# ERRORS AND ANOMALIES IN OBJECT ORIENTED EXPERT SYSTEMS: A SHORT REVIEW

*Frans Coenen*
Department of Computer Science The University of Liverpool L69 7ZF

## 1 BACKGROUND

The object oriented concept has been used, in one form or another, to represent knowledge in *Knowledge Based* and *Expert* software Systems since the late 60s. The earlist examples can be traced back to Quillian's "Semantic Networks" [9], and progress through "frame based" systems [6] and knowledge representation languages such as KRL [3], to the OO features that we find in current "expert system" languages such as CLIPS [5].

The validation and verification of Expert Systems became an issue from the late 1980's onwards with much research effort devoted to methods to prevent, detect and correct errors and anomalies in Expert Systems. Much of this effort was directed towards systems in which the knowledge was encapsulated in a rule format — *rulebased* systems, and resulted in a prolifireation of "rule checking" programs; well known examples include Suwas's rule checking program which was implemented on the ONCOCIN system [?], CHECK [?], EVA [4] and COVER [8].

At the same time many researchers produced lists of "errors and anomlaies" that need to be addressed when attempting to verify and validate expert systems [2, 8, ?]. Much less attention was devoted to frame based systems. One exception can be found in the work of Coenen and Bench-Capon [1] who, although concentrating on the maintenace of expert systems, give a list of "errors and anomalies" found in frame systems on the lines of the many similar lists found in the context of rule based systems. This list can (almost) be translated directly into OO terms as follows:

- Redundancy
- Missing attributes and classes
- Misplaced attributes and classes
- Duplication
- Auxiliary attributes and classes
- Inconsistency and incompleteness
- Errors related to numeric ranges
- Errors associated with cardinality

Each of these is disscussed in some further detail in the following sub-sections. Errors that might be associated with methods, i.e. rules to manipulate the knowledge, are similar to those identified for more traditional "rule-based" systems.

### 1.1 Redundancy

The concept of redundancy in frame based systems is similar to that for rule-based systems. An attribute or class is redundant if it is not used to establish anything that the system is designed to address. It can be argued that classes, other than the "top-class", which are not extensions of the top-class are redundant, since they cannot be linked with the rest of the application. However this will, in practice, be more an indication of the presence of missing classes or connections rather than ture redundancy.

## 1.2 Missing attributes and classes

A OO knowledge representation will contain missing slots if we cannot uniquely identify each class. There are a number of possible situations where this might occur. Firstly a class which is a sub-class of another may not contain an appropriate attribute to distinguish it from its super-class. Alternatively we may havec two identical frames (duplication), not in the same path in the hierachy. In such a case we will not be able to distinguish between the two or sensibly to assign instances to one rather than the other. This will then hamper the operation of any reasoning algoritghm which we might wish to superimpose over the representation since the dupluicates will have different super-classes and sub-classes.

In both cases the situation can be remedied by introducing an additional attribute into one or other the classes so that they can be distinguished. alternatively, in some cases, an existing attribute may be altered, for example the value that it requires may be changed.

An OO reasoning system will, of course, contain a missing class if a class is defined as a sub-class of a class that does not exist. In practice most OO developmemt environments/compilers will prevent the creation of classes which have a non-existent super-class; therefore this situatiion will not usually arise.

## 1.3 Misplaced attributes and classes

Given the property of inheritance the location of a slot in a frame hierarchy can be significant. We can envisage situations where a slot is introduced at a level either too specialised or not specialised enough. Attributes should be introduced into the class hierarchy at the highest point where they will be of use to instantiate an instance of a particular class.

A similar situation exists where the sub-class and super-class relation between two classes is inverted. The situation also exists where the hierarchy displays circularity, i,e a class is defined indirectly as being a sub-class (or super-class) of itself. This latter situation is agian unlikely to occur in practice as most OO environments will not allow this construction.

It is worth noting here that OO knowledge representatuions enable errors in recording knowledge about the *ontology* of the domain to be recognised more easily and at an ealier stage than do rule based representations where the ontology tends to "emerge" from the rule-base as a whole.

## 1.4 Duplication

The concept of subsumtion (a common annoamly found in rule-based systems) does not exist as such in OO systems. The situation does exist, however, where attributes and/or classes are duplicates of one another. We have already discussed the effect of the precence of duplicate classes (see above). Where a class hierarchy contains duplicate attributes in different classes with a common super-class, this may indicate that the attribute should have been introduced at a higher level and inherited down (or some other flaw in the structure of the hierarchy). Many OO environemnts carry out a number of checks on the class hierarchy (usually when it is compiled). During this process duplicate slots are often detected and the developer advised to this effect.

## 1.5 Auxilary classes and attributes

Auxilliary classe are included in a class system to reflect the domain which the system is intended to model (as is also the case with auxiliary rules found in rule-based systems) even though they do not contibute in anyway to the effectiveness of the resoning process. Similarly individual classes may include auxiliary attributes simply to reflect the doamin. As in the case of auxiliary rules, auxiliary classes and attributes are not considered to represent an error, since they can have a role in more accurately describing the domain and provide the basis for future extensions.

### 1.6 Inconsistency and incompleteness

It is hard to give a "clear cut" analysis of the notions of inconsistency and incompleteness in the context of OO knowledge representations (unlike the case with respect to rule-base systems). This is because much of the inference that can give rise to these problems is often associated with the *inference engine* designed to manipulate the classes rather than the classes themselves. If, however, there is a requirement for classes to be exclusive, so that an object can only be an instance of a single class, an inconsistency (or incompleteness) would occur if there exists a possible set of values that would allow an object to be an instance of more than one class. This is more acceptable if the classes in question are in a super/sub-class relation, since the higher instantiation is compatible with the lower and thus can be ignored. If, however, the classes lie on different paths through the hierarchy we either have an inconsistency, if the object is instantiated to both classes, or an incompeteness if we instantiate to neither.

### 1.7 Errors associated with lower and upper numeric ranges

In some OO knowledge representation languuages/environments particular ranges of values that an attribute may take can be specified, and be used to validate data supplied for such attributes, and to perform a variety of inferences. It is important, however, that these restrictions are applied in a consistent way and that the range of the slots for lower level frames are strict specialisations of the ranges they inherit; and that ranges between different slots are consistent.

### 1.8 Errors associated with minimum and maximum cardinality

Cardinality in OO knowledge representations refers to the number of instances that may be associated with a particular slot. For example in a social security system there may be an attribute representing the applicant's spouse which may have a cardinality of one on the assumption that the applicant can have only one spouce at any single moment of time. Again care must be taken that these are of universal applicability rather than simply the nornal case (marriages can be polygamous!)

## 2 CONCLUSION

In this short report an overview/categoristaion of the errors and anomalies that can be found in OO knowedge representations has been presented. It is these errors and anomalies that any OO Expert System V&V system must address.

It is also interesting to note that these errors and anomalies are applicable to OO systems in general and not simply OO representations of knowledge intended for use in reasoning systems.

## References

[1] Coenen, F.P. and Bench-Capon, T.J.M. (1993). Maintenance of Knowledge Based Systems: Theory, Tools and Techniques. Academic Press, London.

[2] Cragen, B.J. and Steudal, H.J. (1987). A Decision-Table-Based Processor for Checking Completeness and Consistency in Rule-Based Expert Systems. International Journal of Man-Machine Studies, Vol 25, pp633-648.

[3] Bobrow, D.G. and Winograd, T. (1979). KRL: Another Perspective. Cognitive Science, 3(1).

[4] Chang, C.L., Combs, J.B. ans Stachowitz, R.A. (1990). A Report on the Expert Systems Validation Associarte (EVA). Jo. of Expert Systems with Applications, Vol 1, No 3. p219-230.

[5] http://www.jsc.nasa.gov? clips/CLIPS.html.

[6] Minsky, M. (1975). A Framework for representing Knowledge. Winston. P.H. (Ed.), The Psychology of Computer Vision, McGraw-Hill, New York, pp211-77.

[7] Nguyen, T.A., Perkins, W.A., Laffey, T.J. and Pecora, D. (1985). Checking an Expertt System's Knowledge Base for Consistency and Completeness. Proc. IJCAI'85, pp375-378.

[8] Preece, A.D. and Shinghal, R. (1991). COVER: A Practical Tool for Berifying Rule-Based Systems. Knowledge-Based Systems Verification, Validationand Testing, Workshop Notes from the 9th National Conference on Artificial Intelligence, AAAI-91, Anaheim, CA.

[9] Quillian, M.R. (1968). Semantic Memory. In Minskey, M., Ed., Semantic Information Processing, MIT press, Cambridge, MA, pp227-70.

[10] Stachowitz, R.A. and Combs, J.B. (1987). Validation of Expert Systems. Proceedings of the 20th Annual Hawaii International Conference on System Science, pp686-695.

[11] Suwa, M., Scott, A.C. and Shortliffe, E.H. (1984). Competeness and Consistency in a Rule-Based System. In Buchanan, B.G. and Shortliffe, E.H. (Eds.), Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project, Addison-Wesley.