StOut

MTM Program Product

Software Requirements Specification

*Version 0.1.1*

*9/11/2017*

Applying MTM SRS VERSION 4.0

**Project Director:** tbs

**Project Manager:** tbs

**Project Team:** tbs

**Document Author:** tbs

Version History

|  |  |  |  |
| --- | --- | --- | --- |
| *Version* | *Date* | *Authors* | *Comment* |
| 0.1 | 9/11/2017 | Jesse Anderson | Properties, Variables, Theming, and other formatting. |
| 0.1.1 | 9/13/2017 | Trevor Brooks | Added some more information from original SRS of AbOut. Also made comments for future updates. As a future note, all references to AbOut should be removed. |

TABLE CONTENTS

[1 Introduction 1](#_Toc331683624)

[1.1 Software Purpose and Scope 1](#_Toc331683625)

[1.2 Document Purpose and Contents 1](#_Toc331683626)

[1.3 Definitions, Acronyms, and Abbreviations 1](#_Toc331683627)

[1.3.1 Definitions 1](#_Toc331683628)

[1.3.2 Acronyms and Abbreviations 3](#_Toc331683629)

[1.3.3 Technical Definitions/Data Dictionary 4](#_Toc331683630)

[1.4 References 4](#_Toc331683631)

[2 General Factors 4](#_Toc331683632)

[2.1 Product Perspective 5](#_Toc331683633)

[2.2 Product Functions 5](#_Toc331683634)

[2.3 Environmental Conditions 5](#_Toc331683635)

[2.4 User Characteristic 6](#_Toc331683636)

[2.5 Dependencies 7](#_Toc331683637)

[2.6 Assumptions 7](#_Toc331683638)

[3 Analysis Use Cases 7](#_Toc331683639)

[4 Explanatory User Interfaces 8](#_Toc331683640)

[5 Specific Requirements 8](#_Toc331683641)

[5.1 Functional Requirements 9](#_Toc331683642)

[5.2 Non-Functional Requirements 9](#_Toc331683643)

[5.2.1 Design Constraints (DC) 9](#_Toc331683644)

[5.2.2 Human Factors (HF) 9](#_Toc331683645)

[5.2.3 External Interface Requirements (XI) 9](#_Toc331683646)

[5.2.4 Security (SC) 10](#_Toc331683647)

[5.2.5 Development Environment (DV) 10](#_Toc331683648)

[5.2.6 Standards (ST) 10](#_Toc331683649)

[5.2.7 Delivery Environment (DL) 10](#_Toc331683650)

[5.2.8 Performance (PR) 10](#_Toc331683651)

[5.2.9 Deliverable Items, Dates and Conditions (DD) 10](#_Toc331683652)

[5.2.10 Cost (CT) 10](#_Toc331683653)

[5.2.11 Quality (QL) 10](#_Toc331683654)

[5.2.12 V&V Activities (VV) 11](#_Toc331683655)

[5.2.13 Database (DB) 11](#_Toc331683656)

[5.2.14 Adaptability 11](#_Toc331683657)

[5.3 Requirements Models 11](#_Toc331683658)

[6 Illustrative Use Cases (IUC) 11](#_Toc331683659)

[7 Future Enhancements (FE) 12](#_Toc331683660)

# Introduction

*[This Software Requirements Specification template is designed to facilitate the definition of processes and procedures relating to software requirements specification activities. This template was developed using IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications.*

Information displayed in brackets is explanatory. Delete the bracketed text items and add your project-specific input. These items are food for thought on the section they address.

*The introduction section should explain the purpose and scope of the project software requirements specification (SRS), as well as, provide clarification of definitions, acronyms, and references. This section should also provide an overview of this document.*

*Place any material here that is not specific to any of the sub-sections below.]*

## Software Purpose and Scope

The goal of this web application is to simplify and standardize how faculty members in the Computer Science (CS) and Electrical Engineering (EE) Departments at Montana Tech assess their courses in relation to ABET student outcomes. Specifically, help CS and EE faculty members determine the extent to which students in their courses have met student outcomes by streamlining the repetitive tasks which the faculty members were doing by hand. This application should make continual course assessment easier. The customers for this web application are the faculty and staff of the CS and EE department. The web application is called AbOut, for Accreditation Board of Engineering and Technology (ABET) Outcomes.

## Document Purpose and Contents

*[This subsection should explain the purpose for writing an SRS for this project and describe the intended audience for the SRS. This subsection should describe the information that will be presented in each of the subsections from §2 on.]*

## Definitions, Acronyms, and Abbreviations

The following tables of definitions, acronyms and references may be useful for reading this document.

### Definitions

|  |  |
| --- | --- |
| Administer | See “Authorized Administrator” |
| Authorized Administrator | Member in the MTECHS domain who is registered in the AbOut system and has been assigned the administrator role for one or more programs. |
| Authorized Faculty Member | Montana Tech faculty member in the MTECHS domain who is registered in the AbOut system and has been assigned to teach one or more course offerings. Faculty members may be “active” or “inactive”. |
| Authorized Observer | Anyone with an account in the MTECHS system who has been assigned the “observer” role. |
| Authorized User | Montana Tech faculty member or staff in the MTECHS domain who is registered in the AbOut system. In addition, anyone with an account in the MTECHS system who has been assigned the “observer” role. |
| CORE | Course Outcome, Review and Evaluation (CORE) |
| CORE Report | A report which faculty members in the CS Department are required to write for each course offering which they teaching in the CS or SE degree. The AbOut web application generates statistics for this report. (See CORE Statistics) |
| CORE Statistics | Report showing the extent to which student outcomes were met by students in a course offering. |
| Course | Course contributing to measuring student outcomes and offered by the CS department. Courses are identified by a prefix and number, such as ESOF 328. The course prefixes are CSCI and ESOF. The course has a name, such as Software Requirements and Specifications. |
| Course Offering | A particular section of a course offered in a particular semester. Course offerings are identified by a course, a section, and a semester. |
| Course Outcome | Criteria which students passing the course should meet. These are specific to the course and are different than ‘Student outcomes’ which are specific to either the CS or SE program. AbOut does not measure course outcomes. This term does not appear elsewhere in this document and is included here to avoid confusing this with student outcomes. |
| Course PC Report | Report showing the extent to which students met performance criteria during the semester(s) of interest. This report is divided by performance criteria and courses. |
| Default Semester | The semester used when no semester is given (for instance, creating a course offering or a new course). |
| Faculty Member | See “Authorized Faculty Member” |
| Matrix Report | Report showing the association of courses to performance criteria and the weights of those associations. This is used to get an overview of the extent to which courses are covering performance criteria. |
| Metric | A metric associated with a course offering. The metric is created by the instructor of the course. It consists of a description, maximum number of points, and a list of the student outcomes which it measures. |
| Metric Goal | The overall (percentage) score which a student needs to meet or exceed to be considered to have met the student outcomes.  This needs to be stored in such a way so the metric goal can be changed easily. Throughout this document, it is assumed that the metric goal is 70%. |
| Modern Browser | Any web browser which reliably implements the latest (as of 2011) standards in HTML and CSS, with complete support for JavaScript. |
| Outcome Report | Report showing the extent to which students met the selected student outcome during the chosen semester. This report is divided by the performance criteria which measure the outcome. |
| PC Semester Report | Report showing the extent to which students met performance criteria during the semester(s) of interest. This report is divided by performance criteria and semesters. |
| Performance Criterion | A criterion which a program will use to assess a student outcome. |
| Performance Criterion Abbreviation | Abbreviation which will be used to identify an performance criterion. |
| Program | A degree program. This system  is to facilitate assessment for the Computer Science, Electrical Engineering and Software Engineering programs. |
| Program Abbreviation | Abbreviation which will be used to identify a program. |
| Raw data | Raw data is the number of students, weight of the course, and percentage. |
| Registered in AbOut | AbOut contains user information for this person. |
| Semester | A Montana Tech semester. Consists of a year and either fall, spring or summer. |
| Semester(s) of Interest | A single semester or a range of semesters for which a report is being generated. |
| Simple Average | An average of each value with no weighting due to characteristics of the values. |
| Student Score | The score that a student earned on a metric. |
| Student Outcome | A criterion which a program will assess for ABET accreditation. |
| Student Outcome Abbreviation | Abbreviation which will be used to identify a student outcome. |
| User | See “Authorized User” |

### Acronyms and Abbreviations

|  |  |
| --- | --- |
| AD | Adaptability |
| AL | Availability |
| AUC | Analysis Use cases |
| CM | Communications |
| CT | Cost |
| DB | Database |
| DC | Design Constraint |
| DD | Delivery Data and Conditions |
| DL | Delivery Environment |
| DV | Development Environment |
| EN | Enhanceability/Extendibility |
| FE | Future Enhancements |
| HF | Human Factors |
| HW | Hardware |
| IUC | Illustrative Use Cases |
| ML | Maintainability |
| OP | Operations |
| PR | Performance |
| PT | Portability |
| QL | Quality |
| RL | Reliability |
| SC | Security |
| SDD | Software Design Description |
| SRS | Software Requirements Specification |
| ST | Standards |
| SI | Site |
| SW | Software |
| UB | Usability |
| VV | Verification & Validation |
| XI | External Interfaces |
| XXX | X of X of X |

### Technical Definitions/Data Dictionary

*[This subsection shall list (alphabetically) and briefly describe all data collections and items mentioned in this SRS. If a data base is not involved the technical items names in the requirements are defined here. These names should be chosen with care. The expectation is that these names will be used later in the design and implementation.]*

|  |  |  |
| --- | --- | --- |
| ItemName | Type | Brief description of data item |
| First item | Table | A description |

## References

ABET, <http://www.abet.org/>

CAS, <https://wiki.jasig.org/display/CAS/Home>

CS Department Student Outcomes, <http://cs.mtech.edu/main/index.php/component/content/article/146>

W3C XHTML validation software, [http://validator.w3.org](http://validator.w3.org/)

W3C CSS validation software, <http://jigsaw.w3.org/css-validator>

# General Factors

*[The General Factors section should describe the general factors that affect the product and its requirements. Place any material here that is not specific to any of the sub-sections.*

*In this and each subsequent major section, briefly describe the purpose of this section from the readers perspective.]*

## Product Perspective

This web application will be independent of other products except CAS (see Section 2.5, Dependencies).

## Product Functions

This section provides a high-level overview of the functionality of the web application.

#### Function Overview

The AbOut web application will be used to:

* Record student outcomes associated with the computer science, electrical engineering and software engineering programs
* Associate performance criteria with student outcomes
* Associate performance criteria with courses
* Associate faculty and students with course offerings
* Enable faculty to record metrics of the course offerings they teach
* Enable faculty to record the scores which a student earned on an metric
* Generate a variety of reports indicating the extent to which performance criteria were met
* Allow observers to see reports and all of the information leading to the report, with student names redacted

These functions are divided into three overlapping sets: faculty, administrative, and reporting functions.

#### Administrative Functions

Authorized faculty members and administrators are able to do the following:

* Add, edit, delete and view users of the AbOut system
* View the semesters in the system and set a default semester
* Tell the system to generate the next chronological semester
* Add, edit, delete and view student outcomes
* Add, edit, delete and view performance criteria for student outcomes
* Add, edit, delete and view courses, along with the performance criteria associated with them
* Add, edit, delete and view offerings of courses
* Add, edit, delete and view students in a course offering
* Import a list of students into a course.

Note that administrators are not able to add, edit or delete metrics associated with a course offering, and that when administrators view student scores on a metric, student names will be redacted.

If a user is both a faculty member and an administrative, that user will be able to view and edit the information associated with a course offering which they teach.

#### Faculty Functions

Authorized faculty members are able to do the following:

* View the course offerings that they are currently teaching or have taught in the past
* Add metrics to course offerings that they have taught or are teaching
* Add or remove students from course offerings that they taught or are teaching
* Enter student scores on the metrics in the course offering which they taught or are teaching
* Export a list of the students in an offering they have taught or are teaching.

#### Reporting Functions

Authorized faculty members and administrators are able to do the following.

* Generate CORE statistics showing the extent to which students enrolled in a course offering met the performance criteria associated with that offering
* Generate a Course PC Report showing the extent to which students met performance criteria per course during the semester(s) of interest.
* Generate a Matrix Report showing the weights associated with courses assessing performance criteria for a specified program. Note that this report generates the information for the current semester.
* Generate an Outcome Report showing the extent to which students met performance criteria for a semester.
* Generate a Course PC Semester Report showing the extent to which students met performance criteria per semester over the semester(s) of interest.

## Environmental Conditions

AbOut will be a web application which is accessed from the CS Department website. The CS Department will need a web and database server in order to serve AbOut.

Users will need a Montana Tech account in the MTECHS domain to access this system. They will also need to be registered within the AbOut system, i.e. AbOut needs to contain the user id for this person.

## User Characteristic

The primary users of this system are the faculty and staff of the CS Department. Someone who has a username in the MTECHS, is registered in AbOut and has been assigned the role of “observer” for one or more programs will also be a user of this system. An understanding of the assessment process, a familiarity with web browsers, and proficiency completing forms on a computer, is assumed.

Users can be assigned one or more of the roles: administrator, faculty, and observer.

## Dependencies

AbOut will utilize the Central Authentication Service (CAS). CAS is a single sign-on protocol for the web. CAS allows web applications to authenticate users without gaining access to a user’s security credentials.

CAS is offered by Montana Tech’s Campus Technology Services and is used to authenticate Montana Tech users for most campus applications. Without CAS, users of AbOut would need to create and remember another username / password combination to login to AbOut.

## Assumptions

*[This subsection should list all assumptions that on which the software resulting from the SRS will depend that have not been covered above. This subsection should be the source for recognizing the impact of any changes to these assumptions on the SRS and resulting software.. This section can highlight unresolved requirement issues that should be recorded on the Project Manager’s Open Issues List.]*

# Analysis Use Cases

## Actors

|  |  |  |
| --- | --- | --- |
| Primary Actor | Description | Use Cases |
| Administrator | A permission that allows users to do administrative functions for assessments. This permission is meant for administrative staff.    Examples:  Secretaries and Department heads | Add/edit/delete/view student courses  Add/edit/delete/view student outcomes  Add performance criteria to student outcome  Edit/delete/view performance criteria  Add/edit/delete/view users  Add/edit/delete/view course offerings  Generate next semester  Import lists of students  Set default semester  View semesters in system |
| Observer | A permission that allows the user to view data, with student names redacted. These users are not allowed to modify data. | View student outcomes  View courses  View course offerings  View metrics associated with a course offering  Generate C.O.R.E. Report  Generate Course PC Report  Generate Matrix Report  Generate Outcome Report  View data of a class, with student names redacted. |
| Faculty member active | A permission that allows the user limited add, edit, view, and delete permissions to offerings they are associated with. The user is also allowed to generate C.O.R.E statics. | Add/remove students from course offering  Add/edit/delete/view metric within course offering  Add score(s) to metric  Export list of students  View course offering  Generate C.O.R.E. statistics |
| Faculty member inactive | A permission for ex faculty members. Meant to help preserve data for historical reference. | None |

## Use Cases

*[Use cases can be used in software requirements engineering for two different purposes: (1) to generally capture, and analyze client/user information about the proposed system or (2) to illustrate specific requirements and operational scenarios and possibly provide additional low-level requirements details. The second purpose is covered in the Illustrative Use Cases section below. This optional section is to be used for the first purpose. These use cases are often developed with the client. These use cases are primarily textual but may include “stick figure” diagrams.*

*If analysis use cases were not developed for this product, this section should read:* Analysis Use Cases were not developed for this specification*.]*

# Explanatory User Interfaces

*[This is an optional section that is used when providing the user with information that could be helpful in understanding the specific requirements in the next section.*

*If this section is used, care must be taken that the general descriptions given here are not presented as requirements.]*

# Specific Requirements

*[The Specific Requirements section should contain all the requirements for the subject software. The details within this section should be defined as individual, specific requirements. Each specific requirement should be stated such that its achievement can be objectively verified by observation, inspection, usability testing, functional testing, analysis, or a combination of these. The method verification must be described. Each requirement should be clearly identified for tracking.]*

## Functional Requirements

*[This subsection should specify how the software product will react to every possible input situation. It describes all the actions that must take place in the software in response to every input. Pertinent changes in the environment are considered to be inputs.*

*Care must be taken to avoid dropping into design details. In the user cannot directly experience the effect of a requirement it probably crossed the line into design.*

*Functional requirements should be logically grouped. Each group should have a short, unique (within the SRS) abbreviation and a number. The word processing section number will probably change as the SRS is developed.*

*For each identified requirement an optional rationale for that requirement may be given.*

*Most modern software should provide at least a modicum of user help. For very complex applications in situ help may be supplemented by a user’s manual (or manual page) but for many simple applications comprehensive in situ help is sufficient.]*

## Non-Functional Requirements

*[This subsection should specify both the static and dynamic numerical requirements placed on the software or human interaction with the software. All the identifiers for requirements in this section should begin with the two letter abbreviation shown below]*

### Design Constraints (DC)

*[Sometimes a client will require certain design constraints, for example the use of a certain system configuration or the use of particular algorithm. Such constraints are described in this subsection.]*

### Human Factors (HF)

*[Not everyone has the same inherent mental and physical capabilities vis-à-vis a given computer application. For example if sound is part of the application, will other clues be given that will enable a hard of hearing user to use the proposed application as well as person with normal hearing; similarly for color blindness. Some these factors have to be defined and validated in specially equipped usability laboratories.]*

### External Interface Requirements (XI)

#### Hardware (HW)

#### Software (SW)

#### Communications (CM)

### Security (SC)

### Development Environment (DV)

### Standards (ST)

### Delivery Environment (DL)

#### Site (SI)

*[This subsection should specify any requirements for installation or operation of the software that might change the pre-existing configuration of the user site.]*

#### Operations (OP)

*[This subsection should specify normal and special operations required by the user to include:*

* *Various modes of operation within the user organization*
* *Periods of interactive operations and unattended operations*
* *Data processing support functions*
* *Backup and recovery operation.]*

### Performance (PR)

### Deliverable Items, Dates and Conditions (DD)

### Cost (CT)

### Quality (QL)

#### Reliability (RL)

*[Reliability is specified as mean-time-to failure of an operational item. An operational profile must be specified.]*

#### Availability (AL)

#### Maintainability (ML)

*[Failures can be classified as occurring in either operational or non- operational delivered items Failures in operational items can be classified by the work products that must be changed to eliminate that failure: code only, code and design, code, possibly design, and requirements. For each class of failure what is the maximum estimated effort required to eliminate that failure and what is the rationale for this estimate.]*

#### Usability (UB)

#### Enhanceability/Extendibility (EN)

*[If the future it might be necessary to change the Functional requirements in specified ways, what is the maximum estimated effort required to make such changes and what is the rationale for this estimate?]*

#### Portability (PT)

*[If in the future it might be necessary to change the above Development or Delivery Environments (DV or DL) to other specified environments, what is the maximum estimated effort required to implement such changes and what is the rationale for this estimate]*

### V&V Activities (VV)

### Database (DB)

*[This optional subsection that specifies requirements for any database to be developed as part of the product. The information in this section will include:*

* *Types of information to be stored*
* *Table attributes (queried, supporting, updated)*
* *Frequency of access*
* *Accessing capabilities and requirements*
* *Data elements and file descriptors*
* *Retention requirements for data.]*

*Care must be taken here to avoid design details. Unless so requested by the client this section should only contain as much information about saved data as is necessary to fully document any of the requirements given above. Since this is the last Non-functional sub-section this sub-section would not appear in the document if the requirements did not involve any data bases information.]*

### Adaptability (AD)

*[If it is specified that in the future it might be necessary to change any of the above Non-Functional requirements, what is the maximum estimated effort required to implement such changes and what is the rationale for this estimate.]*

## Requirements Models

*[This optional subsection, if present, provides models of the functional requirements to aid in clarifying and validating these requirements. A Z language specification is a good example. This sub-section may be skipped entirely if this SRS does not use any requirements models.]*

# Illustrative Use Cases (IUC)

*[This optional section should begin with a hierarchical, logically complete breakdown of all the execution conditions delineated in the functional requirements. Subsections should give detailed use cases for the most important of these conditions. If illustrative use case would not help readers understand the requirements this section should read:* Illustrative User Cases are not developed for this specification.*]*

# Future Enhancements (FE)

*[This section should describe any future enhancements that are contemplated at the time this SRS completed. If there is no known possibility that this product will be enhanced in the future this section should read :* It is not expected that there will be any future enhancements to this product.*]*