Software Requirements Specification

For

On-Call Assistant

[01/23/2015]

[2.1]

Prepared by: Team 3

**Table of Contents**

[Revision History 3](#_Toc409958281)

[1 Introduction 4](#_Toc409958282)

[1.1 Overview 4](#_Toc409958283)

[1.2 Goals and Objectives 4](#_Toc409958284)

[1.3 Scope 5](#_Toc409958285)

[1.4 Definitions 5](#_Toc409958286)

[1.5 Document Conventions 6](#_Toc409958287)

[1.6 Assumptions 6](#_Toc409958288)

[2 General Design Constraints 6](#_Toc409958289)

[2.1 Product Environment 6](#_Toc409958290)

[2.2 User Characteristics 7](#_Toc409958291)

[2.3 Mandated Constraints 7](#_Toc409958292)

[2.4 Potential System Evolution 8](#_Toc409958293)

[3 Nonfunctional Requirements 8](#_Toc409958294)

[3.1 Usability Requirements 8](#_Toc409958295)

[3.2 Operational Requirements 8](#_Toc409958296)

[3.3 Performance Requirements 8](#_Toc409958297)

[3.4 Security Requirements 9](#_Toc409958298)

[3.5 Safety Requirements 9](#_Toc409958299)

[3.6 Legal Requirements 9](#_Toc409958300)

[3.7 Other Quality Attributes 9](#_Toc409958301)

[3.8 Documentation and Training 9](#_Toc409958302)

[3.9 External Interface 9](#_Toc409958303)

[3.9.1 User Interface 9](#_Toc409958304)

[3.9.2 Software Interface 10](#_Toc409958305)

[4 System Features 10](#_Toc409958306)

[4.1 Feature: Time Away Specification 10](#_Toc409958307)

[4.1.1 Description and Priority 10](#_Toc409958308)

[4.1.2 Use Case: Future Time Away 10](#_Toc409958309)

[4.1.3 Use Case: Past Time Away 11](#_Toc409958310)

[4.1.4 Additional Requirements 11](#_Toc409958311)

[4.2 Feature: Employees List Editing 11](#_Toc409958312)

[4.2.1 Description and Priority 11](#_Toc409958313)

[4.2.2 Use Case: Adding an Employee 11](#_Toc409958314)

[4.2.3 Use Case: Removing an Employee 12](#_Toc409958315)

[4.2.4 Use Case: Update Employee Information 12](#_Toc409958316)

[4.2.5 Additional Requirements 12](#_Toc409958317)

[4.3 Feature: Employees List Editing 12](#_Toc409958312)

[4.3.1 Description and Priority 12](#_Toc409958313)

[4.3.2 Use Case: Generate Schedule 13](#_Toc409958314)

[4.3.3 Additional Requirements 13](#_Toc409958317)

[4.4 Feature: Schedule Editor 13](#_Toc409958312)

[4.4.1 Description and Priority 13](#_Toc409958313)

[4.4.2 Use Case: Generate Updated Calendar 13](#_Toc409958314)

[4.4.3 Additional Requirements 13](#_Toc409958317)

[4.5 Feature: Read-only version of Calendar 13](#_Toc409958312)

[4.5.1 Description and Priority 13](#_Toc409958313)

[4.5.2 Use Case: Generate Calendar 13](#_Toc409958314)

[4.5.3 Additional Requirements 14](#_Toc409958317)

[4.6 Feature: Total Time Away 14](#_Toc409958312)

[4.6.1 Description and Priority 14](#_Toc409958313)

[4.6.2 Use Case: Calculate total time away 14](#_Toc409958314)

[4.6.5 Additional Requirements 14](#_Toc409958317)

[4.7 Feature: Total OnCall Rotations 14](#_Toc409958312)

[4.7.1 Description and Priority 14](#_Toc409958313)

[4.7.2 Use Case: Calculate and view total time on rotation 14](#_Toc409958314)

[4.7.3 Additional Requirements 14](#_Toc409958317)

Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Name** | **Description** |
|  | **1/23** | **Samaa Gazaa** | **Added initial features and use cases based off of project scope** |
|  | **1/24** | **Adam Guerrero** | **Collaborated to review features and use cases** |
|  | **1/25** | **Samaa Gazaa** | **Inserted features and use cases relative to system requirements and project scope** |
|  | **1/26** | **Adam Guerrero** | **Edited Use cases and document index** |
|  | **1/27** | **Mike Harris** | **Formatted and organized project document** |
|  | **2/1** | **Adam Guerrero** | **Update for Operational requirements** |
|  | **2/5** | **Adam Guerrero** | **Update** |
|  | **2/9** | **Adam Guerrero** | **Product Backlog Update** |
|  | **2/15** | **Adam Guerrero** | **Formatted and organized project document** |
|  | **2/20** | **Mike Harris** | **Update** |
|  | **2/20** | **Adam Guerrero** | **Edit and Review** |
|  | **2/20** | **John** | **Edit and Review** |

# Introduction

## Overview

The On-Call Assistant software is being developed to help managers generate equitable on-call rotation schedules for the team(s) they manage. This document outlines the purpose of the software and includes a high level view of the software design. Included are sections for primary and secondary users as well as functional and nonfunctional requirements, features, and constraints.

This document will identify functional and nonfunctional requirements as well as implementation constraints. The document will act as a handbook to those within the organization seeking more thorough understanding of the Scheduling Assistant Software. The Software Requirement Specification handbook will include a list of general product features as well as detailed system behavior needed to deliver these features. Detailed system behavior will be expressed as use cases or usage scenarios.

The On-Call Assistant is a web-based application designed to assist upper management with generating and maintaining schedules of on-call rotations. The On-Call Assistant seeks to distribute rotations competently and equitably among team members with each schedule it creates. By automatically generating these schedules, the application will reduce the amount of time managers spend manually creating such schedules.

## Goals and Objectives

1. Provide a usable and efficient system that allows managers to view current and past schedules and to create dependable future schedules.
2. Create reliable schedules that minimize the effort that, in the past, has been required to manually create and update a schedule of on-call rotations. In return this should increase workflow and decrease confusion.
3. The On-Call Assistant will offer a simplistic and user friendly interface. The user will not have to worry about managing complex back-end storage and development. Should the user require more complex functionality, such functionality will be available.

## Scope

The On-Call Assistant will create an equitable schedule of on-call rotations for team members based on the following considerations:

* When team members have requested out-of-office time
* Previous rotations for which team members have been scheduled
* Previous rotations that include paid holidays for which team members have been scheduled

The system, intended for a single user, will operate in a browser hosted locally on the user’s machine. As such, it will not provide security or user authentication measures.

~~System features are described below. The system features section below does specify exactly what will be included in the system; however, it is not presented in a way that makes clear functionality at the boundaries of the system.~~

## Definitions

~~This section defines potentially unfamiliar or ambiguous words, acronyms and abbreviations.~~

**Use case** – describes a goal-oriented interaction between the system and an actor. A use case may define several variants called scenarios that result in different paths through the use case and usually different outcomes.

**Scenario** – one path through a use case

**Actor** – user or other software system that receives value from a use case.

**Role** – category of users that share similar characteristics.

**Product** – what is being described here; the software system specified in this document.

**Project** – activities that will lead to the production of the product described here. Project issues are described in a separate project plan.

**Shall** – adverb used to indicate importance; indicates the requirement is mandatory. “Must” and “will” are synonyms for “shall”.

**Should** – adverb used to indicate importance; indicates the requirement is desired but not mandatory.

**May** – adverb used to indicate an option. For example, “The system may be taken offline for up to one hour every evening for maintenance.” Not used to express a requirement, but rather to specifically allow an option.

**Controls** – the individual elements of a user interface such as buttons and check boxes.

## Document Conventions

~~This section describes presentation conventions use in the document.~~

Portions of this document that are incomplete will be marked with TBD. Each TBD item will have an owner and estimated date for resolving the issue.

## Assumptions

* It is assumed that the application is intended for a single user. Access control will not be provided in the software.
* It is assumed that the application will be hosted locally on the user’s computer with no internet exposure. As such, no authentication or security measures will be included in the initial release.
* It is assumed that a quarter is a ‘fiscal’ quarter. By default, each schedule of on-call rotations will span a fiscal quarter.
* It is assumed that the user will be available to make minor adjustments to a schedule should a team member’s availability change after said schedule has been generated. In the event that such a change should require more than minor adjustments, the user will have the ability to regenerate the schedule within a time span of his/her specification.

# General Design Constraints

## Product Environment

The On-Call Assistant program exists as a part of existing employee management systems at Commerce Bank. Validation of vacation, for example, will be handled by the existing system. The application will automatically generate a schedule of on-call rotations based on team member availability maintained by the team manager. This will replace the current method of managing Excel spreadsheets to manually create schedules and track employee availability.

## User Characteristics

**User:** On-Call Assistant is a single-user program. The user has full liberty to modify schedules, manage out-of-office times for team members, manage team member information, and create schedules. The user is able to manually override rotations as necessary.

## Mandated Constraints

On-Call Assistant will be written in the C# programming language and built as a web application for deployment by the customer. The Visual Studio IDE will be used for development to facilitate maintainability and future expansion by the user.

## Potential System Evolution

By working with ASP.NET it is possible to provide user authentication and access control over the existing application. With these protocols in place, the software could be safely exposed to the internet and allow for the creation of a sub class of users who can view a read-only format of the schedules.

Additionally, the software will allow for new projects to be added and have on-call schedules generated for them. Projects can be customized to alter the length of their rotations.

# Nonfunctional Requirements

The On-Call Assistant will be manageable, scalable, maintainable, and available.

* The system will be manageable in that the user will have easy access and control over the system.
* The system will be scalable in that, as the company grows, the system will continue to perform as initially designed.
* The system will require little maintenance by the user.
* Further nonfunctional requirements are listed below in the sub sections; 3.1, 3.2….etc.

## Usability Requirements

~~It’s hard to image a software system that doesn’t have usability as one of its highest nonfunctional quality requirements. It’s not enough to just say that the system should be usable though. Usability requirements must be stated in a quantifiable and testable way.~~

~~One method of specifying usability requirements is to specify efficiency, effectiveness and satisfaction goals for specific scenarios of use (section 4) carried out by representative users (section 2.2). A simpler alternative is to design a survey to measure user satisfaction and get consensus on who will take the survey and what will be considered an acceptable aggregate score.~~

## Operational Requirements

This system is designed for middle and upper management. The system will have a low error rate and provide fast response time to the user. The system will provide the ability to quickly generate and edit rotational schedules.

## Performance Requirements

~~The main performance characteristics are speed and capacity (memory). Performance requirements are usually stated as a function of the number of concurrent users. Use this section to state the performance requirements of the system as a whole. If specific transactions have their own performance requirements state these requirements below along with the description of the feature.~~

***~~Example:~~***

~~System startup time should be less than 3 seconds. With 30 concurrent users no operation should take more than 5 seconds and 95% of the operations should take less than 2 seconds.~~

## Security Requirements

~~Access to data and features may be limited to specific users. There may also be a requirement to keep an audit trail of system use. This section describes the security requirements including the levels and what needs to be protected.~~

## Safety Requirements

~~The system may affect the safety of the larger environment. For example, there are limits on the intensity of stray electromagnetic radiation from electronic devices used in hospitals. Potential safety concerns should be investigated and documented in this section.~~

## Legal Requirements

~~Some security and safety requirements may also be legal requirements. For example, federal law protects confidentiality of medical records.~~

***~~Example:~~***

~~Student social security numbers will not be visible to other students.~~

## Other Quality Attributes

~~There are specific sections above for non-functional quality attributes such as security, performance, etc. In this section describe any other non-functional quality attributes such as portability, availability, etc.~~

## Documentation and Training

~~An important part of the total system is the documentation and training that is provided with the system. This section should describe the types and quantity of documentation and training that will be provided with the product.~~

## External Interface

~~External interfaces may be user interfaces or software interfaces.~~

### User Interface

The user interface will convey a simple, utilitarian aesthetic. The software provides a mechanism to automate a very specific task and the limited interface will reflect that. Limited does not mean cumbersome however. A flat structure for control is to be used, with no configuration nested far from the landing page which will be directly reachable in any subpage.

Schedule information is to be conveyed via a calendar to provide quick, concise data about who will be scheduled for on-call rotations and when.

### Software Interface

The scheduling algorithm will be provided via an API that can be extracted and used elsewhere if necessary. The inner workings will not be exposed to the program so this API can be replaced should the scheduling behavior need modification.

# System Features

In this section we have documented the main features of the web application. Each feature will include a description and priority rating. The description is a brief overview of functional requirement(s) the feature accomplishes. The priority level is based on three factors cost, risk and value. The developer estimates the feature’s cost and risk and the user estimates its value. If there are any functional and/or nonfunctional requirements not included within the use case or feature description there will be a section including these needed requirements.

## Feature: Out-Of-Office Specification

### Description and Priority

Since the system is expected to assist the user in scheduling time away for each employee, this feature gives the user the ability to include such dates in the system.

Cost: medium

Risk: low

Value: high

### Use Case: Future Out-Of-Office Requests

**Actors:** administrator

**Description:** The use-case begins when the administrator acquires a request for out-of-office time. The administrator will need the ability to enter the out-of-office request into the schedule.

**Basic Path:**

1. The administrator will go to the schedule page.
2. The administrator will provide starting and ending dates/times for the out-of-office request.
3. The system will provide a confirmation and update the schedule.

### Use Case: Past Out-Of-Office Instances

**Actors:** administrator

**Description:** The use-case begins as the administrator attempts to enter past out-of-office instances for a team member. The administrator should have the ability to enter past instances when a team member was out of the office.

**Basic Path:**

1. The administrator will go to the schedule page.
2. The administrator will provide starting and ending dates/times for the past out-of-office instance.
3. The system will provide a confirmation and update the schedule.

### Additional Requirements

When creating an out-of-office instance, the user should be presented with a button that generates a form to enter the relevant date, time, and team member information. In addition, the user should be able to view the changes made instantaneously.

## Feature: Team Member List Editing

### Description and Priority

This feature enables the user to add and delete from the list of team members as well as edit team member information. This feature is highly needed to facilitate change in team member roles and in the hiring process.

Cost: medium

Risk: low

Value: high

### Use Case: Adding a Team Member

**Actors:** administrator

**Description:** The use-case begins as the administrator acquires the information of a new team member. The administrator should have the ability to add said team member to the list.

**Basic Path:**

1. The administrator will go to the team member page.
2. The administrator will provide relevant information about the new team member.
3. The system will introduce the record in the list of team members.

### Use Case: Removing a Team Member

**Actors:** administrator

**Description:** The use-case begins as the administrator needs to remove a team member record from the list. The administrator should have the ability to delete an existing team member from the list.

**Basic Path:**

1. The administrator will go to the team member page.
2. The administrator will select the team member to be removed.
3. The system will remove the specified record from the list of team members.

### Use Case: Update Team Member Information

**Actors:** administrator

**Description:** The use-case begins as the administrator needs to update information for a team member, for example, the application he/she is working on. The administrator should have the ability to easily edit the team member’s information.

**Basic Path:**

1. The administrator will go to the team member page.
2. The administrator will choose the desired team member record to edit.
3. The administrator will update the team member’s information as needed.
4. The system will update the team member’s record in the list of team members.

### Additional Requirements

The user should be able to easily find the buttons that assist in managing team member information.

***4.3 Feature: On-call schedule***

**Actors:** User/Manager

**4.3.1 Description and Priority**

The “generate on-call schedule” feature within the application will automatically generate a schedule of on-call rotations. The schedule will be viewable on screen and via a read-only document that displays the team members and their assigned rotations. Each schedule will span an amount of time defined by the user that is, by default, a fiscal quarter.

Cost: medium

Risk: low

Value: high

**4.3.2 Use Case: Generate schedule**

As a user I want to automatically generate an on-call schedule that will display the team members who are on call for a certain amount of time; week, weekend, month, etc…

**4.3.3 Additional Requirements**

The on-call schedule feature should be easy to access and user friendly. It should require little effort from user and be 99.9% accurate. A read-only format of the schedule should be generated for team members.

***4.4 Feature: Schedule Editor***

**Actors:** User/Manager

**4.4.1 Description and Priority**

The feature to generate a new schedule once it has been edited will allow the user to create a new version of an already generated schedule. This feature will be helpful if the user has already created a schedule and identifies errors or has last minute changes. The feature will allow them to generate a new updated version of the calendar.

Cost: low

Risk: low

Value: high

**4.4.2 Use Case: Generate updated schedule**

As a user I want to generate a new updated schedule that will allow my employees including the person on call to view their weekly schedule obligations.

**4.4.3 Additional Requirements**

**N/A**

***4.5 Feature: Read-only version of calendar***

**Actors:** All employees

**4.5.1 Description and Priority**

The read-only feature allows the user to create a calendar-style view of the schedule embedded in a distributable read-only document.

Cost: med

Risk: low

Value: medium

**4.5.2 Use Case: Generate calendar**

As a user I want to create a read-only version of the calendar that can be read, but not edited by viewers.

**4.5.3 Additional Requirements**

**N/A**

***4.6 Feature: Total time away***

**Actors:** User/Manager

**4.6.1 Description and Priority**

This feature allows the user to get a summary of a team member’s time away from the office. This will include time away for vacation, sick days, and any other days away from the office. It will not include paid federal holidays such as Christmas and New Year’s.

Cost: low

Risk: low

Value: medium

**4.6.2 Use Case: Calculate total days off**

As I user I want to know how many days out-of-office days a team member has accrued so that I can properly assign on call duties.

**4.6.3 Additional Requirements**

***4.7 Feature: Total On-Call Rotations***

**4.7.1 Description and Priority**

This feature will calculate the total number of on-call rotations a team member has participated in.

Cost: med

Risk: low

Value: medium

**4.7.2 Use Case: Calculate and view total time on rotation**

**Actors:** User/Manager

As a user I want to know how many times a team member has been on call to ensure that rotations are scheduled fairly within the organization.

**4.7.3 Additional Requirements**

**N/A**