ISSN 1744-1986

The Open University

Technical Report N° 2008/17

Criteria for assessing object-relational quality

Patricia Roberts

8th December 2008

Department of Computing
Faculty of Mathematics and Computing
The Open University
Walton Hall,
Milton Keynes
MK7 6AA
United Kingdom

http://computing.open.ac.uk

Technical Report

Criteria for assessing object-relational quality

Patricia Roberts

07 December 2007

Abstract

Object-relational databases combine traditional relational database design with object-oriented features. Although object-relational features have been available to database designers for years, the best way to use these features has not been fully evaluated. This approach for assessing the relative quality of object-relational database designs draws on approaches for measuring quality in software engineering and conceptual modelling. This paper proposes a set of four criteria to help assess whether a particular design displays the characteristics of quality: integrity, simplicity, flexibility and seamlessness. The criteria are set in a context of a framework for assessing quality that considers the viewpoints and priorities of different stakeholders in the design: for example, Systems Analysts, Business Analysts, Application Developers and Database Administrators. Future work will address the issue of whether these criteria are sufficient to distinguish between good and bad O-R designs. Once the framework has been used, the criteria they will be revisited to examine whether they have been useful in deriving judgements about the overall quality of the designs. Furthermore, the criteria will be assessed to find whether they could be useful in a wider context.

1. Introduction

Object-relational (OR) extensions had been included in the relational databases offered by the major DBMS vendors for many years. In 1999 the SQL standard (International Standards Organisation (ISO), 1999) was extended to include many of the features that could be grouped together under the heading '*object-relational*', including the introduction of type and table hierarchies: ways to represent class hierarchies in relational databases. The publication of SQL:2003 (International Standards Organisation (ISO), 2003) brought further OR features into the standard.

This paper forms part of research into the use of object-relational database features and the ways that object-relational database design quality could be enhanced, focussing on the single design step of transforming a "correct" UML model into a logical O-R schema. To assess the quality in an object-relational database design, it is first necessary to identify the aspects of the design that represent quality. In the past many attempts have been made to assess the quality of models within Information Systems, as in Moody (1998) and Kesh (1995) but the introduction of object-oriented features such as type hierarchies into the design changes the landscape. To achieve the aim of a quality design the definition of quality must be broken down into criteria that can be assessed. Too many criteria could prove confusing and difficult to apply, so it would be useful to have a minimal set of criteria that could be tightly defined to assess a design. One criterion that has been used to measure the quality of a model is simplicity, as in Piattini et al (2001) and others, but the addition of type hierarchies is unlikely to contribute to the simplicity of a database design. Can other criteria be found that indicate an improvement in quality for non-trivial databases that include hierarchies?

The wider research will investigate the quality criteria that could be used to assess database design when object-oriented features are included. Database design is the process of transforming a conceptual model into a logical database schema. For the purpose of this paper, we assume that the conceptual model provides a "correct" starting point, in the sense that it is a complete representation of the system requirements, and is of appropriate quality. Thus, we can focus our attention on the decisions to be made during the transformation itself, and the impact those decisions may have on the quality of the resulting schema. Since we are exploring the use of O-R structures to represent object-oriented concepts, we assume further that the conceptual model is expressed as a UML class diagram.

In this paper a hierarchic model for the assessment of this object-relational quality is proposed. At the top level, as shown in Fig. 1, is the main aim (a quality design) which is broken down into those characteristics of a design that contribute to its quality: functionality, usability and maintainability. These can then be broken down again into criteria that could be used to assess the design. Bansiya and Davis (2002) developed a quality hierarchy for object-oriented design quality, based on the ISO/9126 standard (International Standards Organisation (ISO), 2001) for software product quality. Their approach is to use Software Engineering metrics and apply them to the design (counts of the various features that can be measured) and map these back to the qualities. Moody (2005) argues that there is little evidence that measuring internal characteristics of a design can be demonstrated to improve the desired external quality characteristics. My approach will be to combine some of the features from Bansiya and Davis (2002), using the ISO quality characteristics, together with criteria that can be used to assess the different designs against each other. Fig.1. shows the relationship of the chosen criteria with the quality characteristics.

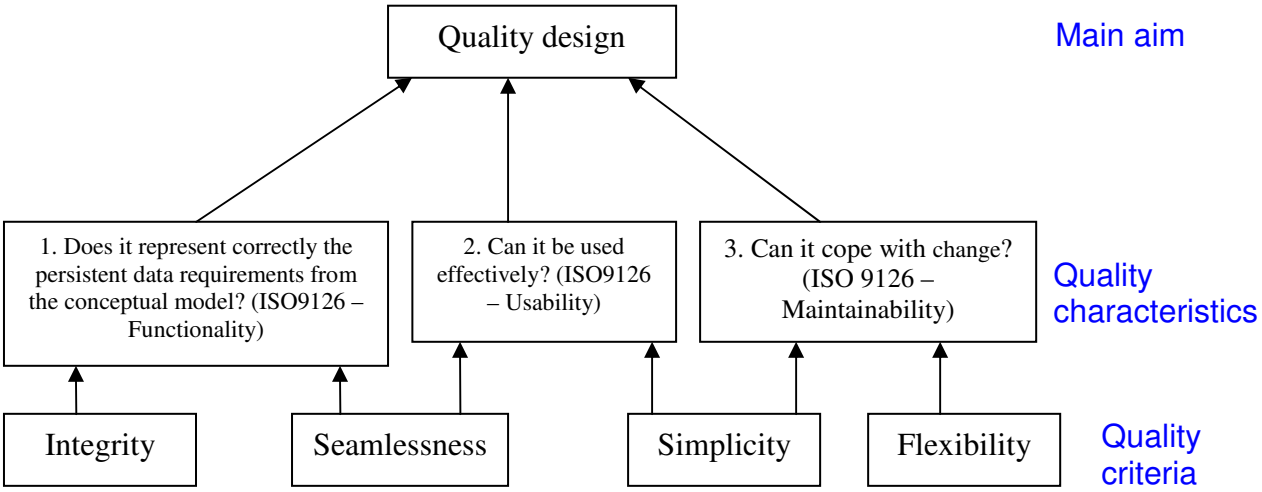

Fig 1. A hierarchic model for object-relational quality assessment

The use of Software Engineering metrics to get an absolute measure of quality will be avoided, and the approach taken will be comparative rather than absolute; not that one design is good (measured on some scale) but that one design is better in one of the aspects (such as simplicity) than another. Of course, the relative importance of the three qualities in the middle level, the 'Quality characteristics' depends upon the view point; a business analyst might be most concerned with 1 (capturing the semantics), an application developer might be most concerned with 2 (the usability of the design) and a database administrator would certainly be interested in 3 (how the design can cope with a changing world).

The proposed set of quality criteria is a development from and extension to the work of other researchers; Kesh (1995) developed a set of criteria to evaluate the quality of entity relationship models which are divided into those concerned with the content and those

concerned with the structure. Three years later, Moody (1998) looked at the same area of ER design quality and proposed another set of criteria, in this case considered from the viewpoint of different stakeholders. Fig 2 shows these two sets of criteria.

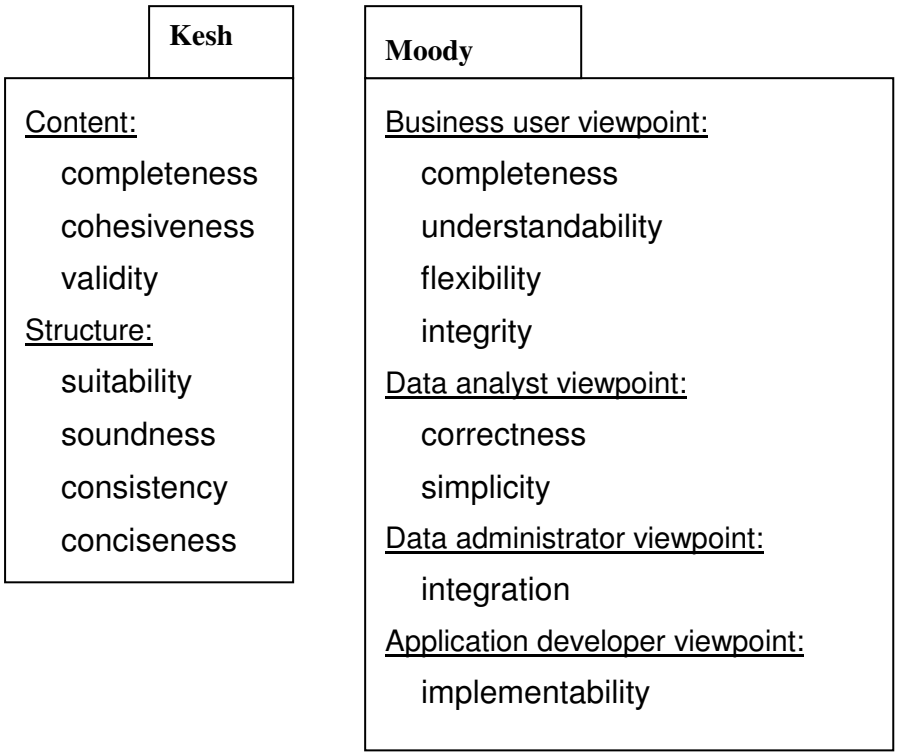

Fig 2. Quality criteria from Kesh (1995) and Moody (1998)

There are potentially a large number of criteria that could be used to assess the quality of a design. It is important, in choosing a set of criteria for assessing the quality of an object-relational database design, to examine whether all the criteria are necessary or whether they overlap or conflict with each other. Some of the criteria used by Kesh (1995) and Moody (1998) fall outside the scope of an object-relational design, as shown in Fig.3. The three criteria from Kesh, which concern the content of the design (completeness, cohesiveness and validity), are outside the scope of this work, as we are assuming that we have a correct and complete conceptual model to work from. However, the notions of completeness and validity of the object-relational design are included in the definition of integrity from Moody's criteria. The criteria from Moody that concern the conceptual model: completeness, understandability and correctness are also outside the scope. At the other end of the design process two criteria from Moody, implementability and integration, would be concerned with implementation in a particular DBMS. This research considers design and so is not restricted to a particular DBMS product, but takes a more general view using the SQL:2003 standard so implementation factors are not under consideration.

There remain three criteria from Moody which are particularly relevant to O-R designers: integrity, simplicity and flexibility which, as highlighted in Fig.3, form the core of criteria that will be used in this work. Integrity relates to Kesh's terms of suitability, soundness and consistency, in that integrity requires that the semantics of the conceptual model are carried forward correctly, without loss or distortion. Conciseness is defined by Kesh (1995) as containing "*the minimum number of elements*" and is therefore contained in the definition by Moody of simplicity. O-R design is concerned with taking a conceptual model and creating an implementation that works for the problem application.

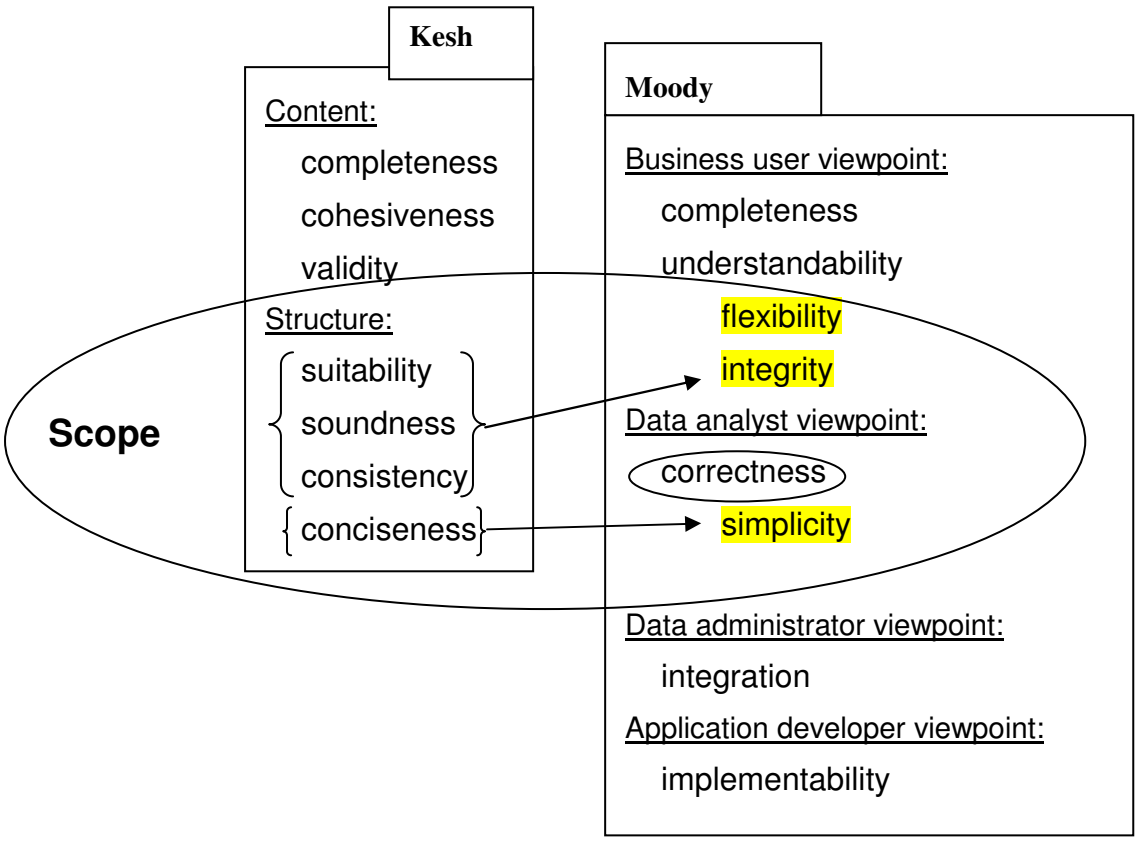

Fig 3. Scope of quality criteria related to Kesh (1995) and Moody (1998)

In this context, the three criteria of integrity, simplicity and flexibility cover the aspects that quality represents. A further criterion proposed in this research, which is not present in previous work, is the criterion of seamlessness: does the object-relational database provide a seamless transition from objects to their database representations. This reflects one of the key motivations for the introduction of O-R structures, to address the impedance mismatch between object-oriented applications and relational databases. Hence, another evaluation of the quality of object-relational database designs is the extent to which the impedance mismatch is reduced. Having proposed this set of four criteria it is necessary to have clear definitions of the terms used, so the following section explains what is meant by the criteria in this context.

2. Criteria descriptions

This section will describe the four criteria proposed in Fig. 1, as they will be used to judge object-relational designs. Section three will examine ways in which the criteria could be assessed.

Simplicity

> **"Everything should be as simple as possible, but not simpler." Albert Einstein (1879-1955)**

An argument for the use of simplicity as an indication of quality is made by Moody (1998) who draws on research to show that a simpler model is more flexible, easier to implement and easier to understand. These views are supported by research from Piattini et al (2001) and Genero et al (2002), linking the structural complexity of an ER model with two quality indicators: understandability and modifiability (flexibility). Object-relational designs, which might include features such as user-defined types, type hierarchies, composite types and REF types, should be as simple as possible, while not losing the semantics of the model. However, the variety of structures available increases the potential complexity. Although simplicity is insufficient alone to guarantee quality, it is one of the indicators of the quality of an object-relational database design.

Integrity

Moody (1998) defines integrity as the preservation (in the database) of the semantics of the conceptual model. Object-relational databases offer new structures to represent those semantics and, although the O-R structures may be less familiar to designers, an indication of the quality of an O-R design will be the preservation of the semantics of the conceptual model represented in a UML class model. This definition of integrity is wide in scope and would contain the three criteria from Kesh (1995) of suitability, soundness and consistency. Integrity is essential as an aspect of quality that would be of interest to designers of object-relational databases and, for this work, the definition from Moody (1998) will be used: when looking at different design choices the one judged to be best in terms of integrity is the one that best preserves the semantics of the conceptual model.

Flexibility

If the database requires major reworking to accommodate a small change of structure this reflects badly on the quality of the design. Flexibility (or modifiability) is an indicator of a quality database design over a period of time. The criterion of flexibility is included here and is defined, in this context, as the ease with which the design can accommodate change. When the criteria of integrity and simplicity are used to assess the quality of an object-relational database they deal with a static view. An object-relational database could not be considered to be of high quality if only the static view was considered.

Seamlessness

One reason for a designer to use O-R features would be to make the database design closer to the object-oriented analysis product that it is derived from, for example a UML class model. If a suitable design could be produced using simple relational tables, then that might be considered preferable, certainly in terms of maintainability, as the relational model is simpler. The UML class model is often used to model the persistent data in a system and can be taken forward to become a conceptual database design, as shown by Ambler (2002). However, UML class models may contain aggregation, hierarchy and composition, which cannot be directly implemented using relational tables, but could be implemented using O-R features. One of the key motivations for the introduction of O-R structures into SQL is to address the impedance mismatch between object-oriented applications and relational databases. Seamlessness is at the heart of the motivation for object-relational databases, with the reduction of the impedance mismatch as a key aim.

One way to arrive at a definition of seamlessness is to examine other processes that are described as seamless. The mapping from an ER diagram (for example, using crow-foot notation) to a relational model could been described as seamless because of the one-to-one mappings; one entity maps to one table and one attribute on the ER model maps to one column of the table. So the one-to-one nature of a mapping could indicate its seamlessness. The further away from a one-to-one mapping indicates a moving away from seamlessness. This rule could be applied when assessing a mapping, with some questions formulated to establish the relative seamlessness.

Another way to approach seamlessness is to say that a mapping must be reversible. From the ER to relational mapping it is possible to say that the relational schema could be reverse-engineered to arrive at the ER model. The definition of seamlessness used in this context will be a combination of these factors: a mapping from an object-oriented class model to an object-relational database would be considered seamless if there were simple mapping rules (as near to a one-to-one mapping as possible) and that the mapping was reversible.

3. Ways to assess the criteria

When comparing different object-relational designs the proposed set of criteria will be used to establish which of the designs are of better quality. However, when making judgements based on the criteria there is a danger of losing objectivity. The judgements, based on the criteria, must to be valid, reliable and consistent and this can only be claimed if judgments against the criteria are objective, so that they could be reproduced. This section addresses the objectivity of the judgements and how they can be assured by finding ways to test the criteria. Focussing on this differentiation, a questioning or interrogatory approach can be used to assess the quality of the options where more quantitative measures are unsuitable or unavailable.

Simplicity

From a software engineering perspective measures of complexity have been developed that determine the complexity of a system according to the "*number and variety of elements and variety of relationships between them*" (Serrano et al, 2007). Simplicity assessments are based either on understandability, as in Moody (1998) or on counts of the various elements in a design, as in Piattini et al (2001). The assessment of understandability of a model, as described by Moody (1998) required reviewers to evaluate a design. This approach introduces problems in assuring the objectivity of the evaluations and the experience and qualification of the reviewers. The characteristic of simplicity, in this context, is an assessment of the relative complexity of object-relational database designs, so the software engineering metrics approach would be sufficient and appropriate. There remains the question of the "*elements and relationships between elements*" that could be counted for an object-relational design. The language of definition of the object-relational database design is SQL:2003, so the elements that could be counted are:

- Number of SQL statements
- Number of terms
- Number of sub-clauses

These measures are useful for SQL statements such as CREATE TYPE, as the complexity of the structures is related to them. It would also be useful in statements for populating tables, as for example, an INSERT statement that contains an embedded SELECT clause to retrieve a record from another table, would increase the complexity. This gives us another element that increases complexity:

- Level of nesting of sub-clauses

These simplicity measures could also be useful in measuring the complexity of retrievals, although there are always different ways in which data can be retrieved and it is important that these are considered. One aspect that is important to bear in mind is that this research is interested in the relative complexity of one model against another and not an absolute measure.

Integrity

It was stated previously that, when looking at different design choices, the one judged to be best in terms of integrity is the one that best preserves the semantics of the conceptual model. To assess whether the semantics of a design artefact have been preserved first we need to know what these semantics are. The conceptual model used for these comparisons is defined

in a UML class model. This class model should capture the semantics of interest, although UML has its critics, such as Berenbach (2004), who argue that the semantics of the class model are ambiguous. However, for the extracts from a class model, as used in these comparisons, the UML model will be supplemented with descriptions of business rules to provide sufficient semantic clarity, where needed.

When assessing the relative quality of different design choices, the integrity aspect can be shown by assessing:

- the number of semantic elements that are fully preserved
- the number that are partially preserved
- the number that are not preserved.

By looking at these three questions a clear distinction can be made between the O-R database design options.

Flexibility

Flexibility is the ease with which the design can accommodate changes in structure and processing. A taxonomy of change for object-oriented schemas was developed by Banerjee et al (1987) and used to analyse the different types of change that would be possible. In a technical report, Barry and Duhl (2001) produced a similar list of changes that should be accommodated in object storage. It is possible to adapt these schema change taxonomies to analyse the change that would be possible in an object-relational design. Fig. 4 shows a taxonomy developed from those of Banerjee (1987) and Barry and Duhl (2001), but tailored to the changes relevant to object-relational databases. It described the types of change that could be expected in a class model used as the basis for an object-relational database.

1. Changes to classes
    a. Add a new class
    b. Drop an existing class
    c. Change the name of a class
2. Changes to associations between classes
    a. Add a new association between classes
    b. Remove an association between classes
    c. Change to the maximum participation of an association
    d. Change to the minimum participation of an association
3. Changes to attributes of a class
    a. Add a new attribute to a class
    b. Drop an existing attribute from a class
    c. Change the name of an attribute of a class
    d. Change the defaults value of an attribute
    e. Change the constraints of an attribute
    f. Change a single-valued attribute to a multi-valued attribute
    g. Change a multi-valued attribute to a single-valued attribute
4. Changes to generalization hierarchies
    a. Add a sub-class to a class hierarchy
    b. Add a new superclass to an existing class
    c. Remove a superclass from a generalization hierarchy
    d. Partition a class into two classes in a hierarchy
    e. Coalesce two existing classes into one class

Fig 4. Taxonomy of object-relational change

The taxonomy in Fig.4. deals with all the changes to the data structure that could be expected in a class model. However, all these changes are not equally likely to occur. We assume that the original conceptual model provides a "correct" starting point, in the sense that it is a complete representation of the system requirements, so we would not expect the wholesale 'refactoring' type of change described by Ambler (2007) to take place. So although we could use this taxonomy to assess whether a particular mapping can accommodate these types of changes to the model, we also need to consider the likelihood of the changes occurring and the impact of the change, in effect, a risk assessment of the change possibilities.

Seamlessness

As previously stated, a mapping from an object-oriented class model to an object-relational database would be considered seamless if there were simple mapping rules, firstly looking at the number of elements in the conceptual model and the number in the O-R database with a one-to-one mapping indicating seamlessness. The further away from a one-to-one mapping indicates a moving away from seamlessness. The second indication of a seamless mapping is that the mapping is reversible. The distinguishing levels for seamlessness are:

- Is this a one-to-one mapping from the conceptual model?
- Is this mapping a simple one-to-many, where the many side is a predictable number?
- Is the mapping reversible?

The answers to these questions would determine the relative level of seamlessness when considering different options.

4. Discussion

The four criteria of integrity, flexibility, seamlessness and simplicity have been proposed to assess the quality of object-relational database designs. Having considered each criterion in turn it is now useful to examine the set of criteria as a whole to review their coverage of the design quality from different perspectives and to address the issue of ways in which the criteria themselves can be evaluated.

Quality criteria to assess a database design from different viewpoints and over time

Moody (1998) categorises quality criteria in terms of the viewpoints of different stakeholders in the design; business users, data analysts, data administrators and application developers. Although the scope of this work does not consider the integration and implementation of the database design it is still necessary to consider the database design from viewpoints other than the designer's.

- At the analysis and design stage the **business analyst** would consider the most important aspect to be *integrity* (preserving the semantics). The business analyst would require that the model has integrity but would be less concerned with the other criteria.
- Moving towards the implementation stage, from the viewpoint of the **database administrator**, *simplicity* would be seen as the most important factor.
- Once the database is in use, the viewpoint of the database administrator is still important but another viewpoint might be considered: the **application developer**. The application developer, if coming from an object-oriented programming environment, would view *seamlessness* as the most important factor.
- Over time, when changes to the design are likely, from the viewpoints of the **database administrator** making the changes, *flexibility* becomes an important factor.

- From the viewpoint of the **systems analyst** the most important aspects in the early stages would be *integrity* but they would also take into account the other considerations of *seamlessness* and *simplicity*. When changes are needed to the data structure over time, *flexibility* also becomes an important factor.

Criteria / Role	Simplicity	Integrity	Flexibility	Seamlessness
Systems Analyst	✓	✓	✓	✓
Business Analyst		✓		
Application Developer				✓
Database Administrator	✓		✓	

Table 1

These different viewpoints and their concern with particular priorities in the design are summarized in Table 1. By including all four criteria, coverage over the lifetime of the design is more likely to be achieved than by focusing on one criterion alone. Other viewpoints could be developed and any stakeholder in a database design could assess their own priorities using a combination of criteria.

Evaluating the quality criteria

To be confident in using these criteria we must be sure that they are reliable and valid. It is proposed that following an experimental period of employing the criteria to assess object-relational database designs, their use will be evaluated so that the criteria might be shown to be valid and reliable. A definition of validity is provided by Moody et al (2003) where validity is assessed using three measures: completeness (do the measures miss anything), parsimony (are all measures needed) and independence (are the measures independent of each other or do they overlap). These are useful measures that will be used to assess the validity of the quality criteria.

Definitions of reliability indicate that its meaning can be interpreted as certainty, dependability, predictability and consistency. When evaluating reliability it is anticipated that the main problem will be in interpretation. Moody (2003) discovered that terms used for evaluation of quality can be misinterpreted by the evaluators. To mitigate against this tendency, the meaning of the criteria must be explained in a way that is unambiguous. This will be facilitated by the use of extensive examples of the usage but, clearly, the ambiguity of the criteria must be examined after they have been used.

5. Conclusion

Criteria are needed to help us assess whether a particular design displays the characteristics of quality, as shown in Fig.1. Too many criteria could prove confusing and difficult to apply so a minimal set of four criteria have been proposed here: integrity, simplicity, flexibility and seamlessness. Fig. 1 shows that to have a quality design, from the ISO 9126 (International

Standards Organisation, 2001) model there are three quality characteristics that are necessary: functionality, usability and flexibility.

The first characteristic is functionality: does the design represent correctly the persistent data requirements from the conceptual model. This is the view of the design from the roles of business analyst and systems analyst and is demonstrated in the criteria of the seamlessness of the mapping from a conceptual model to a design and the integrity, or how the semantics of the design are carried forward. A potential loss of semantics arises when designs change the underlying model. Only a truly seamless O-R design can carry forward all the semantics of the conceptual model.

The second quality characteristic is usability: whether the design can be used effectively. The key viewpoint to be considered for this characteristic is that of the application developer. The aspect that the application developer would be concerned with is the seamlessness of the database, from the application development environment. If that environment is object-oriented, then the seamlessness of the database implementation comes to the fore.

The third quality characteristic is flexibility, which requires a design that can cope with change. Flexibility is the characteristic of a design that addresses the development of the model over time. It may be counter-intuitive but a more complex model may be able to cope with change better than a simple one. The element of flexibility is an aim of object-oriented designs, but it can be applied to database design using object-relational features. The viewpoints of the systems analyst and the database administrator are most important when considering the characteristic of flexibility.

The four criteria of integrity, flexibility, seamlessness and simplicity have been proposed to assess the quality of object-relational database designs. Often a metrics based approach equates quality with simplicity, but simplicity is insufficient alone to be counted as a measure of quality. While simplicity is desirable, a simple design can hide semantics that would make the insertion, update and retrieval of data more difficult. We require that our design can be used effectively. Clearly simplicity has been demonstrated as a characteristic that affects the usability of a design (Piattini, 2001), but seamlessness will also help with understanding, particularly with those who are used to working with an object-oriented design.

Future work will address the issue of whether these criteria are sufficient to distinguish between good and bad O-R designs. The criteria will be used to assess the quality of object-relational database designs. Once they have been used, the criteria they will be revisited to examine whether they have been useful in deriving judgements about the overall quality of the designs. Furthermore, the criteria will be assessed to find whether they could be useful in a wider context.

References

Ambler, S.W., (2002), A UML Profile for Data Modeling, available at http://www.agiledata.org accessed on 17/2/04

Ambler, S.W., (2007), The Process of Database Refactoring , available at http://www.agiledata.org/essays/databaseRefactoring.html accessed on 7/12/07

Banerjee, J., Chou, H-T, Garza, J.F., Kim, W. Woelk, D., Ballou, N. and Kim, H-J., (1987), Data Model Issues for Object-Oriented Applications, TOOIS (Transactions on Office Information Systems)

Bansiya, J. and Davis, C. G., (2002), A Hierarchical Model for Object-Oriented Design Quality Assessment, IEEE Press, Piscataway, NJ, USA, IEEE Transactions on Software Engineering, Volume 28, Issue 1, pages 4-14

Barry, B. and Duhl, J. (2001), Object Storage Fact Book 5.0, Object DBMSs, Barry & Associates, Inc.

Berenbach, B., (2004), The Evaluation of Large, Complex UML Analysis and Design Models, (ICSE'04) 26th International Conference on Software Engineering

Genero M, Poels G. and Piattini M., (2002), Defining and Validating Measures for Conceptual Data Model Quality, Advanced Information Systems Engineering: 14th International Conference, CAiSE 2002 Toronto, Canada, May 27-31, 2002

International Standards Organisation (ISO), (1999) ISO/IEC 9075:1999 SQL standard definition, available from http://www.iso.org

International Standards Organisation (ISO) (2001), ISO/IEC 9126-1:2001: Software engineering. Product quality. Part 1: Quality model, available from http://www.iso.org

International Standards Organisation (ISO), (2003) ISO/IEC 9075:2003 SQL standard definition, available from http://www.iso.org

Kesh, S., (1995), Evaluating the quality of entity relationship models, Information and Software Technology, Vol. 37 (12), p. 681-689

Moody, D.L., (1998) Metrics for Evaluating the Quality of Entity Relationship Models, ER 1998, p.211-225, Singapore, China

Moody, D. L., (2005) Theoretical and practical issues in evaluating the quality of conceptual models: current state and future directions, Elsevier, Data and Knowledge Engineering, vol. 55, p243-276

Piattini, M., Calero, C., Sahraoui, H. and Lounis, H., (2001), Object-relational database metrics, L'objet, March 2001 available at http://www.iro.umontreal.ca/~sahraouh/papers/lobjet00_1.pdf accessed on 31/3/06

Serrano, M., Trujillo, J., Calero, C. and Piattini, M. (2007), Metrics for data warehouse conceptual model understandability, Information and Software Technology 49 (2007) p 851-870, Elsevier