CMD Degree Guide

Ethan Cha, Haley Figone, Yaya Yao, Peyton Elebash

Software Requirements Specification

Table of Contents

1. Revision History

Date	Name	Description
10/07/2023	Peyton Elebash	Updated the title and names. First draft of

Date	Name	Description
		ConOps 2.1 Current system or Situation and first draft of 2.2. Justification for a New System.
10/08/2023	Peyton Elebash	First draft of 2.3 Operational Features of the Proposed System. First draft of 2.4 User Classes. First Draft of 2.6 Operational Scenarios (Use Cases). First draft of 3.1 External Interfaces. First draft of 3.3 Usability Requirements. First draft of 3.4 Performance Requirements. First draft of 3.5 Software System Attributes.
10/08/2023	Haley Figone	Revisions and updates to sections 2.3, 3.1, 3.4, 3.5. Various formatting adjustments.
11/25/2023	Peyton Elebash	Revisions and updates to sections 2.1, 2.2, 2.3, 2.4, 2.5, 2.6, 3.1, 3.3, 3.4, and 3.5.
11/26/2023	Peyton Elebash	Updated sections 3.1 and 3.2.

2. The Concept of Operations (ConOps)

The concept of operations (ConOps) "describes system characteristics for a proposed system from the users' viewpoint." (IEEE Std 1362-1998) The ConOps document communicates overall system characteristics to all stakeholders. It should be readable by any stakeholder familiar with the application domain. The ConOps should clarify the software's context and the capabilities the system will provide the user (Faulk, 2017).

2.1. Current System or Situation

College can be extremely stressful if students do not manage time well. A huge part of this time management is planning their courses in a way that does not overwhelm their workload and ensures that they will graduate in at least 4-years. Usually core university requirements are outlined clearly since all students must meet the same ones. However, major requirements can become tricky for students to keep track of. The current resources for creating a graduation plan involve finding an advisor or consulting the degree guide on duckweb. However, not all students know how to access, use, or feel comfortable utilizing these resources. Not to mention, with the current degree guide format it's more realistic for students to skip this step and go right to an advisor. Everyone at the University of Oregon deserves to feel confident in navigating and generating their projected degree guide, but with changing majors, dropping courses, and life in general, it can be challenging to feel that way. Students need a platform that makes this process more efficient, more accessible, and more accurate.

2.2. Justification for a New System

For students at the University of Oregon, knowing if they will graduate on time is a constant stressor. The university offers an online degree guide that in theory should accurately project if they are on schedule to graduate or not. However, navigating this degree guide is often very frustrating due to the layout and extra/unnecessary information. The current degree guide is laid out as one long PDF, with sections that are not clearly divided. Oftentimes, students cannot guarantee if they are on course to graduate in 4-years unless they seek out an advisor, due to how unorganized the current degree guide is. The goal of the CMD Degree Guide is to minimize all of these complications. We want the students to be able to confidently confirm that they are on-time to graduate from the comfort of their own home. The CMD Degree guide will be a web application similar to the current one, however, it will be laid out in a concise and well-formatted way (fitting on one page), cutting out any unnecessary information. It will also prompt the user for the number of terms left until their projected graduation date, stating if it is an achievable goal or not. CMD Degree Guide will save the student's degree guide once it is generated, so anytime they log on they are able to access it.

2.3. Operational Features of the Proposed System

The new degree guide system, CMD Degree Guide, will be a web application for students seeking out a 4-year graduation plan as well as confirmation they are on track to graduate on time. The student will select the courses they have taken, their desired graduation date, and their major (as of now CS, MATH (Pure Track), or Data Science majors only). Using this information, the CMD Degree Guide will generate a tentative plan for students to follow that includes: which courses to take during each Fall, Winter, and Spring term of their remaining time at the University of Oregon. The degree guide provides specific courses and/or the type of courses the students should take each term based on their selected major's requirements. The CMD will store this generated degree guide in the database with the students UO ID so that they are able to access this information on their next log-in.The Degree Guide will also show if the students desired graduation date is plausible given the courses they have taken. By using CMD Degree Guide, students finally have a platform that provides them with a readable and easy way to confirm they are on the right track to graduate in 4 years.

2.4. User Classes

The main user class will be students at the University of Oregon looking to graduate with a BS in either Computer Science, Mathematics (Pure track) , or Data Science. This degree guide will not be beneficial for university students outside of the University of Oregon or in majors other than the three listed. The secondary user class will be the programmers (Ethan, Haley, Yaya, and Peyton) whose main interaction with the software will be to add additional course/major functionality and address any unforeseen/new bugs.

2.5. Modes of Operation

The student user class will have two modes:

1. View
   - Students can view their generated degree guide in a table format once it is created, anytime they log-in to the website. They are unable to enter in information or change the table in any way.
2. Edit
   - Students will only be in this mode of operation once, after they have created an account. They are prompted for necessary information to generate the degree plan-terms left, major, and courses they have taken. They can edit their responses to each of these topics as they like as long as no information conflicts.

The programmer user class will have one mode, debugging and updates, which allows the programmer to adjust and add to the backend and frontend aspects of the source code, as well as the database.

2.6. Operational Scenarios (aka “Use Cases”)

Use Case 1: Create Account

Brief description: This use case describes how a student would log on to CMD Degree Guide for the first time and generate a degree guide.

Actors: A student.

Preconditions:

1. The student attends, or is planning to attend, the University of Oregon.
2. The student has access to the internet via mobile device or computer to go to the web application.
3. The student is planning to, or in the process of, acquiring a BS in either CS, Math, or Data Science.

4. The student knows which courses they have taken so far.
5. The student knows how many terms they have left until their desired graduation.
6. The student has not used CMD Degree Guide before and needs to create an account and generate a new degree guide.

Steps to Complete the Task:
1. The student opens up the CMD degree guide website.
2. The student clicks “Create Account” on the CMD Degree Guide landing page.
3. The student enters a username and password of their choosing.
4. The student follows the prompts and enters how many terms until their desired graduation date and their major.
5. The student follows the prompts and enters the courses they have taken so far.
6. The student clicks “Generate Degree Guide”.

Postconditions:
1. The student has generated a degree guide that allows them to see what courses they need to take in order to graduate by their projected date.
2. The student has confirmation on if they are on schedule to graduate.

Use Case 2: Log-in

Brief description: This use case describes how a student would log back into CMD Degree Guide in order to access their already generated degree guide.

Actors: A student.

Preconditions:
1. The student has used the CMD Degree Guide web application to generate a degree guide before.

Steps to Complete the Task:
1. The student opens up the CMD degree guide website.
2. The student clicks “Log In” on the CMD Degree Guide landing page.
3. The student enters their username and password.
4. The student’s degree guide is displayed on screen.

Postconditions:
1. The student has access to the degree guide they generated during their last use of CMD Degree Guide.

3. Specific Requirements

This part of the document is where you specify the actual requirements. A requirement describes a behavior or property that a computer program must have, independent of how that behavior or property is achieved. Requirements must be complete, unambiguous, consistent, and objectively verifiable (see van Vliet, 2008, pp. 241-242, for a discussion of these terms). Requirements describe what the system will do but do not commit to specific *design* details of how the system will do it.

Organize the requirements in a hierarchy. A good organization (a) makes requirements easier to read and understand because related requirements will be near each other in the document, (b) makes requirements easier to modify and update, and (c) makes it easier to find a specific requirement. There are several ways to organize requirements to help achieve these goals.

The following section headings provide one way to organize the requirements (feel free to adapt it to your needs). For example, sections 2.1, 2.2, 2.3, and 2.4 (see headings below) describe "behavioral requirements" (Faulk, 2013). If a system supports two major user activities, it might be best to describe the behavioral requirements for each activity separately. For example, in a digital deejaying system, the two major activities could be (a) loading songs into the system and (b) using the system to play songs. It might be best to fully specify everything in sections 2.1 through 2.4, first for the song-loading activity and then for the song-playing activity.

This document must distinguish between "functional" and "non-functional" requirements. The former describes services provided by the system and the latter constraints on the system and its development.

Requirements should be prioritized, with each classified as (a) Must have, (b) Should have, (c) Could have, and (d) Won't have (MoSCoW – Sethi, p. 110). When reading requirements, it should be straightforward to see how each requirement is classified, such as by grouping them by priority.

Throughout the document, lists and sublists of requirements should be indented and numbered to make it easy to read and reference the specification details. Such as:

1. *General Requirement*
    1.1. *Specific Requirement*
        1.1.1 *Requirement Detail*

Note how this permits reference to "SRS Item 1.1.1".

3.1. External Interfaces (Inputs and Outputs)

Log-in

1. Purpose: To allow the student to log-in with their username and password, and signals that the student has used CMD before and wants to display the already generated degree guide.
2. Source of Input: Input is the student clicking "log in". Then, the input is the student's username and password.
3. Destination of Output: The output is a display of the student's degree guide.
4. Valid ranges of inputs and outputs: Username and password must match information stored in the database of login information. A successful login should redirect the student

to the interface for viewing the degree guide. An unsuccessful login should deliver an error message and prompt the user to return to the landing page to create an account or try to enter in a different username and password combination.
5. Units of measure: N/A
6. Data formats: String

Create Account

1. Purpose: To allow the student to create an account with their username and password. Signals that the student has not used CMD before and wants to be given prompts in order to generate a personalized degree guide.
2. Source of Input: Input is the student clicking Create Account. Then, the input is the student's desired username and password.
3. Destination of Output: The output is multiple pages prompting the student for necessary information to generate a degree guide.
4. Valid ranges of inputs and outputs: Username should not match any existing usernames in the login database. Attempting to create an account with a username that already exists should trigger an appropriate error message. A successful account creation will add the user's information to the database and allow them to create a degree guide.
5. Units of measure: N/A
6. Data formats: String

List of Courses

1. Purpose: To allow the student to select courses they have already completed in order to fulfill their majors requirements.
2. Source of Input: Input is the student selecting the number of terms they have until graduation and selecting their major.
3. Destination of Output: The output is a checklist of all courses required for the major.
4. Valid ranges of inputs and outputs: Input should be selectable from drop-down menus. Selectable majors include Computer Science, Data Science, and Mathematics (Pure track). Courses will be selectable from a preprogrammed database.
5. Units of measure: N/A
6. Data formats: String

Generating a Degree Guide

1. Purpose: Asks new user/student for crucial information in order to generate a degree guide.
2. Source of Input: Input is the selected courses off of the checklist of courses required for a major.
3. Destination of Output: A degree guide based on the student's information is generated.
4. Valid ranges of inputs and outputs: The courses should be selected in a manner that doesn't violate prerequisites. If this happens, the degree guide will not account for all

classes a student may have taken. We left it in their hands to make sure they properly entered the courses that they have already taken. They are able to go back and make changes if they missenter.Output should be formatted as a table — with each row being a year and three columns, denoting fall, winter, and spring terms. Each term will display a maximum of 16 credits worth of classes with some being specific major requirements and others being an optional course 'slot'.

5. Units of measure: N/A
6. Data formats: N/A

3.2. Functions

Define the actions that must take place in the software to accept and process inputs and generate outputs (ISO/IEC/IEEE 29148:2011). These definitions must include:

1. Validity checks on the inputs: The program will ensure that if a user is creating an account for the first time that a user cannot use the same username and password combination as an existing account. The program will ensure the user can only select from one of the three valid majors for this program by restricting them to select from computer science, data science, and math (pure track). The program will ensure if the user is logging in that they enter a valid username and password combination.
2. Sequence of operations in processing inputs: After the user has created an account, they will have access to the page prompting them for major information input. After the user has input the prompted information they will be shown a custom degree guide based on their input. If the user has done this before, they will log in instead of creating an account and after they enter their username and password input, their degree guide will be displayed.
3. Responses to abnormal situations, including error handling and recovery: If the user enters an existing account while creating an account, an error will display asking them to choose a different username password combination. If a user enters a username password combination that doesn't exist when they are logging in, an error will display prompting the user to try to enter a valid username and password.

3.3. Usability Requirements

The CMD Degree Guide will allow new users to create an account and generate a personalized degree guide. This degree guide will be based on the following criteria:

1. User-entered information
    a. Terms left until desired graduation date.
    b. Major.
    c. Courses taken so far.
2. Major Requirements
3. Course Restrictions (Prerequisites)

The degree guide is largely based on the user correctly entering their information. Accuracy cannot be expected if the user misreports course information. The CMD Degree guide will also allow students who have used the degree guide before to log-in and display their already generated degree guide.

3.4. Performance Requirements

Students who are further along in their major will need to provide more information to begin with when interacting with the degree guide. The degree guide in turn will have fewer computations to perform. Permutations for degree guides are determined by topological sort and the speed of generation is dependent on the number of courses remaining and the number of prerequisite requirements across all those courses.

There may be multiple possible degree guides that can be generated given a major and a set of completed courses. The CMD degree guide will present the user with the first permutation it finds and allow the user to view this generated schedule that also accounts for genEd/optional courses by filling in terms after adding major requirements.

3.5. Software System Attributes

The CMD degree guide should, above all, be **easy to read**. This will include minimizing abbreviations and jargon in order to make the meaning of the text accessible to a broad audience; formatting text in a way that is optimized for viewing, including a display that is responsive to the user's reading environment; and providing a visual layout for organizing one's schedule.

Correctness and **Reliability** are also extremely important attributes for the CMD Degree Guide because one of the main reasons for needing this degree guide is to combat the lack of these in the current system. Students should feel confident and secure that the generated degree guide is consistently accurate and accessible.The software will be continuously tested on a variety of cases in order to ensure it is consistent (no crashes/bugs) and accurate (valid degree guide).

4. References

This section lists the sources cited in the creation of this template document. An SRS should reference all of the sources that it draws from. This section may not be necessary if sufficient citations are provided “inline” (at the point of reference) in the document.

IEEE Std 1362-1998 (R2007). (2007). IEEE Guide for Information Technology–System Definition–Concept of Operations (ConOps) Document. https://ieeexplore.ieee.org/document/761853

IEEE Std 830-1998. (2007). IEEE Recommended Practice for Software Requirements Specifications. https://ieeexplore.ieee.org/document/720574

ISO/IEC/IEEE Intl Std 29148:2011. (2011). Systems and software engineering — Life cycle processes — Requirements engineering. https://ieeexplore.ieee.org/document/6146379

ISO/IEC/IEEE Intl Std 29148:2018. (2018). Systems and software engineering — Life cycle processes — Requirements engineering. https://ieeexplore.ieee.org/document/8559686

Faulk, Stuart. (2013). *Understanding Software Requirements*. https://projects.cecs.pdx.edu/attachments/download/904/Faulk_SoftwareRequirements_v4.pdf

Oracle. (2007). White Paper on "Getting Started With Use Case Modeling". Available at: https://www.oracle.com/technetwork/testcontent/gettingstartedwithusecasemodeling-133857.pdf

Sethi, Ravi. (2023). *Software Engineering. Basic Principles and Best Practices*. Cambridge Press.

Work Breakdown Structures. In *Wikipedia*, n.d. https://en.wikipedia.org/wiki/Work_breakdown_structure.

N2 Charts. In *Wikipedia*, n.d. https://en.wikipedia.org/wiki/N2_chart.

Functional Flow Block Diagrams. In *Wikipedia*, n.d. https://en.wikipedia.org/wiki/Functional_flow_block_diagram.

Structure Chart. In *Wikipedia*, n.d. https://en.wikipedia.org/wiki/Structure_chart.

Data-flow Diagram. In *Wikipedia*, n.d. https://en.wikipedia.org/wiki/Data-flow_diagram.

Object Diagram. In *Wikipedia*, n.d. https://en.wikipedia.org/wiki/Object_diagram.

System Context Diagram. In *Wikipedia*, n.d. https://en.wikipedia.org/wiki/System_context_diagram.

Storyboard. In *Wikipedia*, n.d. https://en.wikipedia.org/wiki/Storyboard.

Entity Relationship Model. In *Wikipedia*, n.d. https://en.wikipedia.org/wiki/Entity%E2%80%93relationship_model.

5. Acknowledgments

List here all sources you used to create the document and support you received from anyone not on your team.