Isaac Reynolds (isaacr@uw, isaacr@cs)
Jake Roberts (jcwr@uw)
Janelle Van Hofwegen (jvh23@cs)

Purpose

We will create an **interactive, dynamic graph visualization of course prerequisites** at the UW. This will aid **long-term degree planning** and **course discovery**.

UW IT and the MyPlan team support this project. If successful, **parts of this project may be integrated into MyPlan.**

Vision

Bob recently made a long-term plan of the courses he wants to take at the UW. His plan helps him ensure that he's on track to graduate on time, but it will have to be revised repeatedly when new information about course offerings becomes available or if Bob is unable to register for a course he planned to take. Verifying that prerequisites are satisfied in each of Bob's plans is a monotonous, time-consuming, and error-prone process, and there are no tools to help him (or the tens of thousands of other students at the UW engaged in the same work).

Course Explorer's intuitive graph visualization presents information about courses as well as how they relate to each other and Bob's academic past and future. Bob can explore the graph, dynamically exploring new branches and trimming others. As he works, the visualization provides feedback about how close Bob is to satisfying all of the prerequisites of courses he plans to take. **This helps Bob efficiently plan what courses he should take, and in what order he should take them, in order to achieve his educational goals.**

Course Explorer may also be used to implement MyPlan features, such as visualizing prerequisites within a major and verifying that a student's plan satisfies course prerequisites.

What Alternative Solutions Could We Implement?

One option is to replace a graph visualization with a simpler list visualization. This would dramatically reduce the cognitive load for users unfamiliar with the concept of a graph (and be easier to implement), but may not be as visually compelling.

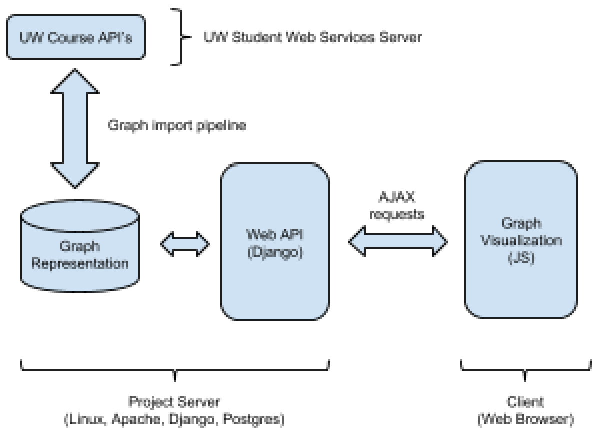
We have also considered using a simpler, less dynamic graph visualization, which would simplify implementation. Instead of adding or removing courses within the graph visualization, users would see a static visualization of possible course paths.

What About Competitors?

Currently students make long-term plans in MyPlan, but there is no software that visualizes course prerequisites to help students make long-term plans or that validates prerequisites in plans that students have made. **We are collaborating with MyPlan to minimize competition and maximize impact.**

Software Architecture

The architecture will be as modular as possible to enable extensibility, maintainability, and integration with the existing MyPlan software. We have completed proof-of-concept prototypes of several components, however significant design, user research, and development work remains.



Graph Import Pipeline

This module translates data from the UW's Student Web Services API into data in our application's storage. It includes submodules that retrieve various types of course data, each of which may be replaced as better sources of data become available (as a result of the UW's three-year plan to move all registration, advising, and planning onto MyPlan).

Data Representation, Storage, and Algorithms

We will use Django and Postgres to organize our graph data and respond to AJAX requests from clients. Graph algorithms over this data may be written in Python or Javascript, depending on whether they execute server-side or client-side.

Our data representation and algorithms must be capable of manipulating complex prerequisites (such as "(CSE 332 OR CSE 326) AND CSE 331"). This requires heavily modifying well-known graph algorithms and creating some entirely new algorithms. Our graph representation must be carefully tailored to efficiently serve queries needed by these algorithms.

Data Visualization

This layer requests structured course data from our web API and produces an interactive visualization from it. The interface between the data visualization layer and the controller layer is well-defined to allow the data representation layer to be easily replaced in the future.

Challenges

The main challenge is choosing a scope that is both useful and complete and can be implemented in ten weeks. We will mitigate this risk by clearly defining the scope at the beginning of the project. We will also choose a minimum viable product and achieve that goal as early as possible. Additional features will be prioritized and completed in the remaining time.