Enhancing Learning with Hierarchical Project Groups in Courses with Multi-level Students

Scott Miller
University of Victoria
smiller@ece.uvic.ca

Abstract
Proposed in this work is the use of hierarchical project groups in engineering design courses with students across multiple years-of-study. Assigning group members roles as either managers, designers, or implementers, students can work on the same core course fundamentals while developing skills at different levels; less experienced students develop skills applying the core theory, while more experienced students can develop their professional practise and communications skills while demonstrating their understanding of the course concepts. As a case study, a course run three times between 2006 and 2008 is presented both using and not using hierarchical project groups. Student evaluations on this course show that such groups can increase student engagement and enjoyment.

1 Introduction
It is common knowledge that any course will have students with a wide range of expertise, but this is especially true in courses that appeal to students in each year-of-study in their degree. There are two primary reasons that higher-level students would be interested in a lower-level course: (i) Genuine interest in a subject not taught at a higher level, and (ii) for an easy mark in an acceptable elective. Even for those higher-level students interested the course material, learning the material from a lower-level course generally leads to an easy course and skewed marks in the course. As a result, if the course is made to be appropriate for the largest number of students, the least experienced and younger students may be made to suffer through a course beyond their ability. If geared towards the lower-level students, the course becomes trivial for the higher-level students. This may be further exacerbated when teaching courses that involve more hobbyist subjects, such as Web Application development, where student expertise may be less dependant on their year-of-study.

Also important in engineering courses is the application of professional practise principles in the classroom to help encourage students to get in the routine of applying them to all of their work. Such principles include working in organized project groups, projecting and logging work hours, documenting the entire design flow, and improving communications skills. Professional practise is a requirement for Canadian engineering accreditation, though many universities include professional practise courses, but do not explicitly include it in general course workloads [1, 2].

To create a course that encourages student learning regardless of expertise, professional practise principles can be applied to develop
different sets of skills based on a student’s level. As a case-study, an introductory course on Web Engineering is presented, where students were expected to complete assignments and projects in groups of mixed expertise levels. These project groups were structured as working groups in industry; higher-level students were assigned roles as project managers and senior software designers, while lower-level students completed the design, implementation, and testing of code for their project. This allowed the younger students to focus on the theoretical, design, and programming principles that formed the core of the course. The more experienced students, to whom the programming and theory were trivial, gained experience managing a project, designing software architecture, and completing proper engineering documentation. To ensure the higher-level students weren’t completing the lower-level work, the students assigned programming tasks were given oral examinations on every deliverable to ensure they knew what they were doing. Marks were distributed based on level, where more mature students lost marks if their groups failed or if their programming students did not understand course material to give them to have a vested stake in the course and encourage mentoring.

The contributions of the paper are as follows:

1. Proposes the use of hierarchical project groups to maximize student learning;
2. Presents the use of professional practise principles in student groups, and;
3. Provides a case-study with student evaluations.

The paper is organized as follows. Section 2 discusses the challenges to maximizing student learning in classes with mixtures of student levels and subject expertise. Section 3 presents hierarchical student groups and how they can be used to reinforce professional practise principles. Section 4 presents a case-study from a course taught over three iterations applying the techniques discussed in Section 3. Section 5 concludes the paper.

2 Challenges with Mixed Levels of Student Expertise

It is common sense that any course will have a mixture of levels of expertise with students bringing in their own personal experience and skill sets. Furthermore, students from varying levels-of-study may take a course out of interest or for an easy mark if they are already experienced in the subject matter. Depending on the course material, this can be further exacerbated with subjects, such as Web Engineering, which are common for hobbyists and therefore students’ common educational background cannot be directly used as a baseline for course materials; for example, in an introductory networking course, if most of the students in a class have experience with networking through work with Web applications, constructing a curriculum with the assumption that there is little-to-no networking experience may reduce the effectiveness of the course.

Having a class with a large split between students at different levels-of-study can create a number of challenges, such as:

- skewing class marks with students in upper levels easily attaining top grades,
- losing the attention of students taking the course for an easy mark,
- polarizing student workloads by addressing questions or subjects that are beyond the scope of class. This can be discouraging to students at the intended skill level, and
- creating an inconsistent curriculum with flow and continuity issues by trying to
accommodate various levels of technical competence.

As students’ attitudes towards academic pursuits change, many students are taking courses strictly for good marks and for less effort. This has been evidenced in course evaluations, as well as on-line with an increasing number of sites becoming available to rate professors on their easiness \(^3\) as well as on which classes take little effort to get a good mark \(^4, 5, 6\). This makes student engagement in courses with these multi-level splits a very relevant issue, as students are choosing courses based on the perceived effort required to obtain a good grade in the course over other factors, such as genuine interest. This may exaggerate the challenges listed above.

In addition to students looking for easy marks, there are many topics that may interest students, but that do not have a stream or large presence in a department. Many of these courses become electives for interested parties, even if they are set grade-levels below senior students. Consequently, even students who enrol in a class due to interest may end up taking very little from a course, as the concepts may be new, but they may be easier to learn by someone with more academic experience.

In order to minimize the effects of these challenges, hierarchical project groups are proposed for placing different professional practice and technical goals on students depending on their level-of-study. These groups are discussed in the following section.

3 Hierarchical Project Groups

When designing a course to accommodate students of various years-of-study, it is a tricky balancing act to ensure that the material doesn’t become too difficult for the students of the targeted year, while not becoming a free high grade for more experienced students. One possible solution to this problem is to design curriculums with a number of different criteria for each level of class, but this solution requires very careful planning of what essentially becomes a personalized topics course for the students involved. An instructor would have to design different materials for each level of student, and would have to organize class as to allow for questions from each tier of students without alienating the other tiers. This is a very difficult and time-consuming method, and is not considered any further.

Another solution to this problem is to target different skills for each group of students to develop based on their level of study. This approach allows for one set of technical course objectives, while weighting the type of work performed by each group of students differently. Although there are many criteria that could be re-weighted, core deliverables such as code, mathematics, etc. should not be drastically altered to avoid the scenario presented above. What can be changed easily, however, are the skills that are expected to be developed over a student’s degree, such as their professional practise skills; i.e. communications skills, team work and management, budgeting and time estimation, work logging, etc.

In order to clearly define what is expected by students at each level, hierarchical project groups representing a real-world working team have been used. At a basic level, these groups consist of positions mirroring those found in industry: (i) managers, (ii) senior designers, (iii) implementation engineers, and (iv) test engineers. Students are then placed in these positions based on their year-of-study and experience with the course material. For example:

Managers – Upper-year students have the role as management to coordinate project group meetings, oversee the design and success of the project, document the group’s activities, and take responsibility for the group’s overall success.

Senior Designers – Upper- or Mid-level
students have the role as senior designers to come up with the overall system partitioning, functionality, modularity, interfaces, etc. They work with the manager to realize the overall design, and with the implementation engineers to assign implementation tasks to them.

Implementation Engineers – Lower-level students perform the bulk of the intended course material at a design level. This allows them to learn the fundamentals of the course with out-of-class help from their more experienced peers.

Test Engineers – Each of the above levels have their own vested interest in testing their deliverables, as the implementation engineers are expected to ensure their code/hardware/etc is working and will interface correctly with their peer’s work. Both senior levels test their projects, as their mark is also dependant on how their project functions and have more experience to catch errors that the lower-level students may have missed.

With student roles assigned, it is important to note that the technical deliverables for the course have not changed; any individual assignments, midterms, exams, etc. are expected to be satisfactorily completed by students at all years without major modification. For the project, all students are expected to take part and collaborate with all roles, as well as by adding their input. For example, all students will receive a mark on the manager’s documentation, and therefore all group members should be interested in ensuring the manager is on-task. The weighting of the mark, however, is varying amongst roles in the group. For example, the manager students may receive a 40% of their project mark through the design meeting their specifications, their documentation, presentation and leadership on the project, etc.

Lower-level students may receive 10% of their mark based on the same factors, with the other 30% being placed on the functionality and clarity of their code/hardware/etc design and implementation process.

These roles allow the course content to remain standard, while focusing on developing different skills at each student level. However, what these groups do not yet address are the original problem of higher level students having an easier time, and they possibly miss out on an opportunity to encourage mentoring across student years. In order to maximize the benefits of having upper- and lower-level interacting over the same project, the upper-year students should be given an interest in helping the others. To accomplish this, the upper-year students marks have been scaled to reflect the understanding of course material of their lower-level students. This creates an atmosphere where the managers and senior designers have to check in on their group members to assess their course knowledge and help them do better with the course material itself. This also eliminates the problem of having a manager quickly write documentation and leave the project to their peers without adequate work, and it also eliminates the manager from doing all the work leaving their peers to lose practical experience. It is the senior member’s role to help ensure their peers understand the course content.

The groups proposed above can encourage the development of professional practise skills, foster student-student mentoring, and can enhance learning opportunities for upper-year students looking for an easy course. These groups have been applied in a course at the University of Victoria, and reflections on the course have been provided below as a case study.
4 SENG130: Introduction to Software Engineering – A Case Study

As a case study, SENG130: Introduction to Software Engineering is presented below and was run through three consecutive Spring terms from 2006-2008. This first-year course teaches basic Software Engineering principles, such as writing clean, commented, maintainable code, modular system design, following an iterative design process, etc., and uses Web Application Engineering as a practical framework to learn these principles. As one of the only classes in the program that explicitly deals with Web Engineering, the class drew a nearly equal amount of students from first through fourth year of the program. Furthermore, as web application design is freely accessible and popular by hobbyists, many students in all years of the course had very different backgrounds on the course material.

In order to fairly organize students into groups with fair roles, a survey was circulated in class to determine the expertise and year of every student. Student groups and roles were assigned by the instructor, which allowed for a good mix of expertise in each group and allowed for lower-level students with sufficient design experience to fulfil the senior designer role, and so on. In general, first and second year students were assigned to be implementation engineers, third year students were assigned as senior designers, and fourth year students were assigned as managers.

As discussed in Section 3 to ensure that the project leaders (the managers and senior designers) don’t produce most of their group’s code to allow the lower-levels to the practical experience, there is the potential for mark scaling to occur if their group mates do not understand their group’s code. As well, if a group’s code is well written and clear, but junior members of the group do not clearly understand the course principles marks may be deducted from senior members, as their group does not understand the code they are in charge of verifying. An example of the weighting for each role and potential scaling is shown in Table 1.

Upon completion of the group projects, both anonymous course surveys and official evaluations were conducted to determine how effective the hierarchical groups were from the student’s perspectives. In Spring 2006, non-hierarchical project groups were used and students tended to work with their peers which led to a sharp discrepancy in project quality and marks. Student feedback showed that the upper-level students who received high marks provided positive feedback, whereas lower-level students reported finding the class harder and were frustrated to see the upper-level students easily receiving easy marks. In Spring 2007, hierarchical project groups were used and the course registered a higher level of student satisfaction across all years-of-study, although mentoring and group success for senior students were not enforced. As a consequence, lower-level continued to report the course as very difficult, but less than in the previous iteration as they had more experienced students helping design the project, which provided them with guidance they were lacking in the previous year. As a positive benefit as well, the quality and scope of the student project improved along with the reported interest and utility of the course. In Spring 2008, hierarchical project groups were strengthened

<table>
<thead>
<tr>
<th>Student Role</th>
<th>Report</th>
<th>Software Design</th>
<th>Code</th>
<th>Scaling</th>
</tr>
</thead>
<tbody>
<tr>
<td>Manager</td>
<td>50</td>
<td>30</td>
<td>20</td>
<td>-10% /student</td>
</tr>
<tr>
<td>Senior Designer</td>
<td>30</td>
<td>50</td>
<td>20</td>
<td>-5% /student</td>
</tr>
<tr>
<td>Impl. Engineer</td>
<td>30</td>
<td>30</td>
<td>40</td>
<td>N/A</td>
</tr>
</tbody>
</table>

Table 1: Example marking scheme from 2008.
Table 2: Results of Student Evaluations from 2006-2008.

<table>
<thead>
<tr>
<th></th>
<th>2006</th>
<th>2007</th>
<th>2008</th>
</tr>
</thead>
<tbody>
<tr>
<td># of students</td>
<td>28</td>
<td>18</td>
<td>25</td>
</tr>
<tr>
<td>1st year</td>
<td>12</td>
<td>8</td>
<td>9</td>
</tr>
<tr>
<td>2nd year</td>
<td>7</td>
<td>4</td>
<td>8</td>
</tr>
<tr>
<td>3rd year</td>
<td>5</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>4th year</td>
<td>4</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>Difficult?</td>
<td>4.2</td>
<td>3.9</td>
<td>3.1</td>
</tr>
<tr>
<td>Practical?</td>
<td>5.0</td>
<td>5.0</td>
<td>5.0</td>
</tr>
<tr>
<td>Groups worked well?</td>
<td>2.8</td>
<td>3.7</td>
<td>4.1</td>
</tr>
<tr>
<td>Group members fulfilled roles?</td>
<td>2.3</td>
<td>4.0</td>
<td>4.7</td>
</tr>
<tr>
<td>Project level appropriate?</td>
<td>2.1</td>
<td>3.7</td>
<td>4.2</td>
</tr>
</tbody>
</table>

with the requirement that senior group members take some ownership over the junior members' performance. Student evaluations once again showed student enjoyment increasing in the junior years without a decrease in the upper-years. The feedback therefore shows that the senior students did not find it unfair to be marked upon their group members' performance. The numerical figures this analysis is based on is shown in Table 2.

Table 2 shows the percentages of respondents answers on a scale of 1 to 5 with 1 strongly disagreeing and 5 strongly agreeing with the question asked. Sample sizes of respondents of each class are also shown, as are the number of students of each level in the class.

5 Conclusions

Many courses have a sharp divide or large spread of student knowledge, which is exacerbated by upper-level students taking earlier courses as electives for easy marks, or out of interest in the course. This can lead to skewed marks or alienated and inattentive students. Hierarchical project groups can be used to teach a single curriculum, but focus the marks in different areas for students depending on their experience and expertise. Over three iterations of a mixed-level course, project groups were found through student surveys and evaluations to improve student engagement, cooperation, and learning at higher levels of the course without sacrificing the focus and understanding of the lower levels.

References


