Development of a Course Degree Program Browser

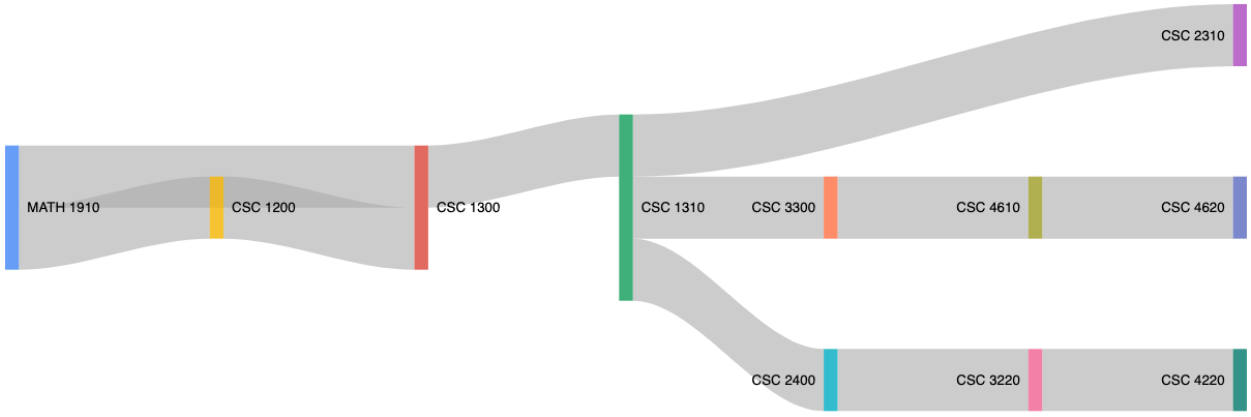
CSC 2310 – Fall 2021

Abstract

Sankey Diagrams facilitate understanding of complex flow systems. In the context of academic programs, a Sankey Diagram can help students, faculty, staff, administrators, and assessment professionals understand the complexity of course pre-requisite chains and dependencies. This project will focus on the development of a web-based system for browsing information about degree programs using a Sankey Diagram.

Degree programs at universities are designed as a scaffolded collection of courses that have dependencies that can bridge across several disciplines. These dependencies create a level of complexity that can often lead to unintended or unforeseen consequences when it comes to creating a personalized (or even generalized) program of study.

In this project, you will design, develop, and test a system for visualizing the dependencies of an arbitrary degree program. Our chosen visualization format is a *Sankey Diagram*, a representation that has typically been used to model flow dynamics. In particular, a Sankey Diagram shows sequential dependencies between different nodes in a graph. For example, the following graph models part of the CS curriculum:



Ultimately, this software system will use a web-based interface that allows current students, prospective students, faculty, administrators, and program accreditors to view the dependency tree of given courses in a program.

In this iteration, you will be doing the following:

- Identify all of the potential user types for this system
- Brainstorm the potential features of the system – for instance, some users might want to add information, others may want to view specific course information (by mousing over a node?)

- Specify user stories for each of the user types based on your problem analysis using the

As you begin the brainstorming process, recall that as a "developer" you have biases, and that you need to try adopt the mindset of different kinds of users. You may want to ask yourself questions such as:

- What other information should we display? What does each type of user need to see and do?
- Should a user be able to add or modify data? Which type of users can do this?
- Can we use a graph to identify inconsistencies in the degree program caused by the pre-requisite structure? Can we find the longest pre-requisite chains? Shortest pre-requisite chains?

As you go through this activity, you should attempt, as much as possible, to avoid thinking about how any of this is going to be implemented. The most important aspect of this activity is identifying all of the *possible* features for this system (noting that identification of all of the possible features does not mean you will have to implement all of the features).

Project Phases

The project will be conducted in two major phases: *concept-initiate* and *iteration-construction*. In the *concept-initiate* phase you will identify project requirements, create models, and prep your development environment to support your implementation activities. In the *iteration-construction* phase of the project you will iteratively develop software for the project by adding support according to the requirements developed during the *concept-initiate* phase. The schedule of the phases of the project and the individual iterations contained therein are defined below:

Date (Midnight)	Phase	Description
September 3	Concept Initiate 0	User stories
September 17	Concept Initiate 1	Use case diagrams
October ~~1~~ 8	Concept Initiate 2	Initial Design
October 22	Iteration 0	Project environment setup and initial execution demonstration
November 5	Iteration 1	Basic parsing and visualization
November 19	Iteration 2	Advanced parsing and graph creation
December 3	Iteration 3	Graph analysis and retrospective

Concept Initiate 0

User Stories

In the initial phase of the project, you will create user stories based on the description provided while also generating ideas of other potential features for such a system. The user stories that you create should be specified in the following form:

As a <user type>, I want <desired feature>, so that <outcome>

You must also include, with each user story, a description of how you would expect to be able to observe this behavior in the system. Your source of information for the user stories you will identify are given in the description provided above as well as any discussion sessions we conduct in class. You may also reach out to faculty, academic advisors, or other students to generate ideas on potential stories. **However, the work you turn in must be your own. You may not copy other students' user stories, descriptions, or other similar work products.**

A template for your user stories is provided below.

User Story Template

Use the following user story template for *concept-initiate 0*. Replicate this as many times as is necessary.

User Story Name: <Short phrase for the user story>	
As a:	<user type>
I want:	<feature>
So that:	<outcome or reason>
Description	<description of how the user story / feature would be observed and tested in the system>

Submission

Submit your user stories as a PDF document only. *Microsoft Word, LibreOffice, or any other native word processing format will not be accepted.*

Rubric

Completeness: You will be graded on the completeness of your turn-in. In particular, you must analyze the description and attempt to identify as many features as you can that are relevant for the project. Note, there is no upper bound on the number of user stories you can define, but there is a lower bound that *will not* be specified.

Correctness: You will be graded based on your adherence to the format of the user story template and on the accuracy of the user stories with respect to relevance to the described project, including your description of how you might expect to observe the user story / feature in the system.

Points: The assignment is worth 20 project points.

Concept Initiate 1

Use Case Diagrams

In our initial step on this project, we did some work to try to identify user stories for the Course Prerequisite system. The baseline set of user stories has been uploaded to the course iLearn site and should be used as a starting point for completing the next activity in the project.

For Concept Initiate 1, you will be creating use case diagrams for the Course Prerequisite system. The baseline user stories fall into a few categories:

Basic – User stories that can be generated with little more than basic information available from the default data source

Edit, Modify, Delete – User stories that require editing permissions or other identity management

Global – User stories that assemble global information using courses available university wide

Personalization – User stories that use individual identity to personalize the experience of a single user

What-if – User stories that perform analysis of a degree program based on various graph-based qualities of the data

As you did in Laboratory 2, you will use StarUML to create your use case diagram(s) for the Course Prerequisite system. The user stories that you will use for creating the use cases in the diagram are included as an attachment to the assignment in iLearn. Some things to note about these user stories:

- There are 25 user stories specified in the reference set across 7 different user types.
- Some of the user stories are literally the same user story and differ only by the user type and the rationale ("so that…"). Those stories should be specified as a single use case with different user types. As such, it is expected that there will be fewer than 25 use cases in the diagram.
- The use case diagram should be partitioned according to the feature categories specified in the reference set.

Submission

For your turn-in, you should use either *Snip and Sketch* for Windows, ⌘-Shift-5 on Mac, or Shift-PrtSc in Linux to do a screen capture of your model from StarUML. Copy and paste the image to word or some other word processing suite and generate a PDF.

Rubric

Completeness: You will be graded on the completeness of your turn-in. In particular, your submission must at the very least include the user stories provided in the baseline set. However, it is also expected that you will include other stories in your diagram.

Correctness: You will be graded on the correctness of your use case diagrams. In particular, your submission must correctly represent use cases, actors (both interactive and external actors), and partitioning of the stories into appropriate subsystems.

Points: The assignment is worth 20 project points.

USER STORY INDEX: 1
Priority: ***COULD***
Category: ***BASIC***

AS A	*Department chair*
I WANT TO	*share prerequisite graphs with other departments*
SO THAT	*I can communicate with other departments regarding the interdependencies of our courses*

USER STORY INDEX: 2
Priority: ***WON'T***
Category: ***BASIC***

AS A	*Department chair*
I WANT TO	*view information about when courses have been offered in a graph*
SO THAT	*I can make plans about future scheduling of courses in the department*

USER STORY INDEX: 3
Priority: ***COULD***
Category: ***BASIC***

AS A	*Department chair*
I WANT TO	*view course enrollment information*
SO THAT	*I can make decisions about how often to offer a course within our pre-requisite structures*

USER STORY INDEX: 4
Priority: ***MUST***
Category: ***BASIC***

AS A	*Department chair*
I WANT TO	*filter courses by subject*
SO THAT	*I can see courses within their proper context*

USER STORY INDEX: 5
Priority: ***MUST***
Category: ***BASIC***

AS A	*Department chair*
I WANT TO	*select a subject code (e.g., "CSC")*
SO THAT	*I can provide context to the courses that I want to view*

USER STORY INDEX: 6
Priority: ***COULD***
Category: ***EDIT, MODIFY, DELETE***

AS A	*Department chair*
I WANT TO	*add new courses to the graph*
SO THAT	*the department can communicate new changes being made*

USER STORY INDEX: 7
Priority: ***WON'T***

Category: ***EDIT, MODIFY, DELETE***

AS A	*Department chair*
I WANT TO	*remove old courses from the graph*
SO THAT	*the department can communicate new changes being made*

USER STORY INDEX: 8
Priority: ***COULD***
Category: ***EDIT, MODIFY, DELETE***

AS A	*Department chair*
I WANT TO	*change course dependencies*
SO THAT	*the department can communicate new changes being made*

USER STORY INDEX: 9
Priority: ***COULD***
Category: ***WHAT-IF***

AS A	*Department chair*
I WANT TO	*run graph algorithms on a pre-requisite graph*
SO THAT	*we can determine whether there are any problems in our program related to dependencies between courses*

USER STORY INDEX: 10
Priority: ***SHOULD***
Category: ***BASIC***

AS A	*Faculty member in department*
I WANT TO	*view course information*
SO THAT	*I can verify that we are communicating correct information to students and other faculty about our courses*

USER STORY INDEX: 11
Priority: ***SHOULD***
Category: ***BASIC***

AS A	*Faculty member in department*
I WANT TO	*view courses outside of our program in different colors*
SO THAT	*I can review the relationships between our courses and courses in other departments*

USER STORY INDEX: 12
Priority: ***MUST***
Category: ***BASIC***

AS A	*Faculty member in department*
I WANT TO	*view a prerequisite graph*
SO THAT	*I can help with future course planning*

USER STORY INDEX: 13
Priority: ***MUST***
Category: ***BASIC***

AS A	*Program Evaluator*
I WANT TO	*view a prerequisite graph*
SO THAT	*I can help identify potential issues in the structure of a degree program*

USER STORY INDEX: 14
Priority: ***WON'T***
Category: ***GLOBAL***

AS A	*Program Evaluator*
I WANT TO	*view several programs in a single graph*
SO THAT	*I can identify potential overlaps between different degree programs*

USER STORY INDEX: 15
Priority: ***MUST***
Category: ***BASIC***

AS A	*Prospective Student*
I WANT TO	*browse prerequisite graphs*
SO THAT	*I can see how long it will take before I can take a particular course*

USER STORY INDEX: 16
Priority: ***MUST***
Category: ***BASIC***

AS A	*Student in general*
I WANT TO	*select a subject code (e.g., "CSC")*
SO THAT	*I can provide context to the courses that I want to view*

USER STORY INDEX: 17
Priority: ***MUST***
Category: ***BASIC***

AS A	*Student in general*
I WANT TO	*view a prerequisite graph*
SO THAT	*I can see dependencies between courses in a program*

USER STORY INDEX: 18
Priority: ***MUST***
Category: ***BASIC***

AS A	*Student in general*
I WANT TO	*view course information*
SO THAT	*I can get details about specific courses*

USER STORY INDEX: 19
Priority: ***MUST***
Category: ***BASIC***

AS A	*Student in general*
I WANT TO	*highlight course paths*
SO THAT	*I can see how long it will take before I can take a particular course*

USER STORY INDEX: 20
Priority: ***SHOULD***
Category: ***PERSONALIZATION***

AS A	*Student Major*
I WANT TO	*view a graph of just my concentration*
SO THAT	*I can see which courses are specific to my program of study*

USER STORY INDEX: 21
Priority: ***MUST***
Category: ***PERSONALIZATION***

AS A	*Student Major*
I WANT TO	*view a graph of my specific program of study (including minor courses)*
SO THAT	*I can see how long it might take me to complete my degree*

USER STORY INDEX: 22
Priority: ***COULD***
Category: ***PERSONALIZATION***

AS A	*Student in general*
I WANT TO	*connect the graph to my degreeworks profile*
SO THAT	*I have a record of which courses I have completed*

USER STORY INDEX: 23
Priority: ***SHOULD***
Category: ***PERSONALIZATION***

AS A	*Student in general*
I WANT TO	*have the graph highlight courses I have taken*
SO THAT	*I can see what courses I am eligible to take next*

USER STORY INDEX: 24
Priority: ***WON'T***
Category: ***WHAT-IF***

AS A	*Transfer Student*
I WANT TO	*map courses from my previous school*
SO THAT	*I can view which courses I am eligble to take based on my transferred credits*

USER STORY INDEX: 25
Priority: ***COULD***
Category: ***WHAT-IF***

AS A	*Student in general*
I WANT TO	*view a what-if graph*
SO THAT	*I can see what possible courses can be taken given completion of an individual course*

Concept Initiate 2

Class Diagrams

The course pre-requisite system we are building uses a *3-tier architecture* common to many *web-based* applications. In particular, the system has the following structure:

where the *classes* named *client*, *server*, and *webservices* represent the *presentation*, *logic*, and *data* layers. These layers are responsible for the following:

- Client / Presentation: provides an interface to the end-user so they can access the functions provided by the application. In web-based applications, this interface is usually accessed via a web browser.
- Server / Logic: controls the logic needed to control an application, including aggregating data in forms usable by the application.
- WebServices / Data: stores the data used by the application; the data can come in multiple forms and be provided from a number of difference sources.

In an ideal world, *service-oriented architectures* of this sort would have *web services* readily available to developers for applications of the sort we are building. Such an ideal world does not exist for Tennessee Technological University ¯_(ツ)_/¯. As such, as part of our implementation of the pre-requisite browser we will be creating web services that provide a uniform way of accessing the data needed to drive the application.

In order to manage the data needed for our application we need to analyze the data available to us and design appropriate classes. For our purposes, we have three sources of data at our disposal:

- Web pages – we will use *web scraping* technology to pull data from publicly available websites, including course titles, descriptions, and other *unstructured* data.
- Banner web services – we will use a REST web service to pull data from a data source that provides information about courses offered in a given semester.
- Comma Separated Value (CSV) files – we will use stored files that contain information about pre-requisites for various courses

Course Catalog Info

The university maintains a website/application for publishing course catalog information at the following url: `https://ttuss1.tntech.edu/PROD/bwckctlg.p_disp_course_detail`. This will later be used in our application as a source for using web scraping to pull data needed for our application. The above url expects the use of all of the following parameters:

- **cat_term_in**: specified catalog year in *yyyyss* format, where *yyyy* is the four-digit year, and *ss* is the semester code
    - Fall = 80
    - Spring = 10
    - Summer = 50

  For example: 202180 specifies Fall 2021.

- **subj_code_in**: Subject code, such as “CSC”
- **crse_numb_in**: Course number, such as 1200

The following url is a valid construction:

https://ttuss1.tntech.edu/PROD/bwckctlg.p_disp_course_detail?cat_term_in=202180&subj_code_in=CSC&crse_numb_in=1300

This url will return the following information (in addition to page header and footer information):

CSC 1300 - Intro/Prob Solving-Comp Prog
Prerequisite: CSC 1200 or MATH 1845 or MATH 1910. MATH 1845 or MATH 1910 may be taken concurrently. Digital computers; problem solving and algorithm development; programming is introduced using a procedural approach, but classes and object-orientation are introduced; design and testing are emphasized. Students complete a series of weekly laboratory exercises for developing proficiency in problem solving and computer programming.
0.000 OR 4.000 Credit hours
0.000 OR 3.000 Lecture hours
0.000 OR 2.000 Lab hours

Levels: Undergraduate
Schedule Types: Laboratory, Lecture
All Sections for this Course

Computer Science Department

Prerequisites:
Prereq for CSC 1300

General Requirements:
(Course or Test: CSC 1200
Minimum Grade of D
May be taken concurrently.)
or
(Course or Test: MATH 1845
Minimum Grade of D
May be taken concurrently.)
or
(Course or Test: MATH 1910
Minimum Grade of D
May be taken concurrently.)

Course Offering Data

In addition to course catalog data, we have available to us a web service that can be used to retrieve information about specific offerings of a course. Specifically, using an API endpoint (which will be revealed in a later iteration), the following data can be retrieved:

```
{
    "CRN":"81954",
    "DEPARTMENT":"CSC",
    "COURSE":"2310",
    "SECTION":"002",
    "CREDITS":4.0,
    "ISTIMEDETERMINED":"true",
    "STARTTIME":"1600",
    "STARTAM_PM":"pm",
    "ENDTIME":"1650",
    "ENDAM_PM":"pm",
    "CLASSDAYS":"MWF",
    "ISLOCDETERMINED":"true",
    "BUILDING":"BRUN",
    "ROOM":"228",
    "ISONLINE":"false",
    "PROF":"Gannod, Gerald C",
    "STUDENTCOUNT":51,
    "MAXIMUMSTUDENTS":55,
    "ISOPEN":"true",
    "TITLE":"Object-Oriented Prgrming/Dsgn"
  }
```


This example shows data specific to Section 002 of the CSC 2310 course.

Design Task

Using a UML class diagram,

- Create a class for storing course catalog information,
- Class that aggregates all course data into collections according to subject. Specifically, a subject area (such as Computer Science) is a collection of several courses.
- Class for storing course instance information,
- Class that aggregates all course instance data into collections according to subject

Some things to make note of:

- Your class model should include attributes relevant to a course (as you determine relevant) with the various types specified using the standard UML notation.
- You do not need to include operations.
- It is expected that your model will include at least four classes, but could include more
- There are relationships that exist between catalog information and course instance information that can be captured

Preparation

There are four videos available on the course iLearn site that explain, in detail, the Class Diagram notation.

Submission

For your turn-in, you should use either *Snip and Sketch* for Windows, ⌘-Shift-5 on Mac, or Shift-PrtSc in Linux to do a screen capture of your models from StarUML. Copy and paste the image to word or some other word processing suite and generate a PDF.

Rubric

Completeness: You will be graded on the completeness of your submission, mostly notably the inclusion of the classes as specified above and the attributes of the data described above.

Correctness: You will be graded based on the correctness of the class specifications you have created including correct capture of relationships between the classes and the specification of attributes based on the descriptions provided above.

Points: The assignment is worth 30 project points