Software Requirement specification

Heyan Huang

November 30, 2013

Contents

Part I

A Part Heading

Chapter 1

A Main Heading

Most of this example applies to `article` and `book` classes as well as to `report` class. In `article` class, however, the default position for the title information is at the top of the first text page rather than on a separate page. Also, it is not usual to request a table of contents with `article` class.

1.1 A Subheading

The following sectioning commands are available:

part
chapter
section
subsection
subsubsection
paragraph
subparagraph

But note that—unlike the `book` and `report` classes—the `article` class does not have a "chapter" command.

Software Requirements Specification for ¡Project¿ Version 1.0 approved Prepared by ¡author¿ ¡organization¿ ¡date created¿

Table of Contents Table of Contents ii Revision History ii 1. Introductio 1 1.1 Purpose 1 1.2 Document Conventions 1 1.3 Intended Audience and Reading Suggestions 1 1.4 Product Scope 1 1.5 References 1 2. Overall Description 2 2.1 Product Perspective 2 2.2 Product Functions 2 2.3 Use Classes and Characteristics 2 2.4 Operating Environment 2 2.5 Design a Implementation Constraints 2 2.6 User Documentation 2 2.7 Assumptio and Dependencies 3 3. External Interface Requirements 3 3.1 User Interfaces 3 3.2 Hardware Interfaces 3 3.3 Software Interfaces 3 3.4 Communications Interfaces 3 4. System Features 4 4.1 System Feature 1 4.2 System Feature 2 (and so on) 4 5. Other Nonfunctional Requirement 4 5.1 Performance Requirements 4 5.2 Safety Requirements 5 5.3 Securit Requirements 5 5.4 Software Quality Attributes 5 5.5 Business Rules 5 6 Other Requirements 5 Appendix A: Glossary 5 Appendix B: Analysis Models 5 Appendix C: To Be Determined List 6

Revision History Name Date Reason For Changes Version

1. Introduction 1.1 Purpose ¡Identify the product whose software requirements are specified in this document, including the revision or release number. Describe the scope of the product that is covered by thi SRS, particularly if this SRS describes only part of the system or a singl subsystem.¿

The purpose of this document is to present a detailed description of the Web Appointment Scheduling System. It will explain the purpose and features of the system, the interfaces of the system, what the system will do, and the constraints under which it must operate and how the system will react to requests from different user classes. This document is intended for both the stakeholders and the developers of the system and will be proposed to the clients (faculties) for its approval. 1.2 Document Conventions ¡Describe any standards or typographical conventions that were followed when writing this SRS, such as fonts or highlighting that have special significance. For example, state whether priorities for higher-level requirements are assumed to be inherited by detailed requirements, or whether every requirement statement is to have its own priority.¿

This software system will be a Web Appointment Scheduling System for faculties and students in University of Idaho. This system will be designe to let faculties to post available hours to students and let students reserv preferred time slots online to meet with faculty. The software will facilita

communication between faculties, students and third party users such as secretaries. By maximizing the faculty's work efficiency and flexibility the system will meet the users needs while remaining easy to understand and use.

More specifically, this system is designed to allow faculties to set up available time length and schedule slots for appointments. On the other hand, this system is designed to allow selected students or group of students to sign up a time for an appointment. The software will allow students to change cancel and move their appointments ahead of certain amount of time. Faculties and Secretary are allowed to cancel and reschedule appointments any time. The software will facilitate communication between faculties, students, and the secretaries via E-Mail. Memo of time and meeting contents of appointments will be sent to both students and faculties once an appointment is confirmed. The system also regularly sends coming appointment summary to faculties.

1.3 Intended Audience and Reading Suggestions ¡Describe the different types of reader that the document is intended for, such as developers, project managers, marketing staff, users, testers, and documentation writers. Describe what the rest of this SRS contains and how it is organized. Suggest a sequence for reading the document, beginning with the overview sections and proceeding through the sections that are most pertinent to each reader type.¿

1.4 Product Scope ¡Provide a short description of the software being specified and its purpose, including relevant benefits, objectives, and goals. Relate the software to corporate goals or business strategies. If a separate vision and scope document is available, refer to it rather than duplicating its contents here.¿ 1.5 References ¡List any other documents or Web addresses to which this SRS refers. These may include user interface style guides, contracts, standards, system requirements specifications, use case documents, or a vision and scope document. Provide enough information so that the reader could access a copy of each reference, including title, author, version number, date, and source or location.¿ 2. Overall Description 2.1 Product Perspective

This is a new software product designed for university students to make appointments with instructors. The on campus instructors and the unviersity students are the major two group of users who will benefit from this system. In case both instructors and students may need some extra

help, department sectaries are a third class of users to help modify the weekly schedules of instructors and help rearrange schedules when it com to be the rush periods (near difficult homework deadlines, or exams are coming) and many students want to get appointed but all available perio are occupied already.

2.2 Product Functions

The main events involved in this software includes:

From instructor's side: post weekly scheduling, modify schedule, modify appointment, delete/cancel student's appointment, and add(rearrange) appointments, and receiving daily emails when there will exist appointments in that specific or coming day.

From Student's side: add(schedule) appointment, delete(cancel) appointment, receiving automatic emails once add/delete/(be modified b instructor/secretaries) any appointments

From Secetary's side: modify instructor's schedule, add/delete appointments

2.2.1 Instructor's Functions 2.2.1.1 Post Schedule 2.2.1.2 Modify Schedul 2.2.1.3 Modify Appointments 2.2.1.3 Add Appointments 2.2.1.4 Delete Appointments 2.2.1.5 Receive Email

2.2.2 Student's Functions 2.2.2.1 Add Appointments 2.2.2.2 Delete Appointments 2.2.2.3 Receive Email

2.2.3 Secetery's Functions 2.2.3.1 Modify Schedule 2.2.3.2 Add Appointmenst 2.2.3.3 Delete Appointments

2.3 User Classes and Characteristics

According to main functions from 2.2, there will be three main user groups, and according to group's operation and software interface, each user group function should at least have one sperate Java class definition or class member functions so that the software implementation will inclu at least three major classes, and each class would need to define all possible/potential member functions which makes all three user groups' operations logical and executable.

Followed are the user classes, member functions and possible/potential arguments:

2.2.1 Instructor's Functions: Class Instructor 2.2.1.1 Post Schedule: start date, end date, 2.2.1.2 Modify Schedule: date, schetype, 2.2.1.3 Modify Appointments: date, appflag, applenpre, starttime 2.2.1.3 Add Appointments: date, starttime, applenpre, appflag 2.2.1.4 Delete Appointments: date, starttime 2.2.1.5 Receive Email: date, time, dailyflag

2.2.2 Student's Functions: Class Student 2.2.2.1 Add Appointments: date, starttime, applenpre, msg 2.2.2.2 Delete Appointments: date, starttime, applenpre, msg 2.2.2.3 Receive Email: date, starttime, applenpre, msg(reasons)

2.2.3 Secetery's Functions: Class Secretary 2.2.3.1 Modify Schedule: date, schetype, 2.2.3.2 Add Appointmenst: date, starttime, applenpre, msg 2.2.3.3 Delete Appointments: date, starttime, applenpre, msg

2.4 Operating Environment

This software will be Window's, Linux and potentially mac cooperate. The software requirements would be internet explore, or firefox, and internet surfing software needed. For access to the system, valid university/campus email address and password are required for accessing the software system.

2.5 Design and Implementation Constraints

Right now, we assume/suppose we would get university account system safely accessible for our software. This is only the initial ideas without feasibility consideration and research yet, so we are not sure yet we can completely achieve this goal. so far no server accessibility consideration yet, we are not sure if we have available resource for this software to be implemented to use yet. we assume we are going to use university student email account or employee email account as the user check. But we have not consider any potential security issues yet. there may exist some restrictions from the security point of view.

2.6 User Documentation

There are certain rules for all the user groups. and there will be a simple user's guide/manual for each group displayed right behind the scheduling 2-dimensional table, which will least from the most important

concerns/issues to least significant ones, so that every user within any group would easily catch the essentials and succcessfully satisfy their requirements.

2.7 Assumptions and Dependencies

Right now, we assume/suppose we would get university account system safely accessible for our software. This is only the initial ideas without feasibility consideration and research yet, so we are not sure yet we can completely achieve this goal.

By designing and reviewing the software specifications so far, we assume the project goal is clear, the implementation language is feasible to achie the software design goals, and we would be able to fair easily complete th goal.

But from the software application in everyday life, whether the software can be widely used by instructors or students will still depends on the logging systems, software sever availability and user performs. So far, we assume all users are normal, regular user for appointment and studying, solving problem propose, but potential malicous operations may be a threat for the software application. 3. External Interface Requirements 3 User Interfaces ¡Describe the logical characteristics of each interface between the software product and the users. This may include sample screen images, any GUI standards or product family style guides that are to be followed, screen layout constraints, standard buttons and functions (e.g., help) that will appear on every screen, keyboard shortcuts, error message display standards, and so on. Define the software components fo which a user interface is needed. Details of the user interface design shou be documented in a separate user interface specification.¿

Student Interface: Week schedule/ Daily schedule with the following buttons: Select, Confirm, Cancel, Memo.

Faculty Interface: Week schedule/ Daily schedule with the following buttons: Select Starting time, End time; Select length of time slots; Confirm and publish schedule, Cancel, Block, Memo.

Secretary Interface: Week schedule/ Daily schedule with the following buttons: Select Starting time, End time; Select length of time slots; Confirm and publish schedule, Cancel, Block, Memo.

3.2 Hardware Interfaces ¡Describe the logical and physical characteristics of each interface between the software product and the hardware components of the system. This may include the supported device types, the nature of the data and control interactions between the software and the hardware, and communication protocols to be used.¿ 3.3 Software Interfaces ¡Describe the connections between this product and other specific software components (name and version), including databases, operating systems, tools, libraries, and integrated commercial components. Identify the data items or messages coming into the system and going out and describe the purpose of each. Describe the services needed and the nature of communications. Refer to documents that describe detailed application programming interface protocols. Identify data that will be shared across software components. If the data sharing mechanism must be implemented in a specific way (for example, use of a global data area in a multitasking operating system), specify this as an implementation constraint.¿ 3.4 Communications Interfaces ¡Describe the requirements associated with any communications functions required by this product, including e-mail, web browser, network server communications protocols, electronic forms, and so on. Define any pertinent message formatting. Identify any communication standards that will be used, such as FTP or HTTP. Specify any communication security or encryption issues, data transfer rates, and synchronization mechanisms.¿

The software is associated with vandal-mail, web browser, vandalweb network server. 4. System Features ¡This template illustrates organizing the functional requirements for the product by system features, the major services provided by the product. You may prefer to organize this section by use case, mode of operation, user class, object class, functional hierarchy, or combinations of these, whatever makes the most logical sense for your product.¿ 4.1 System Feature 1 ¡Don't really say "System Feature 1." State the feature name in just a few words.¿ 4.1.1 Description and Priority ¡Provide a short description of the feature and indicate whether it is of High, Medium, or Low priority. You could also include specific priority component ratings, such as benefit, penalty, cost, and risk (each rated on a relative scale from a low of 1 to a high of 9).¿ 4.1.2 Stimulus/Response Sequences ¡List the sequences of user actions and system responses that stimulate the behavior defined for this feature. These will correspond to the dialog elements associated with use cases.¿ 4.1.3 Functional Requirements ¡Itemize the detailed functional requirements associated with this feature. These are the software capabilities that must be present in

order for the user to carry out the services provided by the feature, or to execute the use case. Include how the product should respond to anticipated error conditions or invalid inputs. Requirements should be concise, complete, unambiguous, verifiable, and necessary. Use "TBD" as placeholder to indicate when necessary information is not yet available.¿

¡Each requirement should be uniquely identified with a sequence number a meaningful tag of some kind.¿

REQ-1: REQ-2: 4.2 System Feature 2 (and so on) 5. Other Nonfunction Requirements 5.1 Performance Requirements ¡If there are performance requirements for the product under various circumstances, state them he and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for rea time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.¿

Right now, we assume/suppose we would get university account system safely accessible for our software. This is only the initial ideas without feasibility consideration and research yet, so we are not sure yet we can completely achieve this goal.

5.2 Safety Requirements ¡Specify those requirements that are concerned with possible loss, damage, or harm that could result from the use of the product. Define any safeguards or actions that must be taken, as well as actions that must be prevented. Refer to any external policies or regulations that state safety issues that affect the product's design or use Define any safety certifications that must be satisfied.¿

Having not considering if we use student email account, if there will be a safety issues with it. 5.3 Security Requirements ¡Specify any requirement regarding security or privacy issues surrounding use of the product or protection of the data used or created by the product. Define any user identity authentication requirements. Refer to any external policies or regulations containing security issues that affect the product. Define any security or privacy certifications that must be satisfied.¿

But from the software application in everyday life, whether the software can be widely used by instructors or students will still depends on the logging systems, software sever availability and user performs. So far, we

assume all users are normal, regular user for appointment and studying, solving problem propose, but potential malicous operations may be a threat for the software application.

5.4 Software Quality Attributes ¡Specify any additional quality characteristics for the product that will be important to either the customers or the developers. Some to consider are: adaptability, availability, correctness, flexibility, interoperability, maintainability, portability, reliability, reusability, robustness, testability, and usability. Write these to be specific, quantitative, and verifiable when possible. At the least, clarify the relative preferences for various attributes, such as ease of use over ease of learning.¿

Will send of servay to all groups of users to get quality evaluation from users.

5.5 Business Rules ¡List any operating principles about the product, such as which individuals or roles can perform which functions under specific circumstances. These are not functional requirements in themselves, but they may imply certain functional requirements to enforce the rules.¿

Initiative creation. No copying. 6. Other Requirements ¡Define any other requirements not covered elsewhere in the SRS. This might include database requirements, internationalization requirements, legal requirements, reuse objectives for the project, and so on. Add any new sections that are pertinent to the project.¿ Appendix A: Glossary ¡Define all the terms necessary to properly interpret the SRS, including acronyms and abbreviations. You may wish to build a separate glossary that spans multiple projects or the entire organization, and just include terms specific to a single project in each SRS.¿ Appendix B: Analysis Models ¡Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.¿ Appendix C: To Be Determined List ¡Collect a numbered list of the TBD (to be determined) references that remain in the SRS so they can be tracked to closure.¿