

Computer based evaluation: myths, methods and morality

J.F. Coxhead, M.B. Dixon, E.A. Dodman

*Software Environments Research Group, Leeds
Metropolitan University, Beckett Park Campus,
Leeds, LS6 3QS, UK*

Abstract

The use of computer systems to aid in the evaluation of trainee software engineers is a natural step due to the nature of the topic. There are different viewpoints which surround the use of evaluation covering areas such as practical possibilities, techniques for implementation and the reaction of individuals involved. This paper discusses each of these areas in turn by outlining the expectations of such a system, offering a design solution with practical background and concluding by examining human factors.

1. Introduction

The education of software engineers must include a considerable amount of computer based work. The practical application of implementation techniques is traditionally the bulk of this type of work. With the advent of CASE tool availability however almost all areas of software engineering training and development can include the use of computers, with the advantage that tool usage increases student productivity[1]. This predominant use of computer based technology in the application of taught techniques invites the idea of automatically collecting and assessing student work in order to provide computer based evaluation (CBE). Initially this may seem a straightforward procedure; judgement however should be reserved until a more in depth study of the requirements for such a system is made. The effects on teaching practices and student behaviour must be considered as important criteria when measuring the success of a CBE system.

The term computer based evaluation refers to a situation where computer based work undertaken by an individual or group is automatically monitored and/or assessed by the computer system. A distinction should be made between this and the term computer aided learning (CAL), which refers to the technique of using computer based applications in a direct teaching mode. However one of the concepts of CBE is to enhance understanding and teaching through the use of fast objective assessment techniques. The results of these assessments can be used to identify problem areas and allow feedback to take place in a much tighter time scale than with manual evaluation methods.

2. Myths and Expectations

In order to allow the examination of the requirements and associated problems of implementing a CBE system, it is first necessary to provide an overview of what such a system can realistically be expected to achieve. Without an appreciation of what can actually be developed it is difficult to judge whether the requirements have been fulfilled. The capabilities of a CBE system can easily be overestimated. The promise of wonderful and unlimited evaluation power automatically working in the background tends to increase expectations beyond practicality. There are common misconceptions with regards to the use of CBE which must be identified, three of the most important are now described.

2.1 Replacement of Manual Evaluation

Where people have considered the use of CBE there has often been an initial incorrect expectation that it will replace the need for almost all human interaction with regards to assessment. This expectation promotes the view that once a student software engineer has input his or her specification, design and program, the final grade for this assignment will be automatically displayed on the lecturer's terminal.

2.2 Limited Evaluation Capabilities

There is a school of thought that CBE can only evaluate mathematically oriented work to which commonly used numerical metrics can be applied. The idea exists that less mathematically based areas such as natural language specification and diagrammatic system designs cannot be evaluated, therefore limiting CBE to the examination of products such as programs. This means that feedback would be slow if a system development life-cycle was imposed.

2.3 Reduced Manual Effort

The final myth is that using a CBE system will vastly reduce the workload involved in assessing student progress. It is nice to think that the days of spending more time on evaluating work than actually teaching it are gone forever. Since a large amount of evaluation can be performed automatically it is easy to assume that the accurate assessment of students will require little effort compared to manual evaluation techniques.

3. Facts About CBE

The previous section described the major misconceptions which we feel have a warping effect on the expectations of an automated evaluation system. The viewpoints for each of these myths are now discussed from the findings of a feasibility study for CBE in the education of software engineers carried out at Leeds Metropolitan University. These viewpoints more truly reflect a CBE system by taking into account what is practically viable and should present a more accurate picture of what can realistically be achieved.

3.1 Complement to Manual Evaluation

When using CBE it is still necessary to include manual assessment. A CBE system will work in conjunction with manual based evaluation rather than be a full replacement. There are certain situations and factors which cannot be taken into account by a computerised system, hence manual evaluation may be required when it is necessary to evaluate more than just the product produced. For instance it is usual that additional factors such as a persons experience or background must be considered when assessing how well an individual has performed. As another example consider the situation where low marks obtained using CBE could be caused by problems with the student's understanding of natural language rather than difficulties in performing the given work.

3.2 Wide Scope of Evaluation

The use of a CBE system can allow almost all areas of system development to be evaluated. In the past it is true to say that metric techniques concentrated on later stages of system development. Much work however has recently been done on the earlier non-numerically based stages. Areas such as design can now be evaluated using numerous metric techniques (see Yu & Lamb[2] for a review of eight such methods). This continuing research allows CBE to be effective through the full range of modern development practices and not just the implementation stages. With this in mind it is important for CBE system designers to recognise the facilities required to implement new evaluation techniques and not limit assessment to product based analysis.

3.3 Similar Effort Requirements

The overall workload involved when using CBE will not be substantially reduced compared to manual evaluation. Although the assessment process may be faster the actual amount of work going into the evaluation procedure will remain about the same. The main difference is how effort must be applied. There will be a substantial shift in effort to the beginning of assessment where the system must be correctly setup to accurately reflect desired criteria. Once the initial effort has been made the actual evaluation may be performed with a reduced workload. Of course the longer a particular assessment exercise is used the less setting up of a system will be required, and hence a reduction in evaluation effort may then begin to occur.

4. Application Areas

There are various areas of application for CBE. The obvious area is that of simple evaluation by assigning a value to a piece of computer based work by using pre-set criteria. There are however less obvious application areas which could prove to be advantageous.

4.1 Problem Identification

The use of automated evaluation allows problem areas to be identified quickly, especially if making use of real-time feedback. These problem areas could range from basic understanding difficulties of an individual to unfriendly features exhibited by certain CASE tools. Also if any one area of study continues to show anomalous results from all students, then it can be safely assumed that some aspect of that area is deficient.

4.2 Course Design

By collating the results taken from evaluation and examining identified problem areas the design of a course can be changed in order to attack areas of difficulty. More than just course design could be influenced, the selection of such things as tools, methods, teaching techniques and even lecturing staff could be affected.

4.3 Plagiarism Detection

By storing the evaluation information of system users individual profiles can be built. These profiles would reflect the areas in which an individual found difficulty as well as the basic evaluation figures. By comparing these profiles any similarities between users would become apparent, possibly indicating plagiarism. This technique for detection could even work if some aspects of the work appeared quite different!

5. Methods for Implementation

This section describes the requirements and techniques which may be used to implement a CBE system.

5.1 Requirements Overview

The implementation of an evaluation system should support the ideas of flexibility by allowing tailoring to specific requirements and extendibility for future enhancements. Attention must be focused on techniques for the collection of relevant information and the appropriate use of the evaluation information once it has been collated. One of the of the most important areas of consideration when designing an evaluation system relates to the interface between the system and the CASE tools from which evaluation is made. Lack of tool openness and interoperability can cause difficulties and increase the problems associated with evaluation. In order to reduce this problem the underlying environment used in the LMU system is common to both the

evaluation system and the tools themselves. The system referred to in this paper is based on Meta-CASE technology and is described along with rationale for its use by Coxhead (*et al*) [3]. An overview of such an environment is also described by Ratcliffe (*et al*) [4]. The use of Meta-CASE allows a higher degree of co-operation to exist than would be possible when using a non-integrated environment.

5.2 Capturing Evaluation Data

In order for computer based evaluation to take place information on student behaviour must be captured. This information should provide as much evaluation data as possible. To fulfil these needs it is necessary to be able to combine data from the products produced at each stage and from the process involved in developing these products. Data can be collected on the products by examining the database which stores the details of the development. Information on the process can be captured by examining messages which are given to the user during development. Message capture may be achieved by redirecting output to logs which contain all relevant information regarding displayed messages.

5.3 Provisions for Flexibility

There must be some method to instruct the system what information to retrieve and what assessment techniques to apply. The message analysis will be common to all tools, since all tools generate similar classes of messages. The product analysis however is highly dependent on the methods and tools used for development. For example an evaluation of a data flow diagram could involve checking the names of each process against a list of approved action verbs[5], while the evaluation of a program structure chart could involve measuring levels of coupling.

For the evaluation system to judge what is a quality product for a specific development activity a facility must be included which allows quality definition scripts to be read. These definition scripts will be utilised by the evaluation system and will contain instructions for the capture and analysis of evaluation data. Definition scripts also allow the system to be flexible with regards to future tools, methods or evaluation techniques.

5.4 Feedback of Information

Once evaluation information has been captured various methods must be available to allow feedback of the results to the lecturer. There are three areas which should be supported to allow this. Firstly, the ability must exist to archive results for future study (in our experience compacting must be used to manage the large amounts of evaluation data). Secondly, real-time feedback should be included to allow monitoring of group tutorial sessions, allowing problems to be addressed immediately. Thirdly, export facilities should be available to allow figures to be input into third party software packages such as statistical analysis programs or database/query packages.

150 Software Engineering in Higher Education

5.5 Problems Experienced

Work on a prototype CBE system is currently being carried out and the message capture section of the system has already been developed[6]. Certain problems were highlighted during this work with regards to interaction between the evaluation system and the CASE toolsets. The problems included

- Variable date formats from different software
- Problems with messages during exceptions
- Less than 100% message redirection abilities

These may seem minor but are a subtle reminder of the kind of difficulties which appear even when the CASE tools provide the required facilities. These problems primarily relate to the message redirection; the actual assessment of produced work requires specialised additions to be made to the integrated environments. These additions, such as the quality definition script language referred to above, are currently in the specification stages.

6. Moral Implications

Now that the realistic expectations and implementation techniques for CBE have been described it is necessary to examine what human factors and changes to practice should be considered when contemplating using such a system.

6.1 Student Concerns

The primary examination of morals must consider student reaction to CBE. We have conducted various experiments involving closed shop sessions where students were monitored while using integrated project support toolsets. The students were fully aware that they were being evaluated automatically and expressed little or no concern. An important factor of this however was that an assurance was made to all students involved that the assessment information would be kept confidential with no abuse of the results gained from the evaluation procedure.

6.2 System Abuse

It is important to control access to information captured and generated by the CBE system. This is primarily to avoid problems with students confidence in the system due to lack of confidentiality as mentioned above. Abuse may also occur through misuse by lecturing staff or from deliberate tampering by students. In order to protect against this all data and files relevant to the CBE system should have restricted access with password facilities, such as are used for personal accounts on a UNIX system. We are not suggesting that all abuse must be deliberate and in a badly designed system accidental forms of abuse could occur, such as the removing or corrupting of files containing evaluation information.

6.3 Changes to Current Practice

There is no question that changes to current assessment practice will occur with the introduction of CBE. One such change is the effort shift due to the initial setting up of the system as described in section 3.3. A second change in practice relates to consultation between lecturers and students with regards to student queries of the marking scheme. With CBE a well defined marking structure must exist (this is not always the case with manual assessment). Any questions of why a mark was awarded to a student can be quickly pointed out by referring to the relevant evaluation criteria used during assessment. Finally, staff must be fully briefed on the CBE philosophy in order to help them to adapt their current practices.

6.4 Morals in Industry

The morality issue of using CBE in a teaching environment is effected by the fact that student work is always assessed. In an educational situation this is expected and accepted by the students. If CBE was placed in an industrial development environment it would effectively become a quality assessment system, capable of performing automated audits with minimal physical collection. In this situation employees may not accept monitoring as easily. However we feel that, compared to the physical alternative required for an industrial quality programme, automated collection would reduce the human concerns surrounding the data capture and analysis. One important difference in an industrial setting is that individual anonymity should exist. Any problems in the development should be seen as the responsibility of the whole team and not an individual.

7. Conclusions

The predominant use of computerised training facilities in the education of software engineers allows automated evaluation to take place; expectations however must be realistic about what can be achieved when using such a system. Some aspects of the initial implementation of CBE may require more work than expected. Once this initial work is complete however an automated evaluation system is a realistic proposition. An implemented CBE system must examine information from both products and process. In order to achieve this the system should allow flexibility by providing definition scripts which contain instructions describing methods for the capture and evaluation of information. Several methods of feedback to the lecturer should be provided, including real-time and archived strategies. An essential element for the success of such a system relates to the ease in which data can be accessed from the CASE tools via facilities provided. At the moment such facilities are limited and we hope tool vendors will recognise this necessity for future releases. The idea of automatically collecting and assessing information from an integrated environment is also applicable to industry, and provides a means of performing a quality audit.

References

1. Sherman M, & Drysdale R.L. Teaching Software Engineering in a workstation Environment, *IEEE Software*, pp 68-76. May, 1988.
2. Yu X. & Lamb D.A. Metrics applicable to software design, in *Proc. Software Quality Management 94 Conference*, Vol.1 ,pages 333-348. Computational Mechanics Publications. 1994
3. Coxhead J.F, Harvey J, Dodman E.A. An integrated Environment and Teaching Philosophy for the Education of Software Engineers, *Proceedings of the fifth U.K. Conf. Informations System Teaching: Improving the Practice*. 1992.
4. Ratcliffe M.B, Bott M.F. Stotter-Brooks T.J. and Whittle B.R. The TIPSE: an IPSE for teaching, *Software Engineering Journal*, pp 347-356. September, 1992.
5. Gane C, *Computer-Aided Software Engineering*, page 116, Prentice-Hall International, 1990.
6. Coxhead J.F. & Dodman E.A. Quality Assurance of CASE tools : a monitoring perspective, in *Proc. Software Quality Management 94 Conference*, Vol 2. pages 649-641, Computational Mechanics Publications. 1994.