Princeton Plan-O-Matic

Eirik Bakke
Arthi Ramachandran
Alison Reynolds
David Weiss
Motivation

- Long-term academic planning is hard
  - Departments and programs have complicated requirements
  - No centralized mechanism for experimentation
- Current systems inadequate
  - SCORE / Degree Progress Report
  - Student Course Guide
Princeton Plan-O-Matic

- A tool for students and advisors to explore possible 4-year schedules

- Major features:
  - Easy access to degree, major, program, and other requirements
  - Immediate feedback
  - Simple, visually driven interface
  - Experiment with multiple schedules
Demo

- Princeton Plan-O-Matic
Architecture Overview

Data Import

MySQL Server

Database Interface

Report Generator

GUI
Back-end

- MySQL Server
- Data Import
- MySQL Server Database Interface
- Report Generator
- Database Interface
- GUI
- Data Import

Connections:
- MySQL Server → Data Import
- Data Import → MySQL Server Database Interface
- MySQL Server Database Interface → Report Generator
- Report Generator → Database Interface
- Database Interface → GUI
Data Import

- Course information downloaded from registrar
- Manually specified program requirements
- Automated database build and update
(Arrows indicate foreign key relationships)
Example: Cross-listings

- cos306
- ele206
- genCourse

“Introduction to Logic Design”

“Introduction to Programming”

- cos126
Example: Course Codes

- genST
- frs137
- genHA

“Rocket Science”

“Changing Life Through Art”
Example: Multiple Offerings

- cos126
- "Introduction to Programming"
  - Fall 2004
  - Spring 2005
  - Fall 2005
Database Interface

Data Import

MySQL Server

Report Generator

Database Interface

GUI
Database Interface

- Object-oriented (PHP)
- Abstract all interaction with database
- Process user login
- Search implementation
Interface Classes

- User, Group, Pursuit, Schedule, Semester, PursuitSearch
- Store primary key of corresponding database row
GUI Design & Implementation

- Design Goals
  - Intuitive, easy to use
  - Display lots of information in clear manner
  - Visually appealing across browsers

- Implementation Goals
  - Efficiency
  - Maintainable, Modular
Overview of Data Flow

User

HTML/CSS

Javascript

PHP

Database Interface
Implementation Details

- schedulePane
  - Semester
    - Pursuit
  - Droppable
  - Collapsible
  - Draggable

- searchPane
  - Pursuit
Prerequisite Expressions

- “PHY 103 and PHY 104 are prerequisites, MAT 203 is a corequisite.”

  \( \text{phy103} \land \text{phy104} \land \text{mat203} \leq \text{same} \)

- “Three HA courses and five ST or STX courses.”

  \( 3 @ \text{genHA} \land 5 @ (\text{genST}, \text{genSTX}) \)

- “Three courses distributed among four areas of concentration (no more than one course from each area may count towards the total of three).”

  \( 3 @ (1 \% \text{cheBio}, 1 \% \text{cheEnt}, 1 \% \text{cheEnv}, 1 \% \text{cheMat}) \)
Prex (PRerequisite Expression) Parsing

6 @ (genHA, genSA, genEM) ; cos333 & rel252 & ele302

- “Six courses from three distribution areas, as well as three specific other courses.”
- Suppose you have taken COS 333 as well as one SA course and three HA courses
- Operators are ; & | @ % $
6 @ (genHA, genSA, genEM) & ((rel252 & ele302) & cos333)
After credits resolved

6 @ (genHA=3, genSA=1, genEM) & ((rel252 & ele302) & 1)

Note: course references disappear, groups stay
Bubble up constant terms

2 @ (genHA, genSA, genEM) & ((rel252 & ele302) & 1)
Eliminate fulfilled branches

2 @ (genHA, genSA, genEM) & ((rel252 & ele302) & 0)

Diagram:

```
2 @ 0 & 0
   /   \
  /     \ /     \ & 0
/       /       /     \
genHA 0  genSA 0   genEM 0 rel252 0 ele302 0
```
Collapse associative chains

2 @ (genHA, genSA, genEM) & (rel252 & ele302)
Collapse associative chains

2 @ (genHA, genSA, genEM) & rel252 & ele302
2 @ (genHA, genSA, genEM) & rel252 & ele302

- Freshman Spring '05
- AAS 209/ENG 209 (S04-05)
  - Introduction to African-American Literature: Harlem Renaissance to Present
  - Each of the following:
    - ele302
    - At least 2 of genHA, genSA, genEM
    - rel252
Possible Features

- Warn about time conflicts
- Generic courses
- User override
- Helpful course suggestions
- Search for nearly satisfied programs
- Link to Student Course Guide reviews
Fun Facts

- Languages used: awk, Java, Javascript, PHP, C, yacc, HTML, CSS, bash, SQL
- Current lines of code:
  - >1000 interfaces, >1500 GUI, >1000 import, >1300 report generator
- Database:
  - 2508 pursuits, 3542 groups, 8672 memberships
- SVN commits:
  - 310 revisions (34/week avg.)
- …Questions?