**CIS 422**

**Software Requirements Specification (SRS) for UNAMED\_CALENDAR**

The document in this file has been adapted from the IEEE Guide to Software Requirements Specifications (Std. 830-1993)

**CIS 422**

Team Number 6

499ms

Software Requirements Specifications

Version 1 Date: (01/14/2019)

Table of Contents

[**1. Introduction**](#_v2a23oazh367) **3**

[1.1. Purpose](#_qywwjnle4krl) 3

[1.2. Scope](#_wg034dfx8r87) 3

[1.3. Definitions, Acronyms, and Abbreviations](#_e1u5x72qdtom) 3

[1.4. References](#_tkcm9f2mb1vx) 4

[1.5. Overview](#_5xt2423bdi4k) 5

[**2. Overall Description**](#_d6dklkhlf6v2) **5**

[2.1. Product Interaction](#_t3b9ae1qabi1) 5

[2.1.1. System Interfaces](#_msmufzcn9x7g) 5

[2.1.2. Software Interfaces](#_61xqmqwjhpxf) 6

[2.1.3. Communication Interfaces](#_9qfiwuvpieod) 6

[2.1.4. Memory Constraints](#_unn3bojkwuo1) 6

[2.2. Product Use and Case Scenarios](#_46o0nk7rw5zx) 6

[2.3. User Characteristics](#_abean3gh1xfo) 8

[2.4. Constraints](#_3z2ksst2ymbo) 8

[2.5. Assumptions and Dependencies](#_kw3hb0810kwi) 9

[**3. Requirements and Additional Specifications**](#_b3bgkzb64r51) **9**

[3.1. Functional Requirements](#_37ieajw31av6) 9

[3.1.1. Data Collection and Output](#_kfs4w8pkein0) 9

[3.1.2. Events](#_p4du73ncng4h) 9

[3.1.3. Program Interface Properties](#_y4xsaixsy2go) 10

[3.1.4. Calendar Operations](#_lkgkwodm52vn) 10

[3.2. Nonfunctional Requirements](#_mbp3ozzfrt82) 10

[3.2.1. Software Performance](#_7i7iwm7vp1v1) 10

[3.2.2. Hardware](#_reefym7zbb3k) 10

[3.2.3. Ease of Development](#_ebwj8ccwramb) 11

[3.2.4. Output File](#_kluezeyztc89) 11

[3.2.5. Security with Input Format](#_t5a2he8vhx3w) 11

[3.2.6. System Documentation](#_pjrcc3kmiymc) 11

[3.3. Additional Specifications](#_d9jgk9gq99pa) 11

[3.3.1. Program Interface Properties](#_rusu0rjc9who) 11

[3.3.2. Calendar Layout](#_nbbkecnuvu18) 11

[3.3.3. Data Collection and Output](#_ivjbsgtedulr) 12

## 

# **1. Introduction**

## 1.1. Purpose

This Software Requirements Specification (SRS) describes the computer program UNAMED\_CALENDAR that is currently being designed and built at the University of Oregon by students: Benjamin Yin, Chase Craig, Noah Palmer, Refael Yehuda,and Zachary Bower. UNAMED\_CALENDAR will implement features explained in the Project 1 description assigned by A. Hornof.

UNAMED\_CALENDAR will solve an end users organizational needs by providing the end user with a calendar application with the ability to add, edit, and remove events specified by the user.

As the software evolves this document will contain the current specifications for said features. This document is for developers and end consumers of UNAMED\_CALENDAR. This document will allow all developers of UNAMED\_CALENDAR to organize, plan and track all aspects of development. This SRS will outline objectives of the software including functional/non functional features, an overview of expected user experience, a clear measure of accountability, and clear system requirements.

## 1.2. Scope

This software will be a calendar application for end users to store, edit, add and remove events. This software will be designed to maximize end users productivity while keeping the design and layout intuitive. By focusing on these endpoints from the beginning of the development cycle this will allow UNAMED\_CALENDAR to meet an end users needs.

1. The software to be produced is named UNAMED\_CALENDAR
   1. The features of this software are named in the functional/non functional section of this SRS.
   2. Multiple scenarios related to users and their experience consuming UNAMED\_CALENDAR will be presented in this SRS.
2. The goals and objectives of a final product will be listed in this SRS.
3. All system requirements will be provided in this SRS

## 1.3. Definitions, Acronyms, and Abbreviations

This document will reference multiple acronyms definitions for each are defined below:

**Software Design Specifications** (SDS) : A document describing the built product. The SDS describes external visible behavior as precise as possible.

**Initial Project Plan** ( IPP ) : A document outlining the process from beginning to completion of UNAMED\_CALENDAR. This document will include : a management plan, multiple milestones including dates and items required, an outline of scheduled meetings, the responsibilities of each member, the coding style requirements for the development cycle, and a rationale behind each these choices.

**Functional Requirement** ( FR ) : A specification for the required behaviour of a system or component.

**Non-functional Requirement** ( NFR ) : A specification for the judgement of the operation of a system or component.

**User Experience** ( UX ) : The overall experience of the end consumer using UNAMED\_CALENDAR.

**Coding Style Document** ( CSD ) : A document outlining the expected coding style for all developers of UNAMED\_CALENDAR to follow. This will include: parentheses spacing, function naming, variable naming, file naming, comments, and more.

**Citation Expectations Document** ( CED ) : A document outlining the expected style of how a developer of UNAMED\_CALENDAR is to cite references used.

## 1.4. References

*References related to the internal development*

1. **SDS:** <https://docs.google.com/document/d/1WQpj-drVOHYIO6baY02JZA9RLvReq6zHXqaqdEbQN-Q/>
2. **IPP** : <https://docs.google.com/document/d/1ZszO4mlShik_dEfztpab-m4zGZj5UHmMHrU0SFNLGkg/edit>
3. **CSD**: <https://docs.google.com/document/d/1KL8Q_9lhUDCnlcPMZtaPfaDTiaKeyBGpLgd7bN3cE5s/>
4. **CED:**
5. **Project Requirements**
   1. <https://classes.cs.uoregon.edu/19W/cis422/P1.html>
   2. <https://classes.cs.uoregon.edu/19W/cis422/P1_Grading.html>

*References used to create this document*

1. **NRL Dual Task SRS:** <https://www.cs.uoregon.edu/research/cmet/Multimodal/files/SRS/070304/dualtask.html>
2. **SRS Template:** <https://web.cs.dal.ca/~arc/teaching/CS3130/Templates/SRS%20and%20Project%20Plan%20Templates/SRS-template2.doc>

## 1.5. Overview

Section two of this document gives an overview of the functionality of UNAMED\_CALENDAR. The intended audience for this section are end users of UNAMED\_CALENDAR. It describes the functional and nonfunctional requirements an end user should expect upon development cycle end. The functional and nonfunctional requirements are used to establish the technical specifications presented in section three of this document.

Section three of this document give an overview of the technical aspects of the project. The intended audience for this section are developers. This section shall describe in detail the functionality of the product.

# **2. Overall Description**

## 2.1. Product Interaction

UNAMED\_CALENDAR will have one main interface for end users to interact with. The initial screen will have the current date within the current month (including dates) displayed to the user in a matrix format. The month and year will be centered in this matrix display, with left and right arrow right justified.

From this main interface the user will be able to navigate to the previous month by clicking on the left arrow, the next month by clicking on the right arrow, or select the month and year to entered the date desired by the user.

The end user will have the ability to edit any active events. This action will be accomplished by the user double clicking on the event, or selecting an edit icon described further in the SDS.

The end user will have the ability to add an event at any time from the current date, to an end date specified in the requirements section. This action may be accomplished by the user selecting a button labeled ‘add event’. This action may be accomplished by the user selecting the date, selecting the time range and adding the event via a newly displayed interface.

A detailed overview of the user software interaction are available in both the SDS and section 2.2.

### *2.1.1. System Interfaces*

UNAMED\_CALENDAR is a GUI driven application. The user will perform all interactions described in the SDS and section 2.1. The GUI design and interactions are available in the SDS.

### *2.1.2. Software Interfaces*

UNAMED\_CALENDAR is designed for and tested with the following software requirements. Deviating from these requirements may give undesired results.

1. Java Runtime Environment
   1. Oracle
   2. <https://www.java.com>
2. GNU/Linux Operating System
   1. Debian
   2. Ubuntu 18.04
   3. Arch
3. MySQL
   1. Oracle
   2. <https://www.mysql.com>

### *2.1.3. Communication Interfaces*

The end user will communicate through a mouse and keyboard which UNAMED\_CALENDAR will take as input through inputs located with the GUI.

UNAMED\_CALENDAR shall communicate to the user through a standard monitor or display. The display specifics are outlined in section 3.

UNAMED\_CALENDAR uses a database outlined in section 3. End users are expected to have an internet connection of at least 256Kbps to ensure synchronization of database.

### *2.1.4. Memory Constraints*

UNAMED\_CALENDAR requires 128MB of memory

UNAMED\_CALENDAR requires at least 200 MB of disk space

## 2.2. Product Use and Case Scenarios

All examples assume the end user meets the minimum requirements for UNAMED\_CALENDAR. Unless otherwise specified the users assumed starting location is the main interface displayed on program startup described in section 2.1.

*User wants to create an event via the add event button*

1. The user clicks the add event button
2. A new dialogue box appears in front of the calendar view
3. The user must input the following required fields
   1. Start date
   2. End date
   3. Start time
   4. End time
4. The user may input the following optional fields
   1. Event Description
   2. Event Name
   3. Event Type
   4. Recurring event
   5. All day event
5. The user selects the confirmation button
6. A message appears to the user confirming the event has been added

*User wants to edit an event via the calendar view*

1. The user navigates to the date the desired event is to be added
   1. The user may navigate via left and right arrows located in the top right
   2. The user may navigate via a search field in the top center
2. The user selects the space corresponding to the date in the calendar view
3. If no events are scheduled for that day a dialogue box appears
4. If events are scheduled for for that day, the detailed events view is displayed
   1. Within the detailed events view, the user may select a time range via mouse input
5. The user must input the following required fields
   1. Start date
   2. End date
   3. Start time
   4. End time
6. The user may input the following optional fields
   1. Event Description
   2. Event Name
7. The user selects the confirmation button
8. A message appears to the user confirming the event has been added

*User wants to delete an event*

1. The user navigates to the date the desired event is to be deleted
   1. The user may navigate via left and right arrows located in the top right
   2. The user may navigate via a search field in the top center
2. The user selects the space corresponding to the date in the calendar view
3. The detailed events view is displayed
   1. Within the detailed events view, any events scheduled for that date are displayed
4. The user selects the event via mouse input
5. The user may press the backspace/delete key to delete event
6. The user may select the delete icon (describe in SDS)
7. A dialogue box appears in front of the detailed events view requesting action confirmation
8. If cancel is selected the user is taken to the detailed events view without deleting the event
9. If yes is selected the user is taken to the detailed events view with the event no longer present

*User wants to save changes manually*

1. The user navigates to the file menu located in the top left of the application
2. The user navigates to the save sub menu item
3. A dialogue box appears in front of the calendar view
4. If cancel is selected the dialogue box is no longer displayed, with changes being unsaved
5. If yes is selected the dialogue box is no longer displayed, with changes being saved

*User saves upon exit*

1. From any point in the application the user selects the standard close icon
   1. The icon may change between users and will depend upon such factors as: operating system, window manager and user preference.
   2. Any users unaware of the location of the icon should consult the documentation for their specific operating system, window manager and user preference.
2. If unsaved changes have been detected
3. A dialogue box appears in front of the users current window
4. If no is selected the application terminates without saving the changes
5. If cancel is selected the application returns to the state prior to the close button selection
6. If yes is selected the application saves and terminates.

## 2.3. User Characteristics

UNAMED\_CALENDAR will be distributed as a source build. The end user is expected to have some form of technical literacy to build the project, or follow instruction for building the project from source. The end user should have a basic understanding of the GNU/Linux operating system, with the ability to change user permissions.

The end user will expect UNAMED\_CALENDAR to organize the end users events if the end user updates UNAMED\_CALENDAR consistently. The end users satisfaction will be proportional to their use of UNAMED\_CALENDAR.

To solve minor troubleshooting issue and questions the end user will be expected to read and use help documentation. The help documentation will provide answers for many shared questions regarding the user of UNAMED\_CALENDAR.

## 2.4. Constraints

To ensure UNAMED\_CALENDAR performs as described in section 2.1. The end user is expected to meet or exceed the software, hardware, and communication requirements described in section 2.1.

## 2.5. Assumptions and Dependencies

UNAMED\_CALENDAR consists of functional and nonfunctional requirements described in section 3. An end user should expect all requirements to work as documented given the software, hardware, and peripheral requirements given in section 2.1 are met.

# **3. Requirements and Additional Specifications**

## 3.1. Functional Requirements

The following features are defined for the minimum viable product and thus are defined to be absolutely required.

### *3.1.1. Data Collection and Output*

**FR 1.1.1:** The system shall ensure all dates and times are stored in standard format (such as “DD/MM/YYYY HH:MM”).

**FR 1.1.2:** The system shall ensure all dates and times are displayed in standard format.

**FR 1.1.3 Data Entry**

**FR 1.1.3.1:** The system shall ensure all dates and times are entered in standard format.

**FR 1.1.3.2:** The system shall accept any valid date and time combination from 01/01/1970 00:00 to 31/12/2119 23:59.

**FR 1.1.3.3:** The system shall alert the user if their input is malformed or out of range.

**FR 1.1.4 File Saving**

**FR 1.1.4.1:** The system shall store all events in a local file (referenced hereafter as “save file”) on the user’s file system.

**FR 1.1.4.2:** The system shall store a version identifier with the save file for version matching.

**FR 1.1.4.3:** The system shall store a event starting token at the start of every event in the save file.

### *3.1.2. Events*

**FR 1.2.1:** The system shall allow the user to create new events by specifying the date, and the time the event shall occur on.

**FR 1.2.1.1:** The system shall reject the creation of a new event if it is missing either the date or the time, however it should not clear what the user has already added to the event.

**FR 1.2.2:** The system shall define the minimum requirements for an event as an object holding a title (a string), a valid date and a valid time.

### *3.1.3. Program Interface Properties*

**FR 1.3.1:** The system shall ensure that the program interface accept standard window operations (such as minimize window, maximize window and close window).

**FR 1.3.2:** The system shall interrupt “close window” requests and prompt the user if there is unsaved data.

### *3.1.4. Calendar Operations*

**FR 1.4.1:** The system shall allow for the addition of an event, of which will be stored in memory until saved.

**FR 1.4.2:** The system shall allow for the saving of all temporary changes in memory through a standard save icon or through a standard key combination (such as <Ctrl> + <S>).

**FR 1.4.3:** The system shall allow for the selection of an event.

**FR 1.4.4:** The system shall allow for selected events to be modified.

**FR 1.4.4.1:** The system shall ensure that when editing the event that any information not changed still remains in the new event with the changes.

**FR 1.4.5:** The system shall allow for selected events to be deleted.

## 3.2. Nonfunctional Requirements

The following features are defined for the minimum viable product and thus are defined to be absolutely required.

### *3.2.1. Software Performance*

**NFR 1.1.1:** The software shall not stall for more than 500 ms on any operation blocking user interaction besides initialization.

**NFR 1.1.2:** The software shall not require more than 128 MB of memory to run.

**NFR 1.1.3:** The software shall not require more than 200 MB of space for source code once compiled.

### *3.2.2. Hardware*

**NFR 1.2.1:** The software will run on GNU/Linux Operating System.

**NFR 1.2.2:** The software shall be developed to run on most machines that can handle java environments that contains a native display.

### *3.2.3. Ease of Development*

**NFR 1.3.1:** The software will be designed in a modular fashion such that simple changes will remain localized to a single system component.

### *3.2.4. Output File*

**NFR 1.4.1:** The system shall use UTF-8 for all output files.

### *3.2.5. Security with Input Format*

**NFR 1.5.1:** The system shall only allow standard ASCII characters to be typed into any textfield.

### *3.2.6. System Documentation*

**NFR 1.6.1:** The SRS (the document you are reading now) must be editable by all developers. A numbering and indentation scheme will be used that permits easy reference, update and modification by all developers. All versions of the SRS will list the authors and the date.

## 3.3. Additional Specifications

The following features are defined as NOT absolutely required to be included in the minimum viable product.

### *3.3.1. Program Interface Properties*

**FR 2.1.1:** The system should display any “week” like object as starting on Sunday.

**FR 2.1.2:** The system shall allow the user to resize the program interface.

**FR 2.1.3:** The system should keep the program interface limited to becoming no smaller than 600x800.

**FR 2.1.4:** The system should ensure that display elements on the program interface will resize appropriately for any given size of the program interface.

**FR 2.1.5:** The system shall start the calendar in the month overview (discussed in FR 2.2.1)

### 3.3.2. Calendar Layout

**FR 2.2.1:** The software shall contain two views to display to the user, a month overview and a day overview.

**FR 2.2.1.1:** The software shall ensure that in the month overview that the current day of the month is made differentiated from the other days of the month.

**FR 2.2.1.2:** The software should ensure that in the month overview that any days that have passed in the month are differentiated from days in the future.

**FR 2.2.1.3:** The software should allow for the selection of events by left clicking once on the occurrence.

**FR 2.2.1.4:** The software should allow for editing an event by double left clicking on the occurrence, with no more than 250 ms in between the clicks.

**FR 2.2.1.5:** The software should allow for adding an event by single left clicking a day in the month overview. The new event should have the date autofilled.

**FR 2.2.1.6:** The software should allow for adding an event by single left clicking an time slot in the day overview. The new event should have the time and date autofilled.

**FR 2.2.2:** The software shall only display a reasonable number of events in a single day while the user is month overview.

**FR 2.2.2.1:** The software shall display a icon (such as “...”) on the day suggesting that there are more events than what can be displayed.

**FR 2.2.3:** The software shall only display a reasonable number of events at one time while the user is in day overview.

**FR 2.2.3.1:** The software shall have a scrolling view so that the user may view other events in the case where not all events can be displayed at once.

**FR 2.2.4:** The software shall allow for the user to return to month overview through a simple key press (such as <Esc>) and through clicking an icon (such as a return arrow).

**FR 2.2.5:** The software shall allow for jumping to the previous or the next month in the month overview through a simple key press (such as <Left Arrow> or <Right Arrow>) or through clicking an icon.

**FR 2.2.6:** The software shall allow for jumping to the previous or the next day in the day overview through a simple key press (such as <Left Arrow> or <Right Arrow>) or through clicking an icon.

**FR 2.2.7:** The system shall ensure that any events listed shall be sorted in according to the time of which they occur in. In events of ties, the system will sort them in accordance to their titles.

### 3.3.3. Data Collection and Output

**FR 2.3.1 File Saving**

**FR 2.3.1.1:** The system shall accept any save file that has originated from a previous version of the application.

**FR 2.3.1.2:** The system shall, if it determines that the save file in the user’s file system is from a previous version, update the file to the most current version.

**FR 2.3.1.3:** The system shall, if it determines that the save file in the user’s file system is lacking in required fields, it is to treat the event as malformed and it is to be skipped.