122nd ASEE Annual Conference & Exposition
June 14 - 17, 2015
Seattle, WA
Making Value for Society
Paper ID #11538

A Cloud-based Tool for Assigning Students to Projects

Dr. Smitesh Bakrania, Rowan University

Dr. Smitesh Bakrania is an associate professor in Mechanical Engineering at Rowan University. He received his Ph.D. from University of Michigan in 2008 and his B.S. from Union College in 2003. His research interests include combustion synthesis of nanoparticles and combustion catalysis using nanoparticles. He is also involved in developing educational apps for instructional and research purposes.

Mr. Brad Joseph Johnson, Rowan University

Brad Johnson is a junior chemical engineering major and computer science minor. He is the president of the Rowan University Chapter of AIChE and a member of Tau Beta Pi. He was the recipient of AIChE Delaware Valley Section's 2014 Outstanding Sophomore of the Year award and is a three-time Black Box Cooperation Yost Scholar of Excellence. In the future, he intends to pursue a PhD in Chemical Engineering and become a professor.

A Cloud-based Tool for Assigning Students to Projects

As a part of the overall engineering curriculum, senior capstone projects provide the necessary opportunity for engineering graduates to apply their technical skills towards solving contemporary, open-ended, and challenging engineering problems. At Rowan University, both juniors and seniors engage in such hands-on projects across their last four terms in the engineering program. In the recent years, over 80 projects have been presented to over 250 students – demanding a clear need for a software solution to manage student assignment. A cloud-based platform was developed for managing projects and assigning students. This solution used Google Forms to generate a list of projects with details for students to review and accept student preferences. An algorithm that assigned students to projects using multi-level criteria processed the projects and student preferences. Replacing the previously laborious exercise of assignment was a primary motivator for this effort. The algorithm utilized Google Apps Script's ability to generate and manipulate data objects while having full access to Google's variety of cloud services. Furthermore, to create a robust platform for improved student assignment, new project and student requirements that could not previously be accommodated were added. For the Fall 2014 term, the new automated approach was compared to the old manual sorting method by each of the four engineering departments. Beyond the obvious benefits, further advantages of the automated assignment are documented here. This paper describes the automation approach in detail and provides insights for implementing such a platform elsewhere. With the addition of scripted algorithms, several frequently used tasks can be automated with varying degrees of complexity. The effort described here is a robust example of the utility that cloud-based services provide to the academic profession.

Introduction

Engineering Clinics are a series of courses that span the full 4-years of core engineering curriculum at Rowan University. While the primary focus of clinics is providing students with hands-on learning, the freshman and sophomore engineering clinics also involve teaching basic engineering skills to the students such as measurements, problem solving, and parametric design, to name a few[1-4]. Junior and senior engineering clinics, on the other hand, provide students with the opportunity to work on contemporary 'real world problems' that include scientific research, industry-sponsored projects, and engineering design competitions. The junior and senior clinics, specifically, provide an ideal platform to broadly address the ABET A-K criteria with varying degrees of rigor[5]. The last two years of engineering clinics are not unlike typical senior projects common within many engineering programs. These junior and senior engineering clinics are specifically unique in the assignment of students to projects, among other things. The assignment of students to projects is driven by accommodating individual student interests in projects. During the creation of clinics, the college of engineering strongly felt the freedom of students to select their own projects will contribute to the overall success of their participation. The projects are pitched by faculty at the beginning of each term. In other words, students can choose to participate in four different projects for the last two years of their clinics. Therefore, the assignment must account for the individual preferences at the same time satisfying the needs of faculty for their respective projects.

In the past, the assignment task was completed manually using paper-based forms to satisfy student and faculty criteria. However, dramatically increasing enrollment necessitated a degree of automation in the management of assignments. For instance, in the Fall 2014 term over 100 projects were pitched by Mechanical, Electrical, Civil, Chemical, and Biomedical Engineering faculty to over 300 students. With the trend in enrollment expected to continue, a complete automated assignment solution was envisioned that satisfied the past assignment criteria while affording new assignment requirements and constraints. The work described here elaborates on the design and development of an automated solution to replace the existing assignment approach. The solution presented here can be easily extended and applied to other academic settings where automated criteria-based group assignments are needed. This paper also presents the results of this automation and its impact.

Background and Approach

Existing Assignment Approach

An engineering faculty member typically manages 2-3 distinct projects involving 2-5 multidisciplinary and multi-year students for a single semester. To ensure the teams are generated based on student interest in the project, the project managers are required to pitch the project to the students on the first day of classes. Here the project managers introduce the scope of the project for the current term along with other key project details that includes the funding source and the requested number of discipline-specific students. Upon completion of the pitches, students submit a project selection form that ranks projects according to their interest and prior experience. Students can submit up to 3 project choices within their engineering discipline and 2 more projects for out-of-discipline choices. At the same time, each faculty member shares their list of projects, each specifying the maximum number of students required from each discipline. Once all the input is collected, discipline managers (DM's) from each of the engineering departments manually sorted students according to the preferences and project constraints. Figure 1 attempts to visually capture the existing assignment approach.

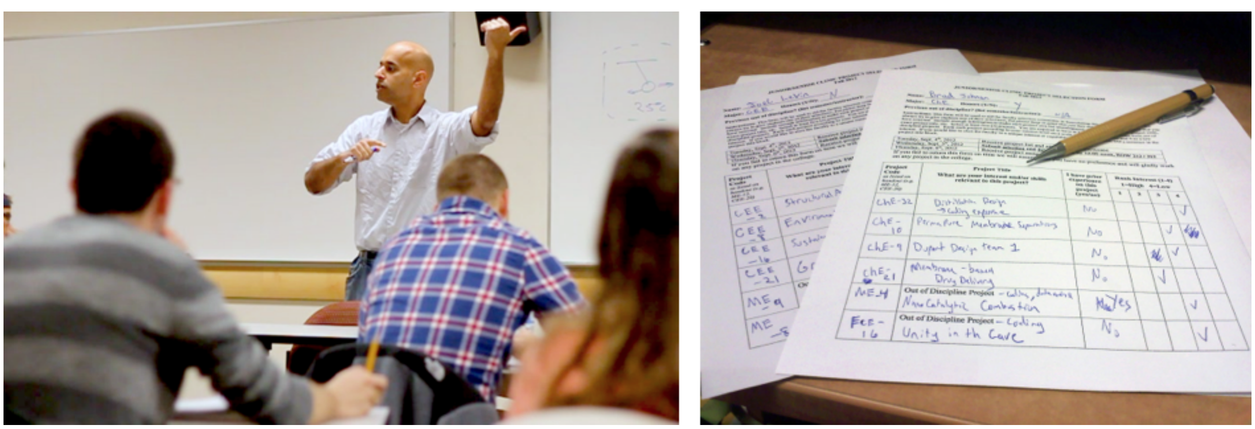

Figure 1. The existing approach involved faculty pitches where faculty members described their project's needs (left) and students submitted their preferences via paper-based formed (right).

Challenges

While the existing approach has worked well since the establishment of clinics in 2000, assignment became progressively more challenging with the growth of the engineering program. This process was further complicated by the faculty's requests for specific students with pertinent skills and a desire to prioritize the assignment of projects that are externally funded or industry sponsored. On the other hand, real-time pitches, where faculty had the luxury to describe each of their projects and their requirements, became gradually squeezed to 2 minute presentations, leaving little room for discussion. Additionally, the available space in the engineering building would soon be unable to accommodate all the participating students within each discipline during the pitching process. All these presented major challenges that needed to be addressed along with other efficiency related considerations. Overall, a new assignment solution was desired that accommodated the original mission of engineering clinics, i.e. the students have the freedom to work on projects that interest them, while meeting the project- and faculty-driven student needs.

High-Level Solution

To address the major challenges described earlier, a digital solution that handled the complete assignment process was considered. The complete assignment process was not confined to the final student assignment but the initial faculty input with regards to the project requirements. Therefore, the five-step clinic assignment process composed of the following sequential steps:

1. Faculty Input
2. Project Pitches
3. Student Input
4. Student Assignment
5. Assignment Broadcast

These steps were typically handled by DM's from each department who coordinated with their faculty members. The overall process was managed by the clinic coordinator. With a new solution, majority of these steps would be managed centrally by a cloud-based platform with minimal input from DM's. To begin with, the following design criteria were adopted to develop the new solution:

- Universal accessibility
- Platform independence
- Low barrier for entry
- Preferably free
- Dynamic
- Expandable

It was decided early on that a cloud-based solution would satisfy many of the design criteria. Already, aspects of the clinic assignment process were recently replaced by Google Forms. As a result, initial designs involved the use of Google Forms which subsequently evolved into the final design solution using the Google Apps Script platform[6].

A number of competitive approaches are detailed in literature. A prior effort at the University of Calgary used Blackboard to achieve a comparable solution, however aspects of that solution fails to meet the design criteria described here[7]. A semi-automated approach using a search algorithm was taken at Olin College of Engineering on a significantly smaller sample space and achieved moderate success but failed to improve the time of assignment[8]. Alternatively, a genetic algorithm was investigated at the University of South Carolina for optimizing student assignment[9]. While an optimization algorithm can prove successful, the intent of this project was to replicate the assignment choices previously developed during the manual sorting process. In other words, the new solution was an attempt to formalize the assignment criteria used by each DM and use it within a decision-based algorithm.

Solution Details

Next, the final solution is described with numbers corresponding to the five-step clinic assignment process listed earlier.

1. Faculty Input

A custom Google Form was created that enabled faculty to propose projects they wished to manage each semester. Faculty can submit multiple project proposals but only projects populated with enough students were set to 'run' for the term. Projects are populated based on the assignment algorithm discussed later. Each faculty project submission required information about the project purpose and scope to generate a project pitch spreadsheet. The source of a project's funding was an important piece of information provided by the faculty. Projects that are externally funded, such as from a federal grant from NSF or NIH or from an industrial partner must be populated with the desired amount of students. Internally funded projects depend upon students' interests to determine which projects were selected to run. Project managers also specified the minimum number of students necessary for their project to run and the maximum number of students their project could accommodate. These bounded the student assignment process and assisted in determining which projects were 'pruned' - set to not run that term. Further refinement to the assignment was accomplished by allowing the project managers to request students for their project. It was assumed that these students showed prior interest in a project to the faculty. If the students requested by a project manager also selected that particular project as their top choice, then they were given top assignment priority for that project. Before the implementation of the new solution, the DMs were responsible for collecting faculty projects within each engineering discipline and therefore a centralized list of all the proposed projects for the term did not exist. Therefore, students only referred to a discipline-specific project list during the project pitches.

2. Project Pitches

After each project submission, a project entry is added to a working list of all projects - the project pitch spreadsheet. The pitch spreadsheet, represented in Fig. 2, is a visual and interactive presentation of each available project. Each project is color-coded to match the discipline it is housed under. Additionally, students can filter out projects that do not specifically needed their discipline. Each project displays the maximum number of students each project required, as specified by the project manager. A short project description introduces the project at a high-level, allowing students to easily filter out projects that do not interest them. For students interested in greater detail, a "details" section contains more specific project information in addition to hyperlinks to papers, videos, or presentations, similar to the approach taken by Brigham Young University[10]. While the new approach eliminated each professor being able to describe their projects in detail, the pitch spreadsheet delivers a wealth of information on each project which the students can explore, if interested.

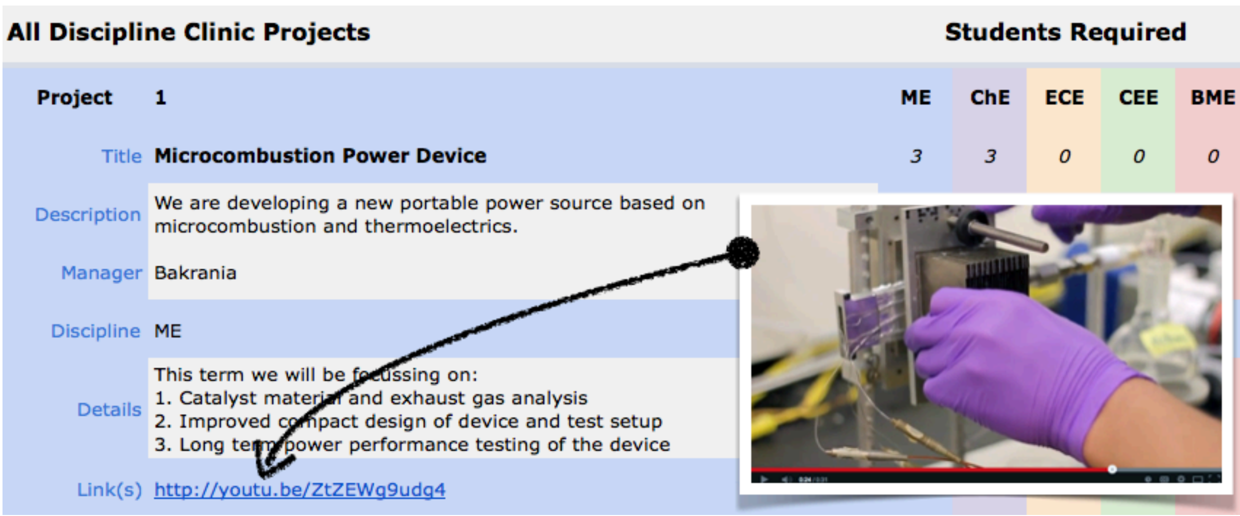

Figure 2. A sample of the project pitch spreadsheet. The inset provides an example of the content a project pitch can supply, for instance, a video describing the project.

3. Student Input

A separate Google Form was designed to allow students to provide their personal information and an ordered list of 8 preferred projects. The students selected projects from a dynamically updating list that is linked to the faculty project submissions (see Fig. 2). Students were only allowed one submission, but could update their response using a hyperlink emailed to them after their submission of the form. Student submissions were used to assign that student to a project using the algorithm discussed later. The two most important pieces of personal information collected were the student's campus username ID and their year (either junior or senior). Additionally, Students specified up to eight projects in the order of preference and specify whether they have prior work experience on that project. Students cannot specify a project multiple times. An engineering student was able to select a project from any discipline, e.g. Mechanical Engineering (ME), Chemical Engineering (ChE), Electrical and Computer Engineering (ECE), or Civil and Environmental Engineering (CEE). After a student submitted their form, they receive an email, confirming all their information.

4. Student Assignment

Once all student and faculty information was collected, a multi-stage assignment algorithm using Google Scripts processed both data sets to generate a complete list of plausible student to project assignments. An example of a single student and project entry is provided in Table 1. The content provided in Table 1 serves as an input to the assignment algorithm. The objective of this algorithm was to satisfy the minimum operational requirements of all funded projects, while considering the project interests of students.

Table 1. The essential information collected in the student input (left) and faculty input (right) forms detailed in the corresponding sections above. Items listed using square brackets, i.e. [item], represent variables specified by the students or faculty.

Student Object ID [username]	
☐ Junior	☐ Senior
	Prior Experience
[Project Preference 1]	☐
[Project Preference 2]	☐
[…]	☐
[Project Preference 8]	☐

Project Object ID [number]		
☐ Externally Funded		
Students	Min	Max
ME	[No.]	[No.]
ChE	[No.]	[No.]
ECE	[No.]	[No.]
CEE	[No.]	[No.]

An iterative, priority-based assignment algorithm was used as a base for each assignment stage. A student's priority is a dynamic value that takes into account their year (Junior or Senior), prior project experience in their current most preferred project (Experienced or Inexperienced), and the time of their form submission. Table 2 provides the specifics of the priority list described here. A priority group is considered to be all students of the same year with the same experience in their current top choice project.

Table 2. A breakdown of student priority groups for students to project assignment

Priority Group	Year	Project Experience
3	Senior	Experienced
2	Senior	Inexperienced
1	Junior	Experienced
0	Junior	Inexperienced

To rank students within their priority group, the time of their project submission is used. Earlier submissions are given greater priority. Randomization was considered within the priority groups; however, it was determined that a concrete order should be established to maintain a consistent, replicable, and objective student-to-project assignment. For example, the first student to be assigned was the first senior to submit their preferences with experience in their top choice clinic and the last student to be assigned was the last junior to submit their preferences with no experience in their top choice clinic. A student's priority group may change during the assignment process because their second, third,…or nth project may have a different experience level than their first project.

The standard assignment algorithm takes an iterative approach to assigning students. The number of iterations is dependent on the current state of the assigned student population. In a single iteration, the first step is to sort the students in order of priority from highest to lowest. After prioritization, the algorithm moves sequentially through three stages of assignment conditions, namely: student-professor agreement, externally funded assignment, and internally funded assignment. After all three stages, the algorithm assesses which projects have met and have not been assigned their minimum number of in-discipline students. The algorithm then selects a maximum 5 such projects based on several criterion and 'prunes' them, removing them from the algorithm's next iteration. These critical assignment steps are elaborated below.

Student-professor agreement assignment was designed to assign students to projects where the student and project managers had a mutual interest in working together. This component uses the campus username ID of students specified by professors to check each students top-choice project. If there is a match, the student is immediately assigned to their top priority project, disregarding all restrictions. If not, then the student is assigned to projects in the subsequent steps.

Externally funded (EF) assignment was designed to ensure sponsored projects received the minimum number of students of its corresponding major, thus the project could never be 'pruned'. EF projects are not discretionary, so they must have the appropriate number of students to run. Students with externally funded projects in their list of preferences will be the first assigned to these projects. The algorithm checks each student's most preferred project first. If it is an EF project, then the student is assigned to it. After all students' first preference projects have been examined, it will look at each student's second projects preference after re-prioritizing the list based on second project experience instead of first project experience. This repeats until either all of the y students preferences are examined or the minimum number of in-discipline students for each EF project has been met. If an EF project has not met its minimum, then students are randomly assigned to it, starting with the students from the lowest priority group.

Internally funded (IF) assignment was designed to assign students to projects the they were most interested in, regardless of discipline. Students are prioritized and then traversed identically to the EF assignment described above without checking each project's funding source. Students are assigned to projects that have not exceeded the maximum number of students in their major, as specified by the project manager. This procedure continues until either all students have been assigned to a project, or all students' project preferences are examined.

Quota evaluation was designed to identify projects with an insufficient number of assigned students so they may be pruned at the start of the next iteration. A project is considered 'under-assigned' if it had less than one student under its minimum in-discipline requirement. These conditions were selected in order to prevent projects that were close to having a sufficient number of in-discipline students assigned from being pruned. The decision to prune or reassign these projects were left to the DMs. To prevent the removal of projects from impacting the algorithm's reliability, only one project per discipline can be designated to be pruned. The designated project was the least populated project among the projects that failed to meet the quota. This policy minimized the number of students that needed to be reshuffled and prevented the unnecessary removal of projects in the long run.

Prune processing was designed to cleanly remove projects from the sorting algorithm after the quota evaluation identified them as under-assigned. Prune processing removes each identified project from the list of available projects, and it unobtrusively removes each project from all students' preferences. This allows the above assignment algorithm to take place iteratively, with each iteration having a smaller pool of projects that better correspond to the students' preferences. Funded projects will never fail the quota check because EF assignment ensure they have a sufficient number of students assigned, thus they will not be pruned. The algorithm reiterates through the above assignment procedure until there no new projects fail the quota evaluation. This exit condition was chosen because it indicates that all projects that have not been pruned have been assigned an appropriate number of students. This completes the assignment procedure and the algorithm proceeds to broadcast the results.

5. Assignment Broadcast

After the algorithm completes assignment, it produces a visual final assignment spreadsheet (see Fig. 3). The spreadsheet can easily be filtered by project discipline and the color-coded students clearly indicate which major they belong to. This proved to be an effective approach to broadcast the assignments, allowing DM's to modify and complete the assignment process without any transitional work. For Spring 2015, each DM was provided two sheets to work from: results of the initial iteration and results of the final iteration.

Num	Name	Manager(s)	Dept	Students		
1	Microcombustior	Bakrania	ME	Langlois, Roy | S	Reilly, Matthew |	luong, raymond |
2	App Developmer	Bakrania	ME	Sewnig, Jonatha	Bede, Daniel | JL	Boszczuk, Brent
3	NJ DMAVA Ener	Everett	CEE	Brangman, Shal	McCarthy, Sean	Ziegler, Jordan |
4	NJ DMAVA Ener	Everett	CEE	Consolloy, Matth	Gulotta, Joseph	Parisi, Nicholas |
5	Engineers Witho	Everett	CEE	McCollum, Evan	D'Eustachio, Mat	Polhemus, Matth
6	Aircraft Hangar F	Dusseau	CEE	lamperti, trevor |	Pompilio, Kyle |	Ledig, Jenna | JL
7	Materials for The	Dahm	ChE	Huitt, Tarynn | SE	Lewis, Michael |	SENIOR | ChE
8	Airport Pavemen	Sukumaran	CEE	Short, Andrew |	Hillis, Lauren | SI	Brevogel, Shelby

Figure 3. Samples of the assignment output listing projects and their assigned students. The output is color-coded based on project and student discipline, respectively.

Results and Discussion

The algorithm satisfied all the high-level objectives of this endeavor; however, the overall success of the algorithm must be evaluated by how well it refines the clinic assignment process. One way to assess the success of the new solution is to look at the reduced role of DM's in facilitating the five-step assignment process for each discipline. The following is a list of key tasks that were eliminated or enhanced by the solution described here.

- **Building project list.** DMs are no longer required to collect project proposals, organize project pitches, and distribute/maintain project lists. In the past, DMs often updated their list shortly before the pitches - to account for any last minute additions and modifications.
- **Maintaining accuracy.** Digital submissions and the subsequent ability to correct/edit submissions also eliminated the need for DM's to correct individual faculty or student submission errors or implement changes.
- **Collecting student preferences.** The DM's are no longer responsible for collecting student preferences. Instead a Google Form fully automates data acquisition. Submitting and pitching projects is fast, dynamic, and accurate. In fact, the new solution allows collection of student information that was previously unaccounted for.
- **Faculty request process.** In the past, DM's managed a list of each faculty's requested student. This operation is fully automated and most importantly has been improved by the student-professor match criteria that confirms if the requested student is also interested.
- **Final Assignment.** The assignment algorithm provides a strong guidance to the ultimate assignment. A manual student-project sort by each DM can take up to 8 hours, as opposed to few minutes that the algorithm takes to generate an assignment list. While this task has not been completely eliminated (as elaborated below), the authors continue to refine the algorithm to match the manual assignment procedure used by the DM's.

In the Fall of 2014, the DMs' manual assignment results were compared to output from a preliminary algorithm. To compare the manual assignment results to the algorithm's output, an agreement heuristic was developed. The percent agreement of an assignment was evaluated by summing up the number of students who were assigned to identical project in both results, subtracting the number of students that were in the manual sort but not the electronic, and dividing the difference by the total number of students in the electronic sort. Students not sorted by the algorithm are accounted for in a numeric difference between the two assignment procedures. If the overall agreement was 50% or greater, then the assignment algorithm was considered a refinement. In general, the natural subjectivity of the DMs in addition to criteria not accounted for in the algorithm, such as the work ethic of students, would result in deviations between the objective automatic assignment and the manual assignment. Additionally, this algorithm was never designed to generate a finalized output. Instead, it was develop to serve as a platform to improve the efficiency of the assignment process. Further modifications by each DM was expected after they receive the algorithms output.

When compared, an overall weighted average of 51.5% agreement existed between the two assignments, where the weight of that discipline was determined by the number of students sorted in that discipline. For comparison, the unweighted average agreement was 50.7%. The standard deviation was determined to be 13.8%. Detailed values are presented in Figure 4.

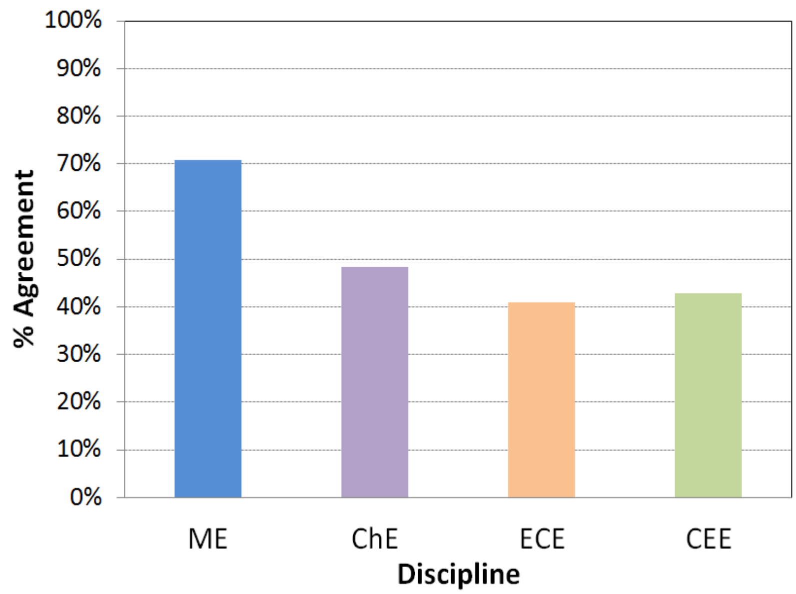

Figure 4. Percent agreement breakdown by discipline.

Most notably, the mechanical engineering disciple had the greatest agreement between the two assignments, suggesting the assignment approach taken by the ME DM was likely the most similar to the algorithm. This is further supported by the ME DM taking an active role in aiding in the design of the assignment algorithm. An aspect that reduced the agreement was instances of 'singular inversions' in assignments. There were multiple instances where disagreement was caused by a simple swap between two students. Each swap would cause substantial collateral changes within the algorithm's output, making quantifying the true extent of these 'singular inversions' challenging. Additionally, subjective groupings such as student peer groups and redistribution based on work habits were also dominant in the manual sort. Two other significant changes made by the DMs that reduced agreement was the revival of pruned projects and the assignment of a student to a more preferred, unfunded project over a funded project. While these artifacts yield ~50% agreement between the manual and algorithm assignment, the solution presented here applies across all discipline as an objective and criteria-based assignment process. Furthermore, the algorithm criteria were designed to prioritize student interests over project needs which is important for a productive relationship between faculty and students.

The algorithm continues to evolve as constructive feedback from the DM's is taken under consideration in order to better fit the manual sorting process. It is envisioned to yield an assignment output that combines the strengths of automated process with the nuances associated with the manual sort. These nuances may include the ability of DM's to 'run' a previously 'pruned' project or accommodate students who forgot to share their preferences before the deadline. For such cases, a fully automated process is inadequate considering the data supplied to the algorithm is insufficient. To summarize the outcome of this work, we must acknowledge the tremendous benefit this platform provides for collecting, processing, presenting, and maintaining data in a centralized location. We recognize the algorithm does not eliminate some level of intervention from faculty; however, this intervention has been dramatically reduced when compared to the five-step procedure used in the past. As a result, the work presented here demonstrates a robust use of cloud-based services to address academic needs, especially for data collection and management. Several other resources exist that use other Google services in similar fashion[11]. Beyond its use as a project assignment tool in an academic setting, the platform is highly effective in situations where criteria-based grouping is desired; especially involving a large population of member elements. The authors intend to freely distribute the final version of the platform for application elsewhere.

Conclusion

To ensure an optimum outcome of a student engineering project, an effective assignment process must balance two basic requirements. First, to populate projects based on student preferences. This is especially true for junior and senior engineering projects, which require higher-level thinking and professional skills. Second, to enroll students for projects that have been designed to primarily serve faculty agenda. Typically, the projects are inspired by existing or potential government or industry funding. However, as the number of projects offered and the student population increases the assignment process becomes markedly demanding. To address this challenge, a cloud-based automation solution was developed. The solution, using Google Services, proved adequate for the overall universal assignment process. Further minor refinement is required to better match the manual discipline-specific sorting. The greatest advantage offered by the cloud-based solution is its ability to: (a) collect detailed information from faculty regarding the projects they are offering, (b) present the projects to the students in a media-rich and sortable format, and (c) solicit student preferences and accurately record project preferences. As a result, majority of the tasks that were managed by each engineering discipline were eliminated and replaced by an automated process that ensured accuracy and consolidated past multiple data streams. It is envisioned, the current platform will necessitate limited intervention from faculty to yield a fair and satisfactory college-wide assignment output; preferably entirely eliminating the need for discipline-managers. In the past, discipline-managers devoted substantial effort towards the assignment exercise during a relatively busy period of the term. With the planned refinements to the assignment algorithm, the solution promises to become a robust platform for future iterations of the overall project assignment process.

References

1. Kadlowec, J., Bhatia, K., Chandrupatla, T.R., Chen, J.C., Constans, E., Hartman, H., Marchese, A.J., von Lockette, P., Zhang, H., "Design Integrated in the Mechanical Engineering Curriculum: Assessment of the Engineering Clinics" Journal of Mechanical Design 129.7 (2007): 682-691.

2. Von Lockette, P., Riddell, W., Dahm, K., Harvey, R., Courtney, J., Pietrucha, B., Diao, C., Accini, D., "Bottle Rockets and Parametric Design in a Converging-Diverging Design Strategy," *Proceedings of the ASEE Annual Conference and Exposition,* June 2006, Chicago, IL.

3. Constans, E., Courtney, J., Dahm, K., Everett, J., Gabler, C., Harvey, R., Head, L., Hutto, D., Zhang, H., "Setting the Multidisciplinary Scene: Engineering Design and Communication in the 'Hoistinator' Project," *ASEE Annual Conference and Exposition*, June 2005, Portland, OR.

4. Bakrania, S., Bhatia, K., Riddell, W., Weiss, L., and Dahm, K., "Wind Turbines to Teach Parametric Design," *ASEE Annual Conference and Exposition*, June 2009, Austin, TX.

5. Engineering Accreditation Commission, Engineering Criteria 2000, Accreditation Board for Engineering and Technology, Inc., Baltimore, MD, 1998.

6. Google Apps Script, http://www.google.com/script/start/

7. Freiheit, T., and Wood, J., "An Algorithm for Project Assignment in Capstone Design," ASEE Annual Conference and Exposition, June 2007, Honolulu, HI.

8. Chang, M., Downey, A., "A Semi-automatic Approach for Project Assignment in a Capstone Course," *ASEE Annual Conference and Exposition*, June 2008, Pittsburgh, PA.

9. Schmidt, P. L., Hoch, D., Bousaba, N. A., Heybruck, W. F., Sharer, D. L., Cecchi, V., Teng, S. G., Sharer, E., "An Optimization Routine for Assigning Students to Capstone Project Groups," *ASEE Annual Conference and Exposition*, June 2011, Vancouver, BC, Canada.

10. Warnick, G., Todd, R., Dymock, R., Mattson, C., "A Student Bidding Process Applied to Industrially Sponsored Senior Capstone Design Projects," *ASEE Annual Conference and Exposition*, June 2009, Austin, TX.

11. Jordan Pedraza, "Google Apps for Education: 32 ways to start using apps for Faculty/Staff," Google Apps for Education Team, goo.gl/vEhBz