Web-based Peer Assessment in Learning Computer Programming

Jirarat Sitthiworachart
Mike Joy
Department of Computer Science
University of Warwick
Coventry CV4 7AL, UK
E-mail: jirarat@dcs.warwick.ac.uk

Abstract

Peer assessment is a method of motivating students in learning computer programming, involving students marking and providing feedback on other students' work. This paper reports on the design and implementation of a novel web-based peer assessment system, and discusses its deployment on a large programming module. The results indicate that this peer assessment system has successfully helped students to develop their understanding of computer programming.

1: Introduction

Assessment is a tool for learning, but traditional assessment methods often encourage "surface learning", characterised by memorisation and comprehension of information. Deep learning, such as creating new ideas, and critical judgement of a student's work, can be encouraged by the use of peer assessment [1,2,10]. When students evaluate each others' work they think more deeply, see how others tackle problems, learn to criticise constructively, and display some important cognitive skills such as critical thinking [3,4]. As part of a study investigating the extent that peer assessment can promote deep learning in a programming course, we have developed a novel web-based peer assessment tool. In this paper, we describe the tool and the peer assessment process it supports, and report on its deployment on a large computer programming course.

2: What is Peer Assessment?

Falchikov [5] defines peer assessment as "the process whereby groups rate their peers". Somervell [6] states that peer assessment engages students in making judgements on the other students' work. In the peer assessment process, students are involved both in the learning and in the assessment process. Peer assessment is primarily a tool for learning rather than for summative assessment [7]. Dochy and McDowell [8] remark that "peer assessment is not only a tool to provide a peer with constructive feedback which is understood by the peer. Above all, peer assessment is a tool for the learner himself."

In addition, peer assessment focuses on providing and receiving feedback, which correlates with effective learning. Receiving many and frequent peer feedbacks can prevent some errors and provide hints for making progress in learning [9]. Thus the peer assessment process provides many benefits to students, including the following:

- encouragement of students' deep learning skills in programming by making judgements and providing feedback on other student's work [10];
- students have opportunities to compare and discuss about what constituted a good or bad piece of work, which help them to improve their programming style and think more deeply about the quality of work [12].
- when marking, students realise mistakes that they had made in their own answers - the more marking students did, the better their own results became[10];
- development of self-assessment and reflective learning [2,6]; and
- deepening of students' understanding of the assessment process [7].

3: Peer Assessment Exercise

The UNIX shell programming module in the Computer Science department at the University of Warwick was chosen for this investigation. This module aims to give students a basic understanding of the UNIX operating system, and competence in programming using a UNIX shell. Students learn how to design and develop programs in the shell, which is a programming language

0-7695-1967-9/03 $17.00 © 2003 IEEE

that allows programs to be written in many styles. There are three programming assignments in this module, which students submit via the department's "BOSS" online submission system [11]. The second of the three assignments was marked using a peer assessment process. The purposes of performing the experiment in peer assessment were:

- to investigate the extent that peer assessment in a programming course promotes deep learning;
- to assess the accuracy of students' judgements during a peer assessment exercise; and
- to provide evidence that peer assessment in computer programming has a positive pedagogical effect.

3.1: Process

This peer assessment exercise was divided into three separate stages, as shown in Figure 1.

Stage I: Students do the assignment in their own time. Then they submit the assignment via the online submission system. Ten automatic tests are then run on the submitted programs.

Stage II: Students were divided into the small groups (three students per group). Each group consisted of students with a range of ability. Each student was assigned three other students' assignments to mark during the first half hour of a lab session. Then they discussed their marking with the other students in their group, who marked the same assignments.

Stage III: In their own time, each student marked the quality of three markers' marking. This additional stage aims to make students take marking more seriously during the previous stage.

3.2: Mark scheme

The marking scheme is illustrated in Figure 2, using the following definitions.

Automatic test: The online submission system tests a student's assignment against different inputs to check whether it functions correctly. Ten tests are used.

Marker: Student marker marks assignments.

Feedback marker: Student feedback marker reports on the quality of the marking given by the three markers.

Script: Assignment that students submit via the online submission system.

In this peer assessment process, 50% of the marks are awarded by the teacher (automatic tests) and the remaining 50% are awarded by the students (peer assessment):

- Automatic Test 50%
- Peer Assessment
  - Part I: mark assignment 30%
  - Part II: mark quality of marking 20%

Peer marks are based on three markers; the average of the three marks is calculated. If one of markers does not appear to have marked work seriously, the mark he or she gives will not be included in the average and the other marks will be scaled. The marking of assignments by students is possible since they are given guidance, automatic test scores and results, a marking scheme, and well explained marking criteria.

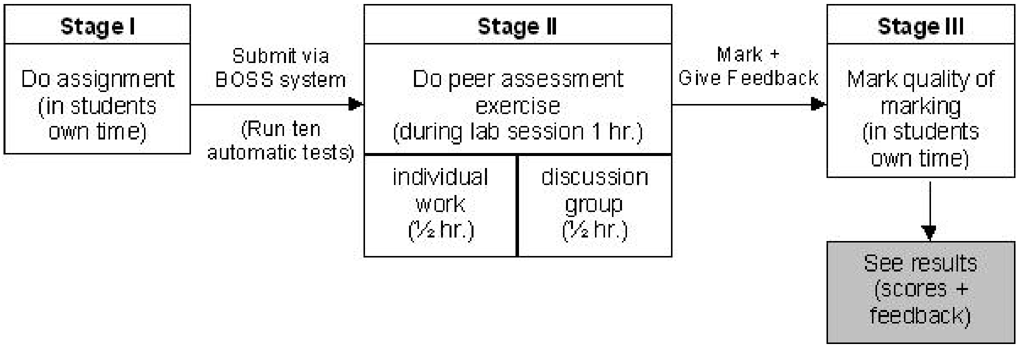

Stage I
Do assignment (in students own time)
Submit via BOSS system
(Run ten automatic tests)
Stage II
Do peer assessment exercise (during lab session 1 hr.)
individual work (½ hr.)
discussion group (½ hr.)
Mark + Give Feedback
Stage III
Mark quality of marking (in students own time)
See results (scores + feedback)

Figure 1 Peer assessment process

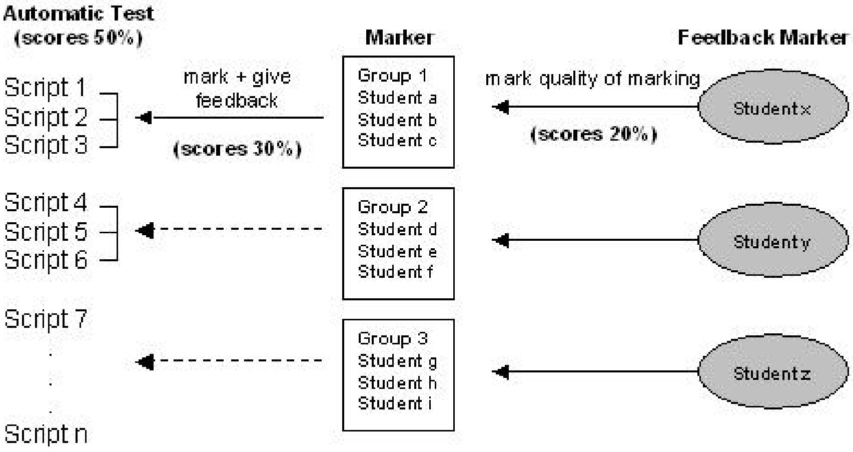


Figure 2 Peer assessment mark scheme

4: Design of Web-based Peer Assessment

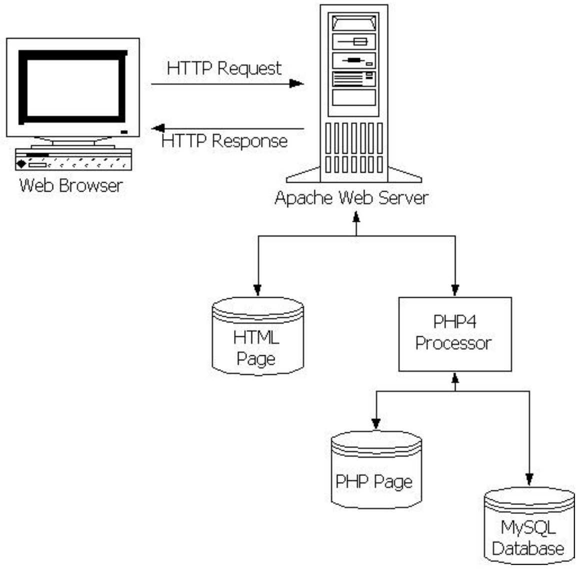
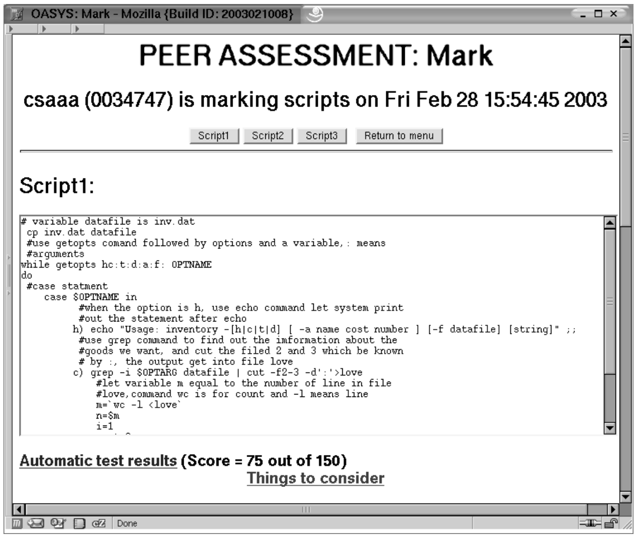


Figure 3 Architecture of the web-based peer assessment system

The web-based peer assessment software uses the standard combination of Apache web server, the PHP4 programming language, and a MySQL database running on a Linux platform. This architecture is illustrated in Figure 3. Dynamic web pages are written in PHP4 and static web pages are written in HTML.

This web-based peer assessment provides anonymity for all users. Students are allowed to revise the marks they give until the marking deadline is reached. They receive a username and password by email before starting the peer assessment exercise. After students login, the menu page displays three steps for students to follow (i.e. mark assignment, mark quality of marking, and see mark). They can see the scripts that they have been assigned to mark easily by clicking on the script buttons (Figure 4). They can view the automatic test results by clicking on the link on each script page to open a popup window displaying the results. A "Things to consider" link is provided below each script, to show the marking guidance.



PEER ASSESSMENT: Mark

csaaa (0034747) is marking scripts on Fri Feb 28 15:54:45 2003

Script1:

```
# variable datafile is inv.dat
 cp inv.dat datafile
 #use getopts comand followed by options and a variable,: means
 #arguments
while getopts hc:t:d:a:f: OPTNAME
do
 #case statment
    case $OPTNAME in
          #when the option is h, use echo command let system print
          #out the statement after echo
         h) echo "Usage: inventory -[h|c|t|d] [ -a name cost number ] [-f datafile] [string]" ;;
          #use grep command to find out the imformation about the
          #goods we want, and cut the filed 2 and 3 which be known
          # by :, the output get into file love
         c) grep -i $OPTARG datafile | cut -f2-3 -d':'>love
             #let variable m equal to the number of line in file
             #love,command wc is for count and -l means line
             m=`wc -l <love`
             n=$m
             i=1
```

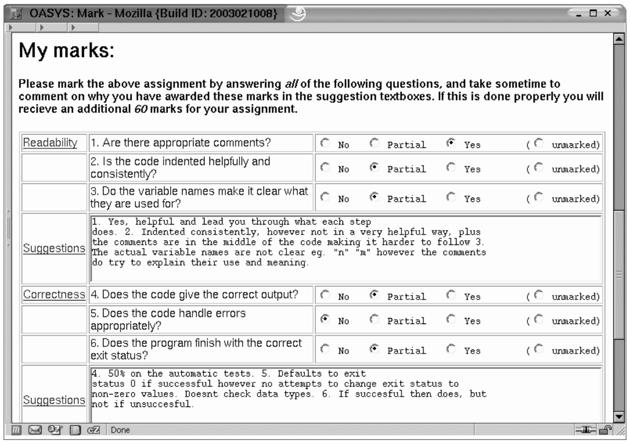
Automatic test results (Score = 75 out of 150)
Things to consider

Figure 4 Assignment script on 'Mark' web page

4.1: Step I: Mark assignment

In this visual inspection step, students mark and provide feedback on other students' assignments by answering nine questions about:

- readability (comments, indentation, variable names);
- correctness (correct output, appropriate error handling, correct exit status); and
- style (easy to follow, well structured, use of appropriate utilities).



My marks:

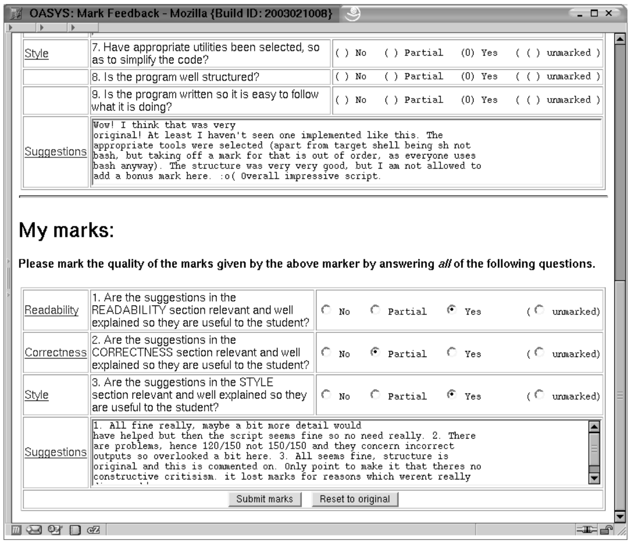
Please mark the above assignment by answering *all* of the following questions, and take sometime to comment on why you have awarded these marks in the suggestion textboxes. If this is done properly you will recieve an additional *60* marks for your assignment.

Readability	1. Are there appropriate comments?	○ No ○ Partial ● Yes (○ unmarked)
	2. Is the code indented helpfully and consistently?	○ No ● Partial ○ Yes (○ unmarked)
	3. Do the variable names make it clear what they are used for?	○ No ● Partial ○ Yes (○ unmarked)
Suggestions	1. Yes, helpful and lead you through what each step does. 2. Indented consistently, however not in a very helpful way, plus the comments are in the middle of the code making it harder to follow 3. The actual variable names are not clear eg. "n" "m" however the comments do try to explain their use and meaning.	
Correctness	4. Does the code give the correct output?	○ No ● Partial ○ Yes (○ unmarked)
	5. Does the code handle errors appropriately?	● No ○ Partial ○ Yes (○ unmarked)
	6. Does the program finish with the correct exit status?	○ No ● Partial ○ Yes (○ unmarked)
Suggestions	4. 50% on the automatic tests. 5. Defaults to exit status 0 if successful however no attempts to change exit status to non-zero values. Doesnt check data types. 6. If succesful then does, but not if unsuccesful.	

Figure 5 Marking criteria on 'Mark' web page

These are answered for each script by selecting simple multiple choices, i.e. 'No', 'Partial', and 'Yes'. The default answer is set as 'unmarked' (Figure 5). Students give a comment for each group of three questions. An explanation of the marking criteria is provided for each group of questions by clicking on the links on the left.

4.2: Step II: Mark quality of marking



Style	7. Have appropriate utilities been selected, so as to simplify the code?	( ) No ( ) Partial (0) Yes (( ) unmarked)
	8. Is the program well structured?	( ) No ( ) Partial (0) Yes (( ) unmarked)
	9. Is the program written so it is easy to follow what it is doing?	( ) No ( ) Partial (0) Yes (( ) unmarked)
Suggestions	Wow! I think that was very original! At least I haven't seen one implemented like this. The appropriate tools were selected (apart from target shell being sh not bash, but taking off a mark for that is out of order, as everyone uses bash anyway). The structure was very very good, but I am not allowed to add a bonus mark here. :o(Overall impressive script.	

My marks:

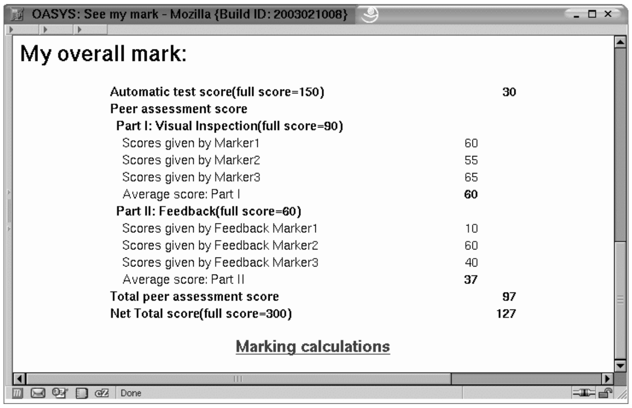
Please mark the quality of the marks given by the above marker by answering *all* of the following questions.

Readability	1. Are the suggestions in the READABILITY section relevant and well explained so they are useful to the student?	○ No ○ Partial ● Yes (○ unmarked)
Correctness	2. Are the suggestions in the CORRECTNESS section relevant and well explained so they are useful to the student?	○ No ● Partial ○ Yes (○ unmarked)
Style	3. Are the suggestions in the STYLE section relevant and well explained so they are useful to the student?	○ No ○ Partial ● Yes (○ unmarked)
Suggestions	1. All fine really, maybe a bit more detail would have helped but then the script seems fine so no need really. 2. There are problems, hence 120/150 not 150/150 and they concern incorrect outputs so overlooked a bit here. 3. All seems fine, structure is original and this is commented on. Only point to make it that theres no constructive critisism. it lost marks for reasons which werent really	

Figure 6 Mark quality of marking web page

In this step, students mark the quality of marking given by each of the three markers on a particular script. They need to answer three questions about whether the suggestions the markers gave in each section (readability, correctness, and style) are relevant, well explained and useful to students. The marking given by the three markers is displayed at the top of the page and the student enters the feedback marks at the bottom (Figure 6).

4.3: Step III: See mark



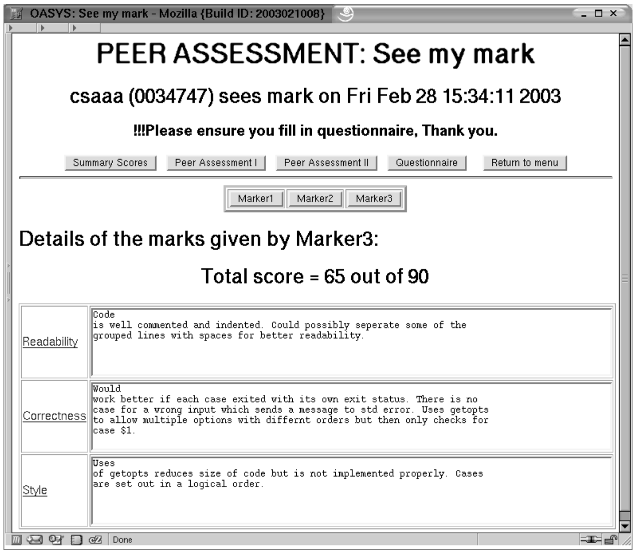
My overall mark:

Automatic test score(full score=150)		**30**
Peer assessment score		
Part I: Visual Inspection(full score=90)		
Scores given by Marker1	60	
Scores given by Marker2	55	
Scores given by Marker3	65	
Average score: Part I	**60**	
Part II: Feedback(full score=60)		
Scores given by Feedback Marker1	10	
Scores given by Feedback Marker2	60	
Scores given by Feedback Marker3	40	
Average score: Part II	**37**	
Total peer assessment score		**97**
Net Total score(full score=300)		**127**

Marking calculations

Figure 7 See mark web page

In this final step, students can see their mark from both the automatic test and the peer assessment (Figure 7). A 'Marking calculations' link at the bottom of the page provides an explanation of how the overall mark is calculated. If the students do not mark any of three scripts, they may lose some marks.



PEER ASSESSMENT: See my mark

csaaa (0034747) sees mark on Fri Feb 28 15:34:11 2003

!!!Please ensure you fill in questionnaire, Thank you.

Details of the marks given by Marker3:

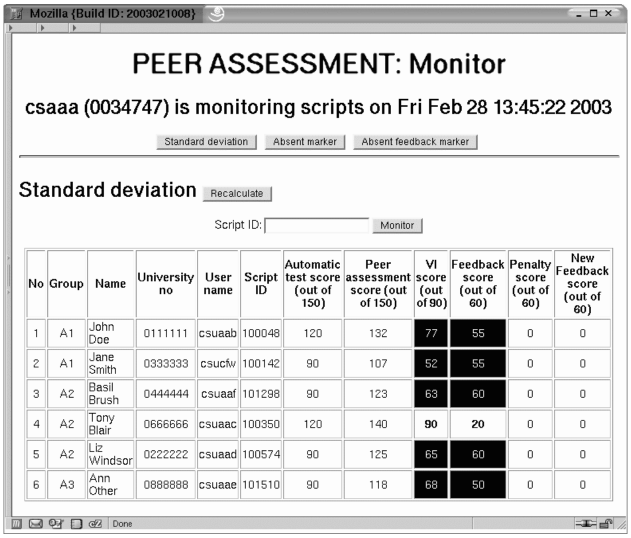
Total score = 65 out of 90

Readability	Code is well commented and indented. Could possibly seperate some of the grouped lines with spaces for better readability.
Correctness	Would work better if each case exited with its own exit status. There is no case for a wrong input which sends a message to std error. Uses getopts to allow multiple options with differnt orders but then only checks for case $1.
Style	Uses of getopts reduces size of code but is not implemented properly. Cases are set out in a logical order.

Figure 8 Feedback from peer on 'See mark' web page

The full mark and comments that the three peer markers gave the student's assignment are also available (Figure 8). This also includes the full mark that they were given based on the quality of their own marking.

4.4: Monitor marking



PEER ASSESSMENT: Monitor

csaaa (0034747) is monitoring scripts on Fri Feb 28 13:45:22 2003

Standard deviation

Script ID:

No	Group	Name	University no	User name	Script ID	Automatic test score (out of 150)	Peer assessment score (out of 150)	VI score (out of 90)	Feedback score (out of 60)	Penalty score (out of 60)	New Feedback score (out of 60)
1	A1	John Doe	0111111	csuaab	100048	120	132	77	55	0	0
2	A1	Jane Smith	0333333	csucfw	100142	90	107	52	55	0	0
3	A2	Basil Brush	0444444	csuaaf	101298	90	123	63	60	0	0
4	A2	Tony Blair	0666666	csuaac	100350	120	140	**90**	**20**	0	0
5	A2	Liz Windsor	0222222	csuaad	100574	90	125	65	60	0	0
6	A3	Ann Other	0888888	csuaae	101510	90	118	68	50	0	0

Figure 9 Monitor marking web page

In addition, Figure 9 shows the 'Monitor marking' web page, which reports the students' marks and any absent markers, and is only available for tutors. The highlighted columns show the standard deviation of the three markers for both Step I and Step II in order to know how spread out the marks are. If the standard deviation is less than a preset value, it is acceptable, but if the standard deviation is more than a given upper limit, it means the marks from the three markers have a very wide range, which means the tutor may have to reconsider the marks for that student. The tutor can access each script by using the 'Script ID' box at the top of the web page.

5: Results and Discussion

At the end of the process, each student was required to fill in an online questionnaire. Summary statistics from the questionnaires yield the following information.

- 69% of students realise mistakes that they made in their own answer when marking other students' work.
- 58% of students feel comfortable when assigning marks. A few students did not fully understand the marking criteria.
- 65% of students are satisfied with their mark from the peer assessment, and considered that the peer feedback they received was relevant and useful.
- 80% of students agree that seeing good and bad programs help them in learning programming, and marking helps them to think more deeply about their own work.

Peers may not have adequate knowledge and experience to evaluate others' work, even when guidance and well-explained marking criteria are provided. The nature of the programming assignment did not lend itself to there being only one model answer, and a variety of styles of solution were possible. The tutors should therefore give students adequate guidance during the marking process to assist students. Also, it was difficult in the students' view to avoid friendship marking, resulting in over-marking (they often felt more favourable towards their friends) [12].

6: Conclusions

We have described a peer assessment process, together with supporting web-based software, which we have used to test the effectiveness of peer assessment in learning programming languages. The process we have used is novel, since students are engaged not only in marking each other's work, but also in evaluating the quality of marking of their peers. Preliminary evaluation of the exercise indicates that it has contributed positively to the students' learning experience.

7: References

[1] AAHE Bulletin, "Deep Learning, Surface Learning", **45**(8), 10-13, 1993

[2] Topping, K., "Peer assessment between students in colleges and universities", *Review of Educational Research,* **68**, 249-276, 1998

[3] Sluijsmans, D., Dochy, F., and Moerkerke, G., "Creating a learning environment by using self- peer- and co-assessment", *Learning Environments Research*, **1**, 293-319, 1999

[4] Tsai, C., Lin, S., and Yuan, S., "Developing science activities through a net worked peer assessment system", *Computers & Education*, **38**, 241-252, 2002

[5] Falchikov, N., Learning Together: Peer tutoring in higher education, Routledge Falmer, London, 2001

[6] Somervell, H., "Issues in assessment, enterprise and higher education: the case for self-, peer and collaborative assessment", *Assessment and Evaluation in Higher Education*, **18**, 221-233, 1993

[7] Brown, G., Bull, J., and Pendlebury, M., Assessing student learning in higher education, Routledge, London, 170-184, 1997

[8] Dochy, F., McDowell, L., "Assessments as a tool for learning", *Studies in Educational Evaluation*, **23**(4), 279-298, 1997

[9] Lin, S., Liu, E., and Yuan, S., "Web-based peer assessment: feedback for students with various thinking-styles", *Journal of Computer Assisted Learning*, **17**, 420-432, 2001

[10] Bhalerao, A. and Ward, A., "Towards electronically assisted peer assessment: a case study", *Association for Learning Technology journal (ALT-J)*, **9**(1), 26-37, 2001

[11] Joy, M., Luck, M., "The BOSS System for On-line Submission and Assessment", *CTI Computing: Monitor*, **10**, 1998

[12] Brindley, C., Scoffield, S., "Peer Assessment in Undergraduate Programmes", *Teaching in Higher Education*, **3**(1), 79-89, 1998