Proposed System: cuPID

by Christine Laurendeau

Problem Statement

In order to better prepare university graduates for future employment, many university courses require students to work on team projects so that they can develop cooperation and teamwork skills. However, a common problem with these team projects is that students within a team are often not compatible with each other, often mismatched in terms of skill set, personalities and work habits. The *Carleton University Project Partner IDentifier (cuPID)* proposes to address this issue by providing a tool that automatically matches together students into teams, based on compatibility. For any given project, the goal of the *cuPID* system is to generate a set of teams, with each team made up of project partners (students) that best match each other in terms of compatibility.

The main features of *cuPID* include:

- the ability to create projects;
- the ability to add students to projects;
- the ability to edit student project partner profiles;
- the ability to manage essential project data, such as team size; and
- the ability to assess project partner compatibility and compute a set of optimal team configurations.

The *cuPID* system supports two categories of users: students and administrators. Administrators create projects, manage project properties such as team size, and launch the *project partner identification (PPID) algorithm* that computes the best team configurations for a specific project. Students can register themselves in an existing project, and they can edit their *project partner profile*. Each student's profile includes two types of information: the student's own *qualifications* (attributes that describe the student), and the qualifications that the student would prefer in his/her project partners on the same team. The PPID algorithm uses the profiles of all students registered in a project in order to compute the best possible set of teams of compatible students.

Features

Student users can view a list of existing projects and add themselves to any number of projects. Students can also edit their own project partner profile. Each student has exactly one profile, to be used by the PPID algorithm when computing teams for any project in which the student is registered. A project partner profile specifies the student's own qualifications, and the qualifications that the student is seeking in his/her project partners.

Administrator users can create and edit projects. They can also modify configuration values required by the PPID algorithm, including team size and possibly other attributes, which will be specific to the algorithm. Most importantly, administrator users can launch the execution of the PPID algorithm for a specific project. This algorithm uses the project partner profile of every student registered in the project, and using the information in these profiles, computes the best matches between all students. The algorithm forms the optimal teams based on the project team size, and prints out the match results. These results must consist of both summary and detailed information. The summary information indicates the names of all students in the project, grouped by team. The detailed information specifies the rules and the data supporting how and why these specific matches were computed.

Technical Specifications

The Linux Ubuntu platform, as provided in the official course virtual machine (VM), will be used as the test bed for evaluating the *cuPID* system. All source code must be written in C++.

GUI

The *cuPID* user interface (UI) must be graphical in nature. In general, the different user features should be easily navigable, either as menu items and/or popup menus. The look-and-feel of the *cuPID* system should be professional and consistent with commercially available UIs, including the use of popup menus where appropriate.

Data Storage

The *cuPID* system will run on a single host, with all its data stored on that same host. The system must be delivered already data-filled with a *minimum* of 25 completed project partner profiles.

Data storage must be organized for ease of retrieval and efficient use of storage space. There should be no duplication of information anywhere. Data may be stored in flat files, or any other mechanism available in the VM provided.

Project Partner Identification (PPID) Algorithm

Every development team implementing the *cuPID* project will define a set of *qualifications* that they believe are important in matching together students who are compatible and likely to work well together. Examples of qualifications could include final grades in some core courses, the level of competence in specific relevant skills, the level of importance of some work habits, etc. These qualifications make up the project partner profile that every student user must fill out. A profile must contain specific values for that student for each qualification (for example: B+ in COMP2404, intermediate skill level in algorithms, strong belief in punctuality). A profile also contains the values for qualifications that the student is looking for in project partners on the same team.

Once a development team has defined a set of qualifications, they must design their own *cuPID* project partner identification (PPID) algorithm. The algorithm must use its own unique set of *rules* for matching together project partner profiles, based on the qualifications contained in those profiles. Each development team's PPID algorithm will be evaluated based on the originality and appropriateness of its matching rules and qualifications.

The algorithm must use a *minimum* of 12 (twelve) separate and unique qualifications, with no more than three (3) of these 12 consisting of course marks. **All the qualifications** defined in a project partner profile must be used by the algorithm's rules to compute the best-matched teams. This includes a student's own qualifications and the ones that he/she requires of his/her project partners.

Team Project

Deliverable #1: Phase #1 Model

The work submitted for this deliverable will consist of:

- a soft copy of the Requirements Analysis Document, in PDF format

The content of this deliverable will be discussed in class. A soft copy of the PDF document must be submitted on cuLearn, on or before Wednesday, October 14 at 2:30 PM sharp. The submitted copy must be typed and legible, and it must look as professional as if it was being submitted to real client, including cover page, table of contents, table of figures, etc. All diagrams and tables must be introduced and explained in the text. Documents that do not conform to these specifications will not be marked.

Deliverable #2: Phase #1 Prototype

The work submitted for this deliverable will consist of:

- the prototype source code, data and configuration files, and operating instructions, packaged as a tar file
- a soft copy of the algorithm design document, in PDF format
- an in-class presentation on the algorithm design
- a soft copy of the PowerPoint or PDF slides for the presentation

The content of this deliverable will be discussed in class. The prototype source code and presentation slides must be submitted on cuLearn, on or before Wednesday, November 4 at 2:30 PM sharp. An official design review will be held during class on November 9, 11 and 16. Each team will be given *five minutes* to present their design.

Deliverable #3: Phase #2 Model

The work submitted for this deliverable will consist of:

- a soft copy of the System Design Document, in PDF format

The content of this deliverable will be discussed in class. A soft copy of the PDF document must be submitted on cuLearn, on or before Monday, November 23 at 2:30 PM sharp. The submitted copy must be typed and legible, and it must look as professional as if it was being submitted to real client, including cover page, table of contents, table of figures, etc. All diagrams and tables must be introduced and explained in the text. Documents that do not conform to these specifications will not be marked.

Deliverable #4: Phase #2 Prototype

The work submitted for this deliverable will consist of:

- the prototype source code, data and configuration files, and operating instructions, packaged as a tar file
- a soft copy of the revised Algorithm Design Document, in PDF format
- a prototype demo to take place by appointment on Demo Day, Tuesday, December 8

The content of this deliverable will be discussed in class. The prototype source code and revised Algorithm Design Document must be submitted on cuLearn, on or before Monday, December 7 at 23:55.