent database design diagnosing should not only consider a two-tier level of generalisation or aggregation hierarchies but in any n-tier level of hierarchies. Similar to the detection of concepts with missing links, IOA detects transitivity by inspecting each nodes and links that corresponds to the following rule.

Semantic inconsistency rules

(i) Transitivity in a generalisation hierarchy

IF (A is-a B), (B is-a C) and (A is-a C) exist in the evolving design model
THEN (A is-a C) is redundant and should be removed.

(ii) Transitivity in an aggregation hierarchy

IF (A part-of B), (B part-of C) and (A part-of C) exist in the evolving design model
THEN (A part-of C) is redundant and should be removed.

Figure 3 illustrates an example of transitivity between the concepts "*Person*", "*Staff*" and "*Surgeon*". Once such an occurrence of transitivity is being detected by IOA, a series of dialogue is generated to the user for confirmation of its removal as illustrated below.

```
IOA >   It was found that SURGEON is a STAFF and SURGEON is a
        PERSON, however, STAFF is a PERSON. This exhibits
        redundancy. The structure SURGEON is a PERSON should be
        removed. Do you    agree? (Y/N)
User >  Y
```

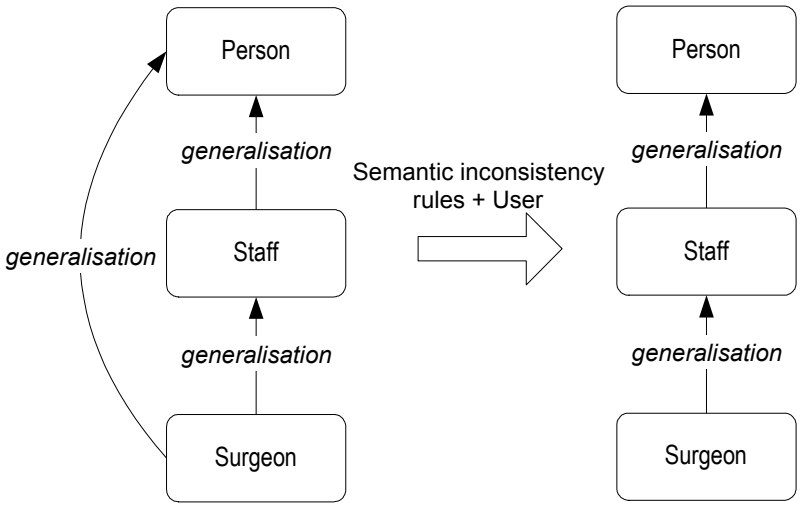

Figure 3 Detection of transitivity in a generalisation hierarchy

3.2 Redundant Elements

Redundant elements which are the result of synonyms such as synonymous concepts and relationships are conceivably the most complicated tasks performed in intelligent database design diagnosing. The most common way of detecting such design inconsistencies is perhaps by examining pairs of relationships between two distinct concepts of the form "*A verb-phrase B*" and "*C verb-phrase D*" [15] or sometimes referred to as object type and mismatch rules. For instance, comparing the structures of "*Surgeon Performs Operation*" and "*Physician Performs Operation*" (as illustrated in Figure 4), would result in the suggestion that the concepts "*Surgeon" and "Physician*" are synonymous. Comparing the structures "*Doctor Examines Patient*" and "*Doctor Inspects Patient*" (as illustrated in Figure 5) on the other hand would result in suggestion that the relationships "*Examines*" and "*Inspects*" are synonymous.

The synonymous concept rules and the synonymous verb phrase rules are used by the IOA to detect such redundant elements respectively. Therefore, the existence of the structures "*Surgeon Performs Operation*" and "*Physician Performs Operation"* (Figure 4) will be detected by the use of synonymous concept rules, the situation being brought to the attention of the user who may give consideration to removing one of the redundant concepts as illustrated by the following dialogue.

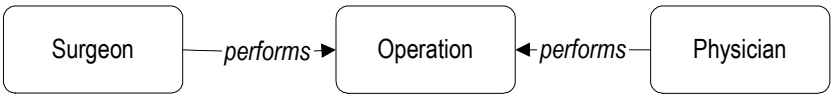

Figure 4 Synonymous concepts

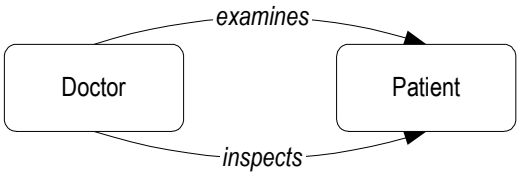

Figure 5 Synonymous verb-phrase relationshi ps

```
IOA >   An element of redundancy may occur in the following
        structures:
        SURGEON PERFORMS OPERATION
        PHYSICIAN PERFORMS OPERATION
        Which of the following is correct?
        a) The concept SURGEON and PHYSICIAN express the same
        thing.
        b) One of the concepts is a specific type of the other
        concept or vice-versa.
        c) One of the structures is wrong.
        d) Leave as it is.
User >  a
IOA>    Since SURGEON and PHYSICIAN are synonymous, one of the
        concepts has to be removed. Which of the concepts would
        you like to remove?
        a) SURGEON b) PHYSICIAN
User>   b
```

Synonymous concepts rules

Let (A v_1 B) and (X v_2 Y) be the pair of associations structures, where A, B, X and Y are the concepts and v_1 and v_2 are the verb-phrases.

(i) **IF** (A = X and $v_1 = v_2$)

THEN B may be a synonym of Y **OR** B is a subclass of Y **OR** Y is a subclass of B.

(ii) IF (A = Y and $v_1 = v_2$)

THEN B may be a synonym of X **OR** B is a subclass of X **OR** X is a subclass of B.

(iii) IF (B = X and $v_1 = v_2$)

THEN A may be a synonym of Y **OR** A is a subclass of Y **OR** Y is a subclass of A.

(iv) IF (B = Y and $v_1 = v_2$)

THEN A may be a synonym of X **OR** A is a subclass of X **OR** X is a subclass of B.

Synonymous/related verb-phrase rules

Let (A v_1 B) and (X v_2 Y) be the pair of associations structures, where A, B, X and Y are the concepts and v_1 and v_2 are the verb-phrases.

IF (A = X **AND** B = Y) **OR** (A = Y **AND** B = X)
THEN v_1 may be a synonym of v_2 **OR** v_1 and v_2 are related by tenses.

However, there are cases where synonymous are apparent within a design, but do not actually exist. For example the structures of "*Student Registers Course*" and "*Graduate-Student Registers Course*" suggest the existence of synonyms, but would be identified as participating in a generalisation relationship via the following dialogue.

```
IOA >   An element of redundancy may occur in the following
        structures: STUDENT REGISTERS COURSE
        GRADUATE-STUDENT REGISTERS COURSE
        Which of the following is correct?
        a) The concept STUDENT and GRADUATE-STUDENT express the
        same thing.
        b) One of the concepts is a specific type of the other
        concept or vice-versa.
        c) One of the structures is wrong.
        d) Leave as it is.
User >  b

IOA>    IOA will create a subclass/superclass relationship between
        the concepts STUDENT and GRADUATE-STUDENT.  Which of the
        following is true?
```

```
        a) GRADUATE-STUDENT is a specific type of STUDENT
        b) STUDENT is a specific type of GRADUATE-STUDENT
User>   a
```

In this case, IOA will update its evolving design model by incorporating the generalisation structure between the concepts "*Graduate-Student*" and "*Student*" and removing the link "*Registers*" between "*Graduate-Student*" and "*Course*".

The synonymous verb-phrase rules on the other hand are used to detect the possible synonyms among verb-phrase relationships as previously illustrated by the structure of "*Doctor Examines Patient*" and "*Doctor Inspects Patient*". Similarly, the removal of such a redundant structure will be brought to the attention of the user as illustrated by the following dialogue.

```
IOA >   Do the structures DOCTOR EXAMINES PATIENT and DOCTOR
        INSPECTS PATIENT represent the same information?
        Meaning that the words EXAMINES and INSPECTS are synonymous
        or are related. (Y/N)
User >  Y

IOA>    Since the above statements represent the same information,
        one should be removed. Which structure would you like to
        remove?
        a) DOCTOR EXAMINES PATIENT b) DOCTOR INSPECTS PATIENT
User>   b
```

3.3 Inconsistent Concepts

Inconsistent concepts refer to concepts with no associated properties. The concept "*Graduate-Student*" in Figure 6 falls into this category, and when detected by IOA, the following dialogue is triggered by IOA's diagnostic engine requesting confirmation from the user of the preferred course of action as whether to provide the property(ies) associated with the concept, or to remove the inconsistent concept from the evolving design model, or to take no immediate action.

```
IOA >   It was found that the concept GRADUATE-STUDENT has no
        associated properties.
        Do you wish to.....
        a) Provide the properties associated with GRADUATE-
        STUDENT now.
        b) Remove the concept GRADUATE-STUDENT from your
        specification.
        c) Take no action for the time being.
User >  a
```

```
IOA >   Please enter properties associated with GRADUATE-STUDENT
        - type END to finish.
User >  course-type
        end
```

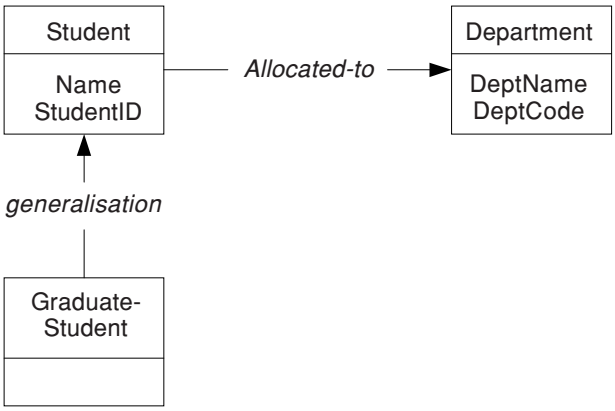

Figure 6 An inconsistent concept

3.4 Redundant Inherited Properties and Relationships

Redundant inherited properties and relationshi ps exist within generalisation hierarchies. For example if a concept "*Student*" has property "*studentId*", therefore the concept "*Graduate-Student*" which is the subclass of "*Student*" inherit the property "*studentId*" from the concept "*Student*". In this case, the property "*studentId*" should not be explicitly represented within the concept "*Graduate-Student*". Similarly any relationships that the concept "*Student"* partici pates, the concept "*Graduate-Student*" will also participate. From this basis, redundancy may occur in terms of inherited properties and inherited association and aggregation relationships of a generalisation hierarchy if a generic concept and its corresponding specific concept(s) explicitly contain the same attributes or participate with the same relationships.

The detection of the aforementioned elements of redundancies is achieved by the use of the following rules.

Redundant inherited properties and relationships rules

(i) **IF** (*x* property-of *A*) **AND** (*x* property-of *B*) **AND** (*A* is-a *B*)

THEN (*x* property-of *A*) may be redundant and has to be removed.

(ii) **IF** (*C* part-of *A*) AND (*C* part-of *B*) **AND** (*A* is-a *B*)

THEN (*C* part-of *A*) may be redundant and has to be removed.

(iii) **IF** (*A association C*) **AND** (*B association C*) **AND** (*A* is-a *B*)

THEN (*A association C*) is redundant and has to be removed.

For example, the structures illustrated in Figure 7, would eventually results in the following dialogues being triggered by the IOA's diagnostic engine.

```
IOA >   The attribute NAME exists in the class PERSON and its
        subclass(es) (i.e. STUDENT). The attribute NAME should
        be removed from the subclass(es) because it can inherit
        the attribute NAME from class PERSON. Do you agree? (Y/
        N)
User >  Y
```

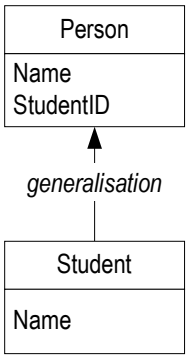

Figure 7 Redundant inherited property

However, in cases involving multilevel inheritance hierarchies or multiple subclasses as illustrated by the structures of Figure 8, the following dialogue instead will be triggered by the IOA's diagnostic engine.

```
IOA >   The attribute RESEARCH-TYPE appears in the concept
        GRADUATE-STUDENT and its superclass PERSON. This exhibit
        redundancy. Is it true to say that the attribute RESEARCH-
        TYPE is also an attribute of the other subclass(es) of
        the concept PERSON (i.e., STUDENT, LECTURER, and
        PROFESSOR)? (Y/N)
User >  N

IOA >   In that case the attribute RESEARCH-TYPE will be removed
        from the concept PERSON and retained in the concept
        GRADUATE-STUDENT since the attribute RESEARCH-TYPE is
```

```
            not an attribute to the other subclasses(es) of the
            concept PERSON (i.e., STUDENT, LECURER, and PROFESSOR).
            Do you agree? (Y/N)
    User >  Y
```

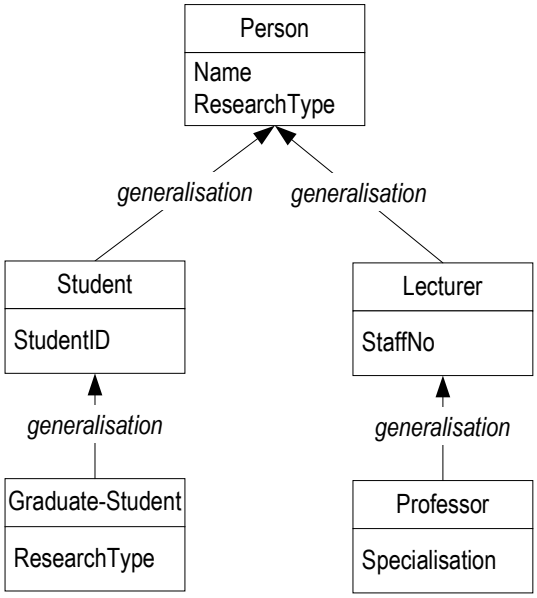

Figure 8 Redundant inherited property in multilevel inheritance hierarchies

Similar user/tool interactions are required during the detection and resolving of redundant inherited aggregation and association relationships within a generalisation hierarchy.

4.0 DISCUSSION

Testing performed on a wide range applications related to the university, healthcare and library domains has so far produced encouraging results. In this case, the IOA was exposed to a range of design problems within the general scope of conceptual modelling design. These design problems were extracted from the available literature, the advantage being that the accompanying solutions could be used as a benchmark and compared with the IOA suggested solution. Initial results indicate that about 79% of the errors introduce for the university domain have been successfully detected and resolved and 74% and 76% respectively for the healthcare and library domains [17].

Although the presented diagnostic rules are seen capable of facilitating detection and correction of the four types of database design inconsistency (semantic inconsistency, inconsistent concepts, redundant inherited properties and relationships,

and redundant elements), there remains great potential for errors to remain undetected by IOA that could easily be identified by a human designer. Indeed, the results of the testing demonstrated that approximately 21%, 26% and 24% of design inconsistencies respectively for the university, healthcare and library domains remain undetected by IOA's diagnostic engine of which all of these errors relate to the redundant elements.

Implicitly, this work also indicated that detection and removal of three types of inconsistency (semantic inconsistencies; inconsistent concepts; and redundant inherited properties and relationships) is a relatively straightforward process which requires IOA to inspect each concept within an evolving design model and its corresponding links to other concepts. Any such inconsistency thus detected is brought to the attention of the user and subsequently resolved. However, inconsistency in the forms of redundant elements (occurring due to the presence of synonymous concepts and relationships) cannot be completely resolved by relying only on the proposed diagnostic rules.

From our observation there are three undiagnosed forms relating to the redundant elements and may be defined by the following cases. Consider the pair of association structures (A v_1 B) and (X v_2 Y), where A, B, X and Y are concepts and v_1 and v_2 are verb-phrases.

- **Case 1**: ((A = X) **AND** (v_1 ¹ v_2) **AND** (B ¹ Y)) **OR** ((A ¹ X) **AND** (v_1 ¹ v_2) **AND** (B = Y)) **OR** ((A = Y) **AND** (v_1 ¹ v_2) **AND** (B ¹ X)) **OR** ((A ¹ Y) **AND** (v_1 ¹ v_2) **AND** (B = X)). For example "*Academic Teaches Course*" and "*Course Taught-By Lecturer*". This case can be resolved if a tool can identify similarities between verb phrases or similarities between concepts.
- **Case 2**: ((A ¹ X) **AND** (v_1 = v_2) **AND** (B ¹ Y)) **OR** ((A ¹ Y) **AND** (v_1 = v_2) **AND** (B ¹ X)). For example "*Lecturer Advises Graduate-Student*" and "*Academic Advises Postgraduate-Student*". This case can be resolved if a tool can identify similarities between either pair of concepts.
- **Case 3**: ((A ¹ X) **AND** (v_1 ¹ v_2) **AND** (B ¹ Y)) **OR** ((A ¹ Y) **AND** (v_1 ¹ v_2) **AND** (B ¹ X)). For example "*Lecturer Advises Graduate-Student*" and "*Academic Consults Postgraduate-Student*". This case can be resolved if a tool can identify similarities between either pair of concepts and similarities between the verb phrases, or if similarities between both pairs of concepts can be identified.

The limitations of detecting the above cases can be potentially overcome by incorporating forms of domain specific knowledge instead of knowledge about database design alone [18]. Such knowledge is also implicitly used by human designers when performing the tasks of database design in order to interact with users, make helpful suggestions and inferences, and identify potential errors and inconsistencies. Although a number of approaches to representing domain knowledge have been proposed such as the dictionary approach [19], the thesaurus approach [20] and the knowledge

reconciliation approach [21], their context has been focused in providing assistance to users engaged in the task of design synthesis. The thesaurus approach incorporated within the Object Design Assistant [19] for instance is intended to obviate the need to ask what may be viewed as being trivial questions of the user during the analysis and design process. The dictionary technique implemented in the Intelligent Interview System of Kawaguchi *et al.* [18] on the other hand, is used during the *interview process,* extracting a series of simple queries that the eventual database will be expected to satisfy. As for the knowledge reconciliation technique implemented by the Common Sense Business Reasoner [20], the knowledge is used as a basis for providing the user with meaningful suggestions of concepts and relationships missing from the evolving database design. Such suggestions are produced after a series of reconciliations between the system's domain knowledge and the user's specified application domain.

5.0 CONCLUSIONS AND FUTURE RESEARCH WORKS

This paper has demonstrated the development of an intelligent database design tool; the Intelligent Object Analyser (IOA) meant to support the conceptual stage of database design process. During use, the IOA has exhibited the capacity of producing a consistent conceptual design model by using a set of diagnostic rules. However, as previously discussed, there are still remains a number of undiagnosed cases which can be resolved by human designers. Therefore, current research work has been focussed on developing knowledge structure corresponding to the dictionary, thesaurus and knowledge reconciliation approaches previously described and integrating these approaches with the proposed diagnostic rules.

Other further research avenues in this area has been directed towards introducing more generic knowledge (as compared to the dictionary, thesaurus and knowledge reconciliation knowledge which are claimed to be too domain specific) either in the form of common sense knowledge or an ontology that could be used across domain applications. Initial studies in this area have been produced by Storey *et al.* [22], and by Storey and Dey [23]. Few researchers are also currently investigating the possibility of applying the concept of domain reusability whereby knowledge from a previous design process could be used in the next design activities, either in the form of domain model [24] or an operational system [25].

Another possible area of future research is that of applying the intelligent diagnostic approach to other more complicated and recent trends in database design such as that of data warehousing. Although a number of tools for supporting data warehouse design are currently available [26, 27], the application of AI technology to such tools, particularly within the context of design diagnosis, would appear to offer many interesting opportunities for further investigation.

REFERENCES

[1] Saake, G. 1994. Conceptual Modelling of Databases Applications, in *Information System and Artificial Intelligence: Integration Aspects*. Berlin: Springer-Verlag, 213-232.

[2] Batra, D. and S. R. Antony. 1994. "Novice errors in conceptual design". *European Journal of Information System,* 3(1):57-69.

[3] Batra, D. and S. H. Zanakis. 1994. "A conceptual database design approach based on rules and heuristics". *European Journal of Information Systems.* 3(3):228-239.

[4] Lloyd-Williams, M. and P. Beynon-Davies. 1992. "Expert system for database design: a comparative review". *Artificial Intelligence Review*. 6:263-283.

[5] V.C. Storey. 1993. "A selective survey of the use of artificial intelligence for database design systems". *Data and Knowledge Engineering*. 11(1):61-102.

[6] Oxman, R and J.S. Gero. 1987. "Using an expert system for design diagnosis and synthesis". *Expert Systems: The International Journal of Knowledge Engineering*. 4(1):4-14.

[7] Storey, V. C. and R.C. Goldstein. 1990. "An expert view creation system for database design", *Expert Systems Review*. 2(3), 19-45.

[8] Lloyd-Williams, M. 1993. "Expert system support for object-oriented database design". *International Journal of Applied Expert Systems.* 1(3):197-212.

[9] Bouzeghoub, M. 1992. Using expert systems in schema design. In: Loucopoulos, P. and Zicari, R. (editors), *Conceptual Modelling, Databases, and CASE: an Integrated View of Information Systems Development*. New York: Wiley, 465-487.

[10] Briand, H., H. Habrias, J.F. Hue, and Y. Simon. 1985. Expert system for translating an E-R diagram into databases. In: Chen, P. (editor), "Entity-Relationship Approach: The Use of ER Concept in Knowledge Representation". Proceeding of the 4th Entity Relationship Conference, Amsterdam: North-Holland, 199-206.

[11] Rumbaugh, J., M. Blaha, W. Premerlani, F. Eddy and W. Lorenson. 1991. *Object-Oriented Modelling and Design*. Englewood-Cliffs, NJ: Prentice-Hall.

[12] Baldiserra, C., S. Ceri, G. Pelagatti, and G. Bracchi. 1979. "Interactive specification and formal verification of user's views in database design". In: Proceeding of the 5th International Conference on Very Large Databases, Rio de Janeiro, Brazil. 262-272.

[13] Bracchi, G., P. Paolini, and G. Pelagatti. 1976. Binary logical associations in data modelling. In: Nijsen, G. M. (editor), *Modelling in Data Base Management Systems*, 125-148. Amsterdam: North-Holland.

[14] Noah, S.A. and M.D. Williams. 2000. "Exploring and Validating the Contributions of Real-World Knowledge to the Diagnostic Performance of Automated Database Design Tools". In Proceedings of 15th IEEE International Conference on Automated Software Engineering (ASE2000), IEEE: New York. 177-186.

[15] Storey, V.C. 1993. "Understanding Semantic Relationships". *Very Large Database Journal.* **2**(4): 455-488.

[16] Winston, M.E., R. Chaffin and D. Herrmann. 1987. "A Taxonomy of Part-Whole Relations". *Cognitive Science*. **11**: 417-444.

[17] Noah, S. A. and Williams, M. 2003. "Integrating artificial intelligence technique and lexicon of verbs in automated database design diagnosing". *Malaysian Journal of Computer Science*. 16(1), 9-23.

[18] Storey, V. C. 1992. "Real world knowledge for databases". *Journal of Database Administration*. 3(1):1-19.

[19] Kawaguchi, A., N. Taoka, R. Mizoguchi, T. Yamaguchi and O. Kokusho. 1986. "An Intelligent Interview System for Conceptual Design Of Database". In Proceedings of The 7th European Conference on Artificial Intelligence, Conference Services Ltd.: London, 1-7.

[20] Lloyd-Williams M. 1997. "Exploiting Domain Knowledge During the Automated Design of Object-Oriented Databases". In Embley, D. W. & Goldstein, R. C. (Eds.) Proceedings of The 16th International Conference on Conceptual Modelling, Spinger-Verlag: Berlin, 16-29.

[21] Storey, V.C., R.C. Goldstein, R.H.L. Chiang, and D. Dey. 1993. "A Common-Sense Reasoning Facility Based on the Entity-Relationship Model". In. Elmasri, R. A., Kouramajian, V. & Thalheim, B. (Eds.) Proceedings of The 12th International Conference on the Entity Relationship Approach, Springer-Verlag: Berlin, 218-229.

[22] Storey, V.C., R.C. Goldstein, and H. Ullrich. 2002. "Naïve Semantics to Support Automated Database Design". *IEEE Transactions on Knowledge and Data Engineering.* **14**(1): 1-12.

[23] Storey, V.C. and D. Dey. 2002. "A Methodology for Learning Across Application Domains for Database Design Systems". *IEEE Transactions on Knowledge and Data Engineering*. 14(1): 13-28.

[24] Sugumaran V., M. Tanniru and V.C. Storey. 2000. "Supporting Reuse in System Analysis". *Communications of the ACM*. 43(11): 312-322.

[25] Tatjana, W., I. Rozman, and M. Druzovec. 2000. "Database Reuse as a Support for Conceptual Database Modelling". In Proceedings of the 4th Joint Conference on Knowledge-Based Software Engineering, Amsterdam: IOS Press, 129-132.

[26] Golfarelli, M and S. Rizzi. 2001. "WanD: a CASE Tool for Data Warehouse Design". In Demo Proceedings of The 17th International Conference on Data Engineering (ICDE 2001), Heidelberg, Germany, 7-9.

[27] Franconi, E. & G., Ng. 2000. "The iŸCom Tool for Intelligent Conceptual Modelling". In Proceedings of The 7th International Workshop on Knowledge Representation meets Databases (KRDB'00), Springer-Verlag: Berlin, Germany, 45-53.