Transactions on Information and Communications Technologies vol 7, © 1994 WIT Press, www.witpress.com, ISSN 1743-3517

Object oriented design: a teaching environment

K. Whiteley, M. Merabti
School of Computing and Mathematical Sciences, Liverpool John Moores University, Liverpool, UK

1 Abstract

Although current Case tool technology offers a range of Object Oriented analysis and design methods, the cost and complexity of such systems often precludes their use in an academic environment. This paper describes the structure of a simple tool for introducing Object Oriented methods to students studying on a degree in Software Engineering. The tool offers two complete phases of the design process. Initially, in conjunction with a component database, the user is able carry out a preliminary identification of objects and operations. These candidate objects can then be assessed against criteria for a "viable" object. Following these stages, an object may be created through a simple form based method.

2 Introduction

The increased availability of computer equipment for student use in a modern academic institution, together with improving standards of basic computing power have naturally led to a move from more traditional teaching modes to techniques involving machine based methods. In addition, teaching staff in these institutions have had to deal with the pressures caused by increasing student numbers which have occurred over the past few years (1). This pressure has also increased the need to provide students with "teacher-independent" learning environments. In the area of Computer Science related subjects, these systems have ranged from large scale funded projects, ultimately intended for general adoption, to more modest locally developed systems. The scope of these systems has also covered a number of purposes, for example from computer based tools for teaching programming languages and concepts, to tools for developing the electronic material for machine based learning systems (2).

The tool described in this paper has been produced to provide students with a mechanism for developing their understanding of the basic concepts in Object Oriented Design. Its use is in the context of a BSc Honours degree in

Software Engineering and is planned to form part of their introduction to the principles involved in object oriented software development. Its structure is based on the techniques currently used in the teaching of these concepts and brings together the paper based and software based methods currently used on the course (3) into a single computer based system.

The module in which these techniques are used is one entitled "Object Oriented Systems" which forms one of the core set of modules in the Software Engineering degree. This module introduces students to the concepts of object oriented software development in the context of a more conventional course in C++. The relationship between the specific object oriented concepts taught in this module to the wider ideas of object oriented development found throughout the course can be best understood by looking briefly at the structure of the degree itself.

3 The Software Engineering Degree

The Software Engineering degree is a semester scheme over three levels of study, with one year (the third) as an industrial placement. The structure of the degree is made up from three major "themes" of study, namely :-

Programming systems : the study of software systems, their programming languages, methods and tools

Software Engineering Methodologies : the study of the tools, techniques and methods used in the development of quality software

Formal Methods : The study of the use of formalism in the specification and design of software

These themes are taught alongside more general aspects of computing from other courses in the School's programs and include topics such as Computer Systems, Data Structures and Database Design.

The development of object oriented concepts in the course takes place through the study of specific topics at each of the three levels. The major contributors to this development are the following modules on the course :-

Level I	Software Engineering with Ada : this module introduces the Ada language as a software engineering "tool". In addition to the fundamentals of Ada, the module emphasises features of the language, such as abstraction, which underpin concepts important in the software engineering process.
Level II	Object Oriented Concepts : this module provides an introduction to the ideas of object oriented design and programming - taught using C++ as the programming language. This course is taught mainly through the use of a number of case studies during which object oriented analysis, design and programming issues are pursued.
Level III	Real Time Systems : In addition to looking at issues specific to real time software, students follow a course in a specific object oriented methodology (currently HOOD is used as the chosen method)

In addition, a number of "support" studies are undertaken to consolidate the ideas presented in the main module. The most important support material comes from the study of database design, which provides the necessary ideas of entity-relationships required for looking at many object oriented design issues. Furthermore, students are given the opportunity to gain practical skills and an in-depth understanding of the issues in the "follow on" Applications Workshop (4). The relevant components of the course and the way in which they relate to the central module are shown in Figure 1 below.

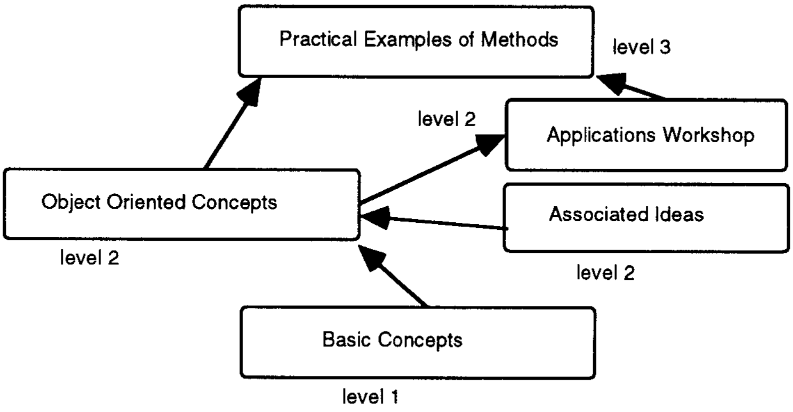

Figure 1: The role of Object Oriented Concepts within the degree

The diagram shows that certain key ideas which are important in the understanding of object oriented techniques (for example ideas of entity-relationships) occur in parallel with the object oriented systems module itself and consequently cannot be fully utilised in that course. The method of object oriented design presented in the course, therefore, must be based on simply understood (and realised) concepts which do not rely on these more "advanced" ideas. It should, however, represent a realistic method which would allow these ideas to be incorporated when appropriate.

Whichever method is chosen must allow an effective realisation of the object model, which is central to object oriented techniques. The method chosen should allow the designer to capture the essential structure of an object and enable the interaction of the object with others to be studied.

4 Responsibility Driven Design

One of the problems inexperienced learners face in the understanding of the object model and its implementation is that of visualising the overall structure of the system and the complexity of the inter-object interactions. One solution that has been found successful in showing the interaction between objects is "Responsibility Driven Design" (5). This approach allows the designer to see the limits of the object's responsibility, and consequently the structure of the control threads within the system. This approach has been complemented by a user interface that allows the construction of Class, Responsibility and

Collaboration cards (CRC cards) (6) which allow the capture of the structure of an object. The use of CRC cards presents a high level representation of the system under construction independent from the implementation language. A related method has also been successfully used in real life projects (7).

Individual CRC cards contain the following fields :-

Responsibilities	This is a list of the actions carried out by the object
Collaborators	Other objects which interact directly with this object
Attributes	Data components which characterise the object

The complete system is represented by a network of individual cards. As an example of how a system can be represented by CRC cards, consider a simple system in which a user has access to a set of basic components through the presentation of a menu. These components are used to construct a composite object, which may be either purpose made, or selected from a database of "ready made" composite objects. Analysis of the system would result in the following candidate objects :-

Presenter	displays the menu to the user
Component Database	holds the basic components
Composite Database	holds the definitions of the composite components
Constructor	assembles the composite object
Component	the component constructed

The CRC card for (say) the Presenter object would contain elements :-

Responsibilities	displays initial message displays a menu of options pass control to Component Database pass control to Constructor
Collaborators	Component Database Constructor
Attributes	Messages Menus

Based on this form of analysis, the system network is shown in Figure 2 below.

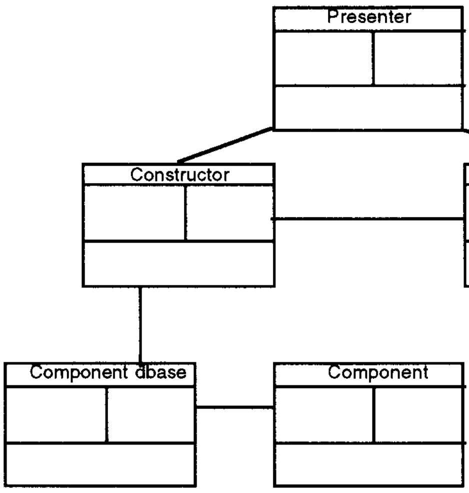

Figure 2: Sample CRC card network

The stages in the analysis and design of the system using this design method are as follows :-

1. Define the system specification
2. Identify the objects
3. Identify the attributes
4. Complete the CRC cards
5. Produce the Network
6. Reconcile the design

The tool supports these stages of analysis through a simple form-based user interface. It also allows access to a database of components from which objects may be selected. Objects created by the user can also be assessed informally against the characteristics of viable objects.

5 A Tool for Object Oriented Analysis and Design

The main aim of the tool is to support the teaching of Object Oriented techniques in the Software Engineering degree. The features provided are as follows :-

- The introduction of a problem statement, with appropriate edit, file and associated facilities

- The analysis of this statement in order to establish a list of candidate objects and attributes

- Operations on a component database, including viewing of components, editing of components and inclusion of new components

- Creation of objects through interaction with a CRC card
- Production of a system network diagram

The structure of the tool is shown in Figure 3 below.

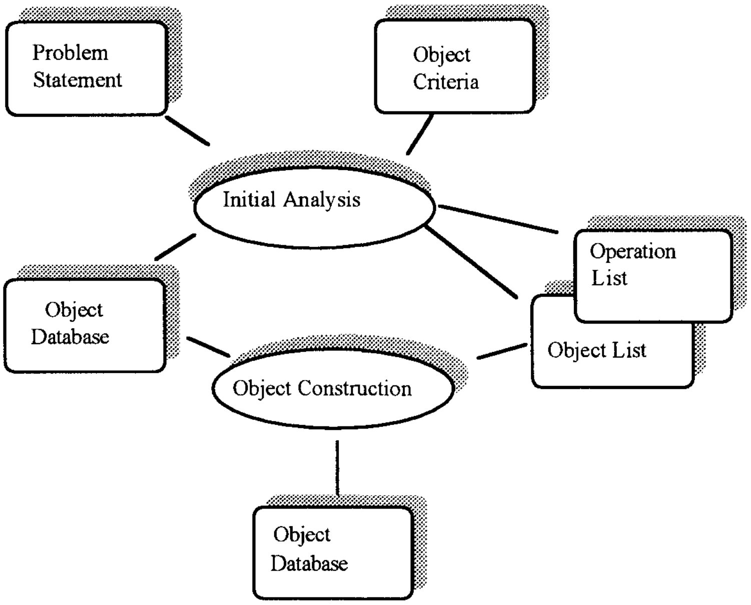

Figure 3: The structure of the Tool

The software has been written using Visual C++. Interaction with the tool takes place through dialogue boxes accessible from the main screen menu options. Dialogue boxes are available for :-

- Entry of Initial Problem Statement
- Selection of Candidate Objects
- Selection of Attributes
- Viewing the Component Database
- Database manipulation

In addition, a form based screen is used for the creation of an object, allowing the user to add and delete operations and attributes. This screen also provides the user with a help feature in order to allow them to assess the viability of the object they are constructing. The object construction screen is shown below in Figure 4 and the Problem Analysis screen in Figure 5.

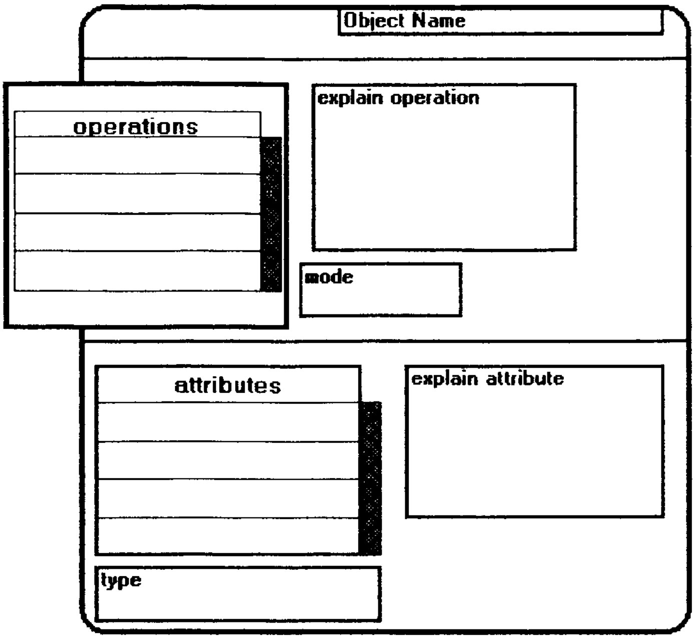

Figure 4: The Object Construction Screen

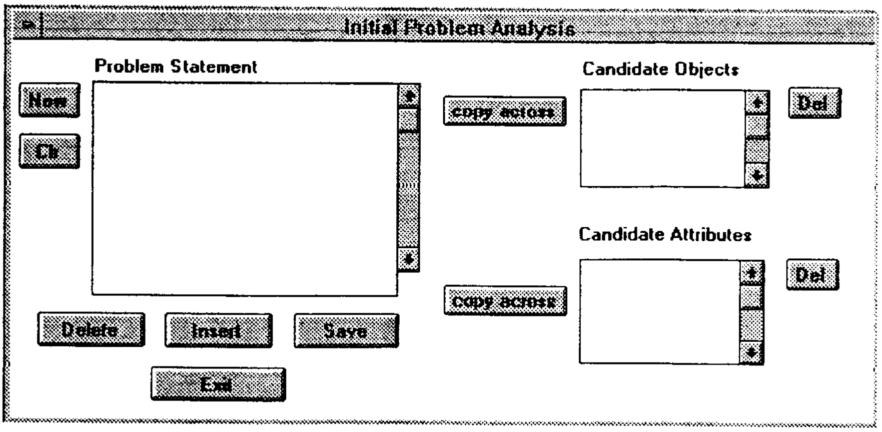

Figure 5: The Problem Analysis Screen

In the current implementation, the assessment of object viability is carried out informally through use of a help feature. This help feature provides details of the characteristics of a "good" object by presenting the criteria proposed by Coad and Yourdon (8).

6. Conclusions

The CRC based system has found to be a useful way of introducing Object Oriented analysis and design methods. It provides the student with an understanding of the principles involved and experience in practising the techniques. The development of this tool seeks to automate the processes involved and allows the user to explore the method in an independent manner. Combining the text based material from the course, particularly the sample case studies, enable the student to carry out the analysis and design of relatively modest, but realistic examples. It also enables the student to appreciate the strengths and weaknesses of the method.

The incorporation of a component database introduces the concepts of software component reuse and illustrates the ability of Object Oriented methods to assist in the process of reusability and in addition, highlight the problems of lack of standardisation of reusable objects.

It is planned to extend the capability of the tool in the following areas :-

- skeleton C++ code generation from an object specification
- extension of the reuse capabilities
- extending the effectiveness of the help guidelines for assessment of objects

7 References

1. Bornat, R., Key Note Talk: Grabbing the Carrot, *Development in the Teaching of Computer Science*, Bateman, D. and Hopkins, T., pp. 1-9, University of Kent, UK, April 1992.

2. Whiteley, K. and Merabti, M., A Toolset for Open Learning Development and Delivery, *29th Annual International Conference of AETT, Computer Assisted and Open Access Education,* Napier University of Edinburgh, April 1994.

3. Merabti, M. and Whiteley, K., OZône: Object Oriented Systems in an Open Learning Environment, Tattoo'94, Leicester, 1994.

4. Merabti, M. and Bamford, C., Software Engineering: from Theory to Practice, Software Engineering in Higher Education, Southampton, 1994.

5. Wirfs-Brock, R. and Wilkerson, B., Object Oriented Design: A Responsibility-Driven Approach, *Proceedings of OOPSLA'89*, pp71-75, 1989.

6. Beck, K. and Cunningham, W., A Laboratory for Teaching Object-Oriented Thinking, *Proceedings of OOPSLA'89*, pp 1-6, 1989.

7. Merabti, M., private communication.

8. Coad, P. and Yourdon, E., *Object Oriented Analysis*, Prentice Hall, 1990.