INTRODUCTION
The authors have undertaken the design and development of a Digital Prototyping course for final-semester students in the Mechanical Engineering Technologies [MET] program at SAIT Polytechnic. The purpose of the course is to equip students with an ability to use advanced solid modeling and analysis tools in a digital environment to limit the number of physical iterations required in a product development process. Participating students present in their fourth semester of study with a basic understanding of techniques associated with solid modeling while undergoing a capstone project in parallel with the course.

The challenge in developing the course is twofold – first a learning environment that facilitates guided exploration of the advanced functionality associated with a software package, in this case the Autodesk suite of product design applications, must be established. Second, the continuously evolving nature of the software must be accommodated. The latter is a key problem for intermediate-to-advanced software courses, as the software enhancement cycle is often shorter than the course frequency.

We proposed a framework in which we could explore the viability of mixed instructor- and student-led learning, relying on the students’ enthusiasm for self-discovery of new software functionality as it related to their specific needs and interests.

METHODS
Course work was divided into two projects. The first focused on the use of Autodesk Inventor to re-design a detailed instructor-provided model of a bicycle brake lever based on student-chosen criteria. Examples included re-design for sustainability, reduced weight, enhanced structural strength, and alternate attachment mechanisms. Students were encouraged to select their own project [potentially related to the capstone project work already underway] for the second course project, which integrated the acquired expertise in Inventor with the use of Autodesk Showcase to produce professional quality renderings and animations. Students were also encouraged to explore other analysis and simulation packages available in this suite if it related to their particular project.

Instruction on various aspects and functionality of the software, as it related to relevant prototyping techniques, was centered around Autodesk’s industry-developed, time-limited teaching technique that relies on short lecture-demo-activity cycles. The instructor provides a short lecture on the technique or tool. Following this, the instructor demonstrates this technique in the software. Students are then invited to complete a short activity that allows them to try the technique themselves on a model provided by the instructor. This structure was also presented to the students to provide a straightforward format in which they could present their own findings.

A typical two-hour class consisted of several of these short ‘loops’ followed by a work period wherein students were provided time to engage in their projects and solicit feedback and individual, project-specific coaching from the instructors.

Evaluation was carried out consistently and informally on a grading sheet week-to-week. Students were asked to show an aspect of their project that demonstrated a level of aptitude with software as defined by a provided grading rubric. Examples of these categories included the ability to generate professional quality parts, assemblies, sketches, and to perform analysis on these parts. Grades for each of these categories were assigned based on the student’s demonstrated ability to produce work on par with the criteria established in the rubric. As such, the student could track their success and improvement through the course by reviewing the weekly grading forms. The instructor assigned final, overall grades for the projects upon final submission.

RESULTS
From the outset of the course the students met the method presented with considerable anxiety and hostility. The choice of software, analogous but not identical to a package they had previously learned, caused considerable stress. Claims were frequently made that tools to which they had become accustomed were not available in the chosen software suite. The use of provided, relatively complex models for the students to explore and alter also seemed to inhibit exploration, as the students appeared unable to imagine how the object could be altered or re-designed.

The chosen grading scheme was also a source of some uncertainty, as the students had become accustomed to a subtractive model of grading, wherein the grade declined from a perfect score as mistakes were made throughout the course. As this course used an additive grading scheme, students appeared to feel helpless and unable to find a path through which they may attain any marks whatsoever.

The class settled in to equilibrium by the third week, at which point students could be encouraged to explore a particular aspect of the software with some verbal instruction and guidance as to what might help them achieve some marks. At this point, the class appeared to stratify into three types of students. A small group (10-15%) of students embraced the structure and explore the software in the context of their own design project as intended. These individuals took considerable initiative in exploring specific aspects of the software’s functionality using texts provided help files and online resources. This group occasionally took the opportunity to present their discoveries to the class in small micro-presentations or loops.

A second set of students (approx. 60%) worked piecewise towards achieving an acceptable mark. These students could only be encouraged to produce work based on a challenge-results model, where verbal instructions about what had to be achieved in the context of a particular project were given by the instructors, and the student worked to achieve this result.

The remaining students typically went ungraded for the majority of the semester, choosing to learn the software entirely on their own or to complete the course work in a compressed period before the due date. This group produced the most surprising and varied results, as occasionally the work produced was exceptional, focused and showed the desired level of engagement with the software. Others in this group produced predictably poor or little work leading to late-stage panic and appropriately poor grades.

DISCUSSION
The methods proposed, particularly the evaluation techniques, might have been too radical a departure from what the students in the program had come to expect. Integral to the success of the course was the assumption that the complexity and functionality available in the software packages could capture the students imagination, particularly as it related to projects that were already underway in other parallel courses, to the extent that they would be motivated to discover, learn and share esoteric functionality available in the product design suite.

The observed results indicate that the students, even in later stages of their design education, are primarily grades-motivated to the extent that they were incapable of using the available class time to explore the software in the context of a design problem. Paradoxically, the anxiety associated with not performing to a high level (achieving an ‘A’ grade) frequently inhibited a student from engaging in any advanced exploration of the software at all -- the very mechanism through which the students could achieve a high grade. The prevalent perception appeared to be that time could not be wasted on self-guided learning, as the time invested might not lead to predictable grades-based results.

A second iteration of the course has been scheduled in which a revised approach to this structure will be employed. It was clear from the results that most students could engage in self-directed learning once they had found a suitable application for the software and had achieved some success related to their efforts. Considerations for improvement include a more rigid demonstration-based approach to applying existing modeling skills to the software suite, goal-based projects or course objectives with clearly defined examples that show ‘A’ level work while leaving the path to achieve that result open for exploration.