Enabling Effective Synoptic Assessment via Algorithmic Constitution of Review Panels

Nisansa Dilushan de Silva
Department of Computer Science and Engineering, Faculty of Engineering, University of Moratuwa, Sri Lanka. nisansads@uom.lk

Shahani Markus Weerawarana
Department of Computer Science and Engineering, Faculty of Engineering, University of Moratuwa, Sri Lanka. shahani.w@gmail.com

Amal Shehan Perera
Department of Computer Science and Engineering, Faculty of Engineering, University of Moratuwa, Sri Lanka. shehan@uom.lk

Abstract - This paper presents an algorithmic tool that was used to create panels of experts for the synoptic assessment of a software engineering project course that is targeted towards fostering innovation and creativity in software engineering students. Synoptic assessments succeed with the ability to formulate expert evaluation panels. Yet many industry experts are busy professionals, and hence, the process of constituting appropriately balanced evaluation panels for project demonstrations is a significant challenge. The discussion includes the outcomes of using the algorithm to automate the panel composition and scheduling process for synoptic assessments of project demonstrations for a batch of 100 students following the Bachelor of Engineering (Honors) degree program in the Department of Computer Science and Engineering at the University of Moratuwa in Sri Lanka. We describe the challenges we faced and our approach towards addressing the issues as well as our encouraging successes.

Keywords - Computer science education; engineering education; software engineering projects; technology in education.

I. INTRODUCTION

The graduates of the Bachelor of Engineering (Honours) program of the Department of Computer Science and Engineering (CSE) at the University of Moratuwa are extremely competitively recruited by the Sri Lankan IT industry. Whilst the undergraduate curriculum offered by the CSE department is based on the ACM/IEEE curriculum guidelines [1], the content and delivery of the courses are strongly influenced by the needs of the local IT industry and the Sri Lankan academic culture. Thus over the past few years, the CSE department has developed a highly synergistic relationship with the IT industry which in turn has positively benefitted the design and delivery of many of the practice-oriented courses in the undergraduate program.

In alignment with Asian values and the general ethos in many Asian countries, Sri Lanka also has a very strong academic culture with high emphasis placed on excellence in education. However, though the Sri Lankan school-based education process produces highly knowledgeable students, the school education system has a low-level of emphasis on the development of soft skills such as communication, leadership, creativity and entrepreneurial thinking. Thus, each year, the cohort of students that enter the public universities in Sri Lanka after a highly competitive and rigorous examination process, tend to be narrowly focused on theoretical knowledge and display high competence in written examination-based assessments. The onus of bridging the obvious gaps in soft skills in practice-oriented capabilities in these students therefore, lies within the higher education sector. Cognizant of this important issue, the CSE department has continuously been researching and experimenting with many teaching, learning and assessment practices that would help address this need.

In this paper, we discuss how we addressed a significant challenge in introducing a synoptic assessment framework for a compulsory software engineering project course in the 5th semester, a course which was specifically designed to foster creativity, innovation and software engineering rigor [2].

II. BACKGROUND

The University of Moratuwa is the premier technical university in Sri Lanka. It consists of three faculties - the Faculty of Architecture, the Faculty of Engineering and the Faculty of Information Technology. Each year the Faculty of Engineering admits approximately 1000 students who obtained the highest marks in the country in the Physics, Chemistry and Mathematics subjects at the highly competitive Advanced Level examination. The Department of Computer Science and Engineering (CSE) was added to the Faculty of Engineering in 1985 to offer a specialization in Computer Science and Engineering in order to address the needs of the burgeoning Sri Lankan IT industry. The expectations of the government and the IT industry were that the graduates from this degree program would eventually provide the expertise and technical leadership required in this rapidly advancing economic growth sector.

The CSE department has strived to maintain very strong links with the IT industry through a Department Industry Consultative Board (DICB) which is represented by all major IT employers, and the Computer Science and Engineering Society, which is the alumni network of the CSE department. This link has been bolstered by an annual recruitment fair and many other informal gatherings where industry participants continue to provide valuable feedback on potential curriculum improvements [2].

The CSE department undergraduate program leads to a Bachelor of Engineering (Honours) degree after an eight-semester program. The CSE program includes courses in a variety of disciplines such as Computer Science, Mathematics,
In addition to the courses taught at the university, in their third year, the undergraduates are exposed to industry experience through a compulsory six-month industrial training program. Thus, the CSE undergraduate program is structured such that the students acquire academic knowledge as well as real world practical skills via industry internships.

Software engineering courses in the curriculum for the CSE undergraduate program systematically introduce students to increasingly sophisticated software engineering skills. Java is taught as an object oriented programming language initially to introduce basic programming skills. In the second course formal object oriented principles are introduced and the students are exposed to apply object oriented principles in designing and implementing software solutions. Aspects of production-level software engineering are introduced in the third course of software engineering. In the following semester, the students follow the Software Engineering Project course (CS 3202). In this course, the students get the opportunity to practice the concepts introduced in the previous semester’s theoretical course. Since the objective is for each student to experience the practical aspects of real world software engineering, the projects are completed individually [2].

The discussions in this paper are based on our experiences in conducting this project-oriented course. The success of the software engineering project course depend on the assessment process, and as noted earlier, the assessment process requires the compilation of specialized panels of experts from the industry.

III. SYNOPTIC ASSESSMENT FRAMEWORK

Since this course is offered in the semester preceding the 6th semester industrial training experience, the requisite individual project in the course is comparable to a mini Capstone project. Thus the course design straddles several program ILOs at a 5th semester level from a Bloom’s taxonomy cognitive skills perspective [5], along with an emphasis on creativity. Therefore, devising a suitable evaluation framework was complicated. We eventually decided to adopt a synoptic assessment approach for this course. In [3], the term ‘synoptic assessment’ is formally defined as a, “form of assessment which tests candidates’ understanding of the connections between the different elements of a subject”. We determined it to be the most appropriate assessment technique for this capstone-like project course, since it features learning elements across many courses that the 5th semester students had completed thus far.

In [8], Jackson noted that synoptic assessment “enables students to integrate their experiences, providing them with important opportunities to demonstrate their creativity”. In [7], Elton stated that assessment of creative work should be ‘viewed in light of the work’, highlighting important aspects such as, “the ability of experts to assess work in their own field of expertise” and “the willingness to employ judgment”. Further, in [6], Balachin said that the reliability of subjective evaluation is enhanced when a mechanism of, “consensual assessment by several judges” is used. Based on these relevant literature findings, we decided to incorporate evaluation panels comprising industry experts and internal faculty members who had been involved in the students’ course modules up to the 6th semester.

However, we still had many challenges in the process of incorporating such expert evaluator panels for our synoptic assessment framework. The underlying issues included the assessment of a significant number of interim deliverables from a large number of students, and the balanced evaluation of the end-of-semester project demonstrations from technical and creative standpoints, which required evaluators with expertise in a wide range of technologies and application domains.

Additionally, we faced many practical problems in activating this evaluation framework including, the limited number of evaluators and conflicting time constraints among evaluators when scheduling the large number of project demonstrations. Compilation of a sufficient number of technically specialized panels based on the time constraints turned out to be a time consuming and tedious task. It became practically impossible to dynamically allocate frequent time constraints based change requests in the manual process and hence we decided to automate the panel composition and scheduling process.

IV. OBJECTIVES OF THE ALGORITHM

We devised and implemented an algorithm with the following primary objective: Each student will be assigned a ‘best fit’ panel of evaluators comprising external industry experts and internal faculty members considering the technologies used in the student’s project.

The secondary objectives in the algorithm design included; (1) optimal allocation based on multiple and often conflicting evaluator availability constraints; (2) balance of external versus internal evaluators in the panels; (3) minimization of the number of panel composition reshuffles; (4) avoidance of the same internal evaluator who assessed the mid-semester project demonstration being included in the end-semester evaluation panel; and (5) preventing internal evaluators who mentored specific projects being included in the end-semester evaluation panel for the same projects.

V. DATA COLLECTION AND FORMULARIZATION

Students were asked to list the technologies that were used in their projects, resulting in a master list of 42 different technologies. The student requests were recorded in matrix A, as shown in Table I. If a certain project needed a particular technology, the relevant matrix element was set to 1 otherwise it was set to 0. The sum of the elements in matrix A (Equation 1) provided the total number of student requests be 244.

The external evaluators (e=9) were also requested to indicate their areas of technology expertise. The evaluator responses were also recorded in matrix B, as shown in Table II. If a certain evaluator was comfortable with evaluating a particular technology, the relevant matrix element was set to 1 otherwise it was set to 0.

\[ \text{Equation 1} \]

\[ \text{Equation 2} \]
TABLE I: STUDENT TECHNOLOGY REQUESTS (MATRIX A)

<table>
<thead>
<tr>
<th></th>
<th>Tech. 1</th>
<th>Tech. 2</th>
<th>...</th>
<th>Tech. 42</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student 1</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>0</td>
</tr>
<tr>
<td>Student 2</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Student 101</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>1</td>
</tr>
</tbody>
</table>

TABLE II: EXTERNAL EVALUATOR TECHNOLOGY EXPERTISE (MATRIX B)

<table>
<thead>
<tr>
<th></th>
<th>Tech. 1</th>
<th>Tech. 2</th>
<th>...</th>
<th>Tech. 42</th>
</tr>
</thead>
<tbody>
<tr>
<td>E. Evaluator 1</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>E. Evaluator 2</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>E. Evaluator 9</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>1</td>
</tr>
</tbody>
</table>

\[ \sum_{j=1}^{42} \sum_{i=1}^{101} A_{ij} = 244 \]

EQUATION 1: TOTAL NUMBER OF STUDENT REQUESTS

The Evaluator Conflict Matrix (Matrix C) shown in Table III was used to fulfil the secondary objectives 4 and 5. If a certain internal evaluator \((r=7)\) was the person who assessed the mid-semester project demonstration of a relevant student project or if a certain internal evaluator was mentoring the project in question, the relevant matrix element was set to 1, and otherwise it was set to 0.

TABLE III: EVALUATOR CONFLICT MATRIX (MATRIX C)

<table>
<thead>
<tr>
<th></th>
<th>I. Evaluator 1</th>
<th>I. Evaluator 2</th>
<th>...</th>
<th>I. Evaluator 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student 1</td>
<td>1</td>
<td>0</td>
<td>...</td>
<td>0</td>
</tr>
<tr>
<td>Student 2</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Student 101</td>
<td>1</td>
<td>1</td>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>

The Evaluator Time-Slot Matrix (Matrix D) shown in Table IV was used to fulfil the secondary objective 1. If a certain evaluator \((r+e = 16)\) was available in the relevant time slot \((t=24)\), the matrix element was set to 1, and otherwise it was set to 0.

TABLE IV: EVALUATOR TIME-SLOT MATRIX (MATRIX D)

<table>
<thead>
<tr>
<th></th>
<th>Time slot 1</th>
<th>Time slot 2</th>
<th>...</th>
<th>Time slot 24</th>
</tr>
</thead>
<tbody>
<tr>
<td>I. Evaluator 1</td>
<td>1</td>
<td>1</td>
<td>...</td>
<td>0</td>
</tr>
<tr>
<td>I. Evaluator 2</td>
<td>1</td>
<td>1</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>I. Evaluator 7</td>
<td>1</td>
<td>1</td>
<td>...</td>
<td>0</td>
</tr>
<tr>
<td>E. Evaluator 1</td>
<td>1</td>
<td>1</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>E. Evaluator 2</td>
<td>0</td>
<td>0</td>
<td>...</td>
<td>1</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>E. Evaluator 9</td>
<td>0</td>
<td>1</td>
<td>...</td>
<td>0</td>
</tr>
</tbody>
</table>

VI. ALGORITHMIC PANEL CONSTITUTION PROCESS

For the fulfilment of secondary objective 2, it was decided to first create a number of motif panels with internal evaluators, considering the seniority and the degree of prior involvement with the CS 3202 course. Further, for the fulfilment of the same objective, it was decided that each panel should contain a minimum of two individuals, where at least one would be an internal evaluator. With these constraints, the number of vacancies for external evaluators for each motif panel was calculated.

The evaluation panels were designed to operate in parallel within each time-slot with multiple panels per time slice as shown in Figure 1. However the first two time-slots were set aside for a special joint evaluation and grading calibration activity. These special evaluations were instituted based on our past experiences with the evaluation of project demonstrations in this course, as a mechanism to address the variances in the grading expectations and assessment styles of the individual evaluators [2]. Through this joint activity, we endeavour to instil a common notion of the assessment style and provide a sense of the levels of expectation in grading, across all internal and external evaluators. Further, these two time-slots were assigned to two individual projects that were representative of projects faring exceptionally well and of projects faring averagely, respectively, at the time of the mid-evaluation. Therefore all time slots except the first two time-slots contained a set of panel slots which was equal in cardinality to the number of active panels.

![Figure 1: Time-Slot to Panel-Slot Mapping](attachment:image.png)

Time-slot categories were calculated using the data from Matrix D in such a way that the categories are divided by the events of entrance or exit of an evaluator. Thus the evaluation panels were kept static within each time-slot category and were used as the seeds for the panels in the next time-slot category to minimize the number of panel composition reshuffles. This property was previously discussed as the secondary objective 3. Since the panels of the latter time-slot category would inherit from the former time-slot category, panels within a given time-slot category were referred to, as a ‘generation of panels’.

The initial motifs were updated with the time-slot category data and a randomized algorithm was used to allocate external evaluators to the initial motif panels with consideration of availability data from Matrix D and the number of vacancies. The panels created in this generation were referred as panels of generation 0. Using the panels from generation 0, the panel
slots of the first time-slot category were populated. As mentioned earlier, the first two time-slots of generation 0 only had one panel-slot each while the rest of the time-slots of generation 0 had a set of panel slots which is equal in cardinality to the number of active panels in the first time-slot category.

Depending on the change that took place at the border of the time-slot category, split and transfer operations were done on the panels from the immediately preceding generation to create the next generation of panels.

Panel splitting was done when new evaluators arrived. Two existing panels were split at each split request and made into three motif panels. The newcomers were assigned to those panels again using the aforementioned randomized algorithm.

Transfers were done when evaluators leave and the panel that they were serving become non-compliant to the panel creation criteria stated above due to their departure. The remaining evaluators were transferred into other existing panels so that the panel creation criteria were preserved.

Each resultant generation of panels was populated with a set of panel slots equal in cardinality to the number of active panels in the relevant time-slot category.

The total number of panel slots created across all time-slots was 108. The reason to have more panel slots than the actual number of projects (n=101) was to reduce the rigidness of the constraints that the algorithmic project assignment process had to operate within.

VII. ALGORITHMIC PROJECT ASSIGNMENT PROCESS

The Student Technology Request Feasibility (Matrix E) shown in Table V was calculated by means of Equation 2. Thereafter, application of Equation 3 revealed that it would be feasible to satisfy 141 of the 244 student requests covering 18 technologies.

\[ E = A \cdot B^T \]

**Equation 2: Student technology requests feasibility matrix calculation**

<table>
<thead>
<tr>
<th>E. Evaluator 1</th>
<th>E. Evaluator 2</th>
<th>...</th>
<th>E. Evaluator 9</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student 1</td>
<td>0</td>
<td>1</td>
<td>...</td>
</tr>
<tr>
<td>Student 2</td>
<td>1</td>
<td>2</td>
<td>...</td>
</tr>
<tr>
<td>Student 101</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

\[ \text{Count}(x | x \text{ is an element of Matrix E} & x \neq 0) = 244 \]

**Equation 3: Satisfiable number of student technology requests calculation**

The value portion of internal evaluators calculated using the data from Matrix C, was given by \( V_{Pro,Int} \) as shown in Equation 4, where \( p \) gives the number of internal evaluators assigned to the relevant panel and \( i \) is the index of the relevant project in matrix C. It is worthy to note here that since these are individual projects, index number of the students has a one to one relationship with index of the projects. Thus \( i \) can also be referred to as the index number of the students.

\[ V_{Pro,Int} = \prod_{k=1}^{p} C_{i,k} \]

**Equation 4: Value portion calculation of internal evaluators**

The value portion of external evaluators, calculated using the data from Matrix E, was given by \( V_{Pro,ext} \) as shown in Equation 5, where \( m \) gives the number of internal evaluators assigned to the relevant panel, \( i \) is the index of the relevant project in matrix E.

\[ V_{Pro,ext} = \sum_{i=1}^{m} E_{i,j} \]

**Equation 5: Value portion calculation of external evaluators**

The reason for taking the product for the internal evaluators and the summation for the external evaluators is the fact that the internal evaluator matrix (C) is essentially a binary matrix about existence or nonexistence of constraints and conflicts and the external evaluator matrix (E) on the other hand is an integer matrix with the relevance factor of each external evaluator for each project as elements.

The gross merit value of a given project being evaluated by a given panel was given by \( V_{Pro,g–pan} \) as shown in Equation 6.

\[ V_{Pro,g–pan} = V_{Pro,Int} \ast V_{Pro,ext} \]

**Equation 6: Gross merit value calculation**

The net merit value of a given project being evaluated by a given panel was given by \( V_{Pro,n–pan} \) as shown in Equation 7.

\[ V_{Pro,n–pan} = w_1 \ast V_{Pro,g–pan} + w_2 \]

**Equation 7: Net merit value calculation**

The \( w_2 \) variable was used to dampen the net merit value and prevent a single panel gaining above average advantage over the others within the same time-slot category. The constant weight \( w_2 \) variable was used so that the calculated gross merit value (\( V_{Pro,g–pan} \)) does not get completely subdued by the damping variable value \( w_2 \). Thus the value of \( w_2 \) is initiated to a variable which is proportional to the size of the relevant time slot category and is reduced each time a certain panel is reused. \( w_1 \) is set to a value which would ensure that the value obtained by the multiplication of the gross merit value and \( w_2 \) will be greater than the maximum possible \( w_2 \) even at the minimum attainable value of the gross merit value.
The calculated merit values, of each project being evaluated through each panel slot, were then put into a matrix $F$ of the form shown in Table VI.

<table>
<thead>
<tr>
<th>Panel-slot 1</th>
<th>Panel-slot 2</th>
<th>...</th>
<th>Panel-slot 108</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student 1</td>
<td>20</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>Student 2</td>
<td>9</td>
<td>32</td>
<td>65</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Student101</td>
<td>0</td>
<td>19</td>
<td>16</td>
</tr>
</tbody>
</table>

The inverted value matrix $G$ shown in Table VII was created by applying Equation 8 on the elements of Matrix $F$, where $M_i$ is the maximum value present in matrix $F$.

$$G_{i,x} = M_i - F_{i,x}$$

Equation VIII: Inverted Value Calculation

<table>
<thead>
<tr>
<th>Panel-slot 1</th>
<th>Panel-slot 2</th>
<th>...</th>
<th>Panel-slot 108</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student 1</td>
<td>66</td>
<td>82</td>
<td>86</td>
</tr>
<tr>
<td>Student 2</td>
<td>77</td>
<td>54</td>
<td>21</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
<tr>
<td>Student101</td>
<td>86</td>
<td>67</td>
<td>70</td>
</tr>
</tbody>
</table>

A well-known combinatorial optimization algorithm known as the “Hungarian algorithm” [4], which is used to solve classical assignment problems in polynomial time, was run on the inverted value matrix $G$. This resulted in a Boolean matrix $H$ indicating whether the given project is assigned to a given panel slot or not.

VIII. LAYERED OPTIMIZATION OF ALGORITHM

A second layer of indirection was placed over the core algorithm to return a pool of panel assignment schedules instead of a single schedule. The technical reason for this was to significantly reduce the possibility of unfairness as a consequence of the underlying randomness of the base ‘seed’ evaluation panels that were generated in the core algorithm.

The total value of each assignment schema was calculated by multiplying each element of Boolean matrix $H$ with each element of the original value matrix $F$ as shown in Equation 9 where; $V$ is the total value of the assignment schema, $H_{i,j}$ is an element from Boolean matrix $H$ expressed as 1 or 0.

$$V = \sum_{i=1}^{101} \sum_{x=1}^{108} H_{i,x} \cdot F_{i,x}$$

Equation IX: Assignment Schema Total Value Calculation

The assignment schemas were sorted by this total value and the most favourable assignment schema was selected as the final assignment schema.

IX. RESULTS

The algorithm was run on the collected data and an experiment run was done varying the pool size from 10 to 2000. Figure 2 shows the resultant values of assignment schemas for pool sizes 50 and 2000.

By these results it was observed that a pool size of 50 was sufficient to overcome the issues introduced by the internal randomized algorithm since there is no tangible increase of the attainable maximum total value of assignment schema.

The selected schema was able to match 120 of the total 141 feasible requests giving an 85.12% success rate. The average number of requests satisfied per student was 92.1% among the 83 students whose requests constituted the 141 feasible requests. The average number of requests satisfied per technology was 71.69% among the 18 technologies. The running time of the algorithm implementation varied from 10 seconds to 20 seconds on a 2.26GHz quad core computer.

Because of this new algorithm, we were able to fulfil 75% of the requests made for the more obscure and rare technologies (the technologies that had 5 requests or less) that were required to be evaluated. If the panel allocation was done manually, more importance would have been given to the common technologies - those which most of the evaluators are comfortable with. On the other hand, if the manual panel assignment was focused on fulfilling the requests made on rare technologies, requests on more common technologies would have been heavily affected. But the algorithm managed to maintain a fulfilment success rate of 86.78% on requests made on common technologies as well.

The experience the students gained with the use of the synoptic approach for the evaluation of the project and the confidence that they have gained may have significantly contributed towards the steady increase in the number of Google Summer of Code (GSoC) awards consistently garnered by CSE department students. The CSE department has been producing the highest number of GSoC award winners in the world, every year, since 2007 [2][9][10][11][12].

Some software innovations developed in this course by students were entered in competitions, and students have won national and international awards. For instance, one student won a Mananthan award [13] in India for developing a browser extension to convert content of non-Unicode Sinhala web sites to Unicode. Another student won an mBillionth award [14] in...
India for developing an Android browser that can render Sinhala and Tamil web content thereby bridging the rendering capability gaps of the Android OS. It is also important to note that some of the completed projects have been uploaded to official sites and have had significantly high downloads. Some of the projects have been accepted and up-streamed into their respective open source codebases. Some of the products developed as part of this course have also attracted interest by commercial companies [2].

X. CONCLUSION

The advantage of automating the panel composition process was evident when some external evaluators notified us of sudden changes in their time constraints or cancelled their commitment mere hours prior to the commencement of the end-of-semester project demonstrations. In this critical situation the algorithm facilitated rapid recalculation that produced an alternate optimal schedule.

In conclusion, we were able to successfully implement a synoptic assessment approach for the CS3202 course by addressing the challenges surrounding the constitution of expert evaluation panels by automating the panel composition and scheduling process. The automation was achieved by implementing an algorithm to assign evaluators with heterogeneous areas of expertise to project-specific evaluation panels, where the projects concerned involved many different technologies.

In future we plan to make the application that enabled us to deploy effective synoptic assessment via algorithmic constitution of review panels available in a generic form for others to use in similar courses if required.

REFERENCES


