SYSTEMS AND SOFTWARE REQUIREMENTS SPECIFICATION (SSRS) FOR

sQuire Collaborative IDE

Picture here.

Version 1.0
February 09, 2016

Prepared for:
CS383-01

Prepared by:
Domn Werner (wern0096) | Robert Carlson (carl7595) | Brian Cartwright (cart1189)
Max Welch (welc2150) | Matthew Daniel (dani2918) | Brandon Ratcli (ratc8795)
Joel Doumit (doum6708) | Eric Gentile-Quant (gent7104)
Team 4 - It Compiled Yesterday, I Swear
University of Idaho
Moscow, ID 83844-1010

sQuire SSRS

RECORD OF CHANGES

Change number	Date com-pleted	Location of change (e.g., page or gure #)	A M D	Brief description of change	Approved by (initials)	Date Approved

*A - ADDED M - MODIFIED D - DELETED

SQUIRE SSRS
TABLE OF CONTENTS

Section Page

1 Introduction

The sQuire Collaborative IDE is a collaborative IDE software project for CS383-01. The intended audience for this project is Java programmers looking for a more social collaborative experience. A large focus of the program is also to help programmers connect with others who may interested in their projects.

1.1 IDENTIFICATION

The software system being considered for development is referred to as sQuire. The customer providing speci cations for the system is Dr. Je ery and the CS383-01 class. The ultimate customer, or end-user, of the system will be Java programmers. This is a new project e ort, so the version under development is version 1.0.

1.2 PURPOSE

The purpose of the system under development is to provide Java programmers with a more social collaborative experience. Instead of individual methods of source control, sQuire will provide an environment where programmers can work together in the same environment and instantly see the e ect of others' code. While the system will be used by Java programmers, this document is intended to be read and understood by UI CS software designers and coders. The document will also be vetted or approved by Team 4.

1.3 SCOPE

This paragraph shall brie y summarize the history of system development, operation, and maintenance; identify the project sponsor, acquirer, user, developer, and support agencies; identify current and planned operating sites; and list other relevant documents.

[insert your text here]

1.4 DEFINITIONS, ACRONYMS, AND ABBREVIATIONS

This section shall list and de ne all special terms, acronyms and abbreviations used throughout this document. A tabular form is preferable, but not mandatory.

Term or Acronym	De nition
Alpha test	Limited release(s) to selected, outside testers
Beta test	Limited release(s) to cooperating customers wanting early access to developing systems
Final test	aka, Acceptance test, release of full functionality to customer for approval
DFD	Data Flow Diagram
SDD	Software Design Document, aka SDS, Software Design Speci cation
SRS	Software Requirements Speci cation
SSRS	System and Software Requirements Speci cation
IDE	Integrated Development Environment

1.5 REFERENCES

This section shall list full bibliographic citations of all documents referenced in this report. This section shall also identify the source for all materials not available in printed form (e.g., web-based information) and list the complete URL along with owner, author, posting date, and date last visited.

[insert your citations here]

1.6 OVERVIEW AND RESTRICTIONS

This paragraph shall describe the organization of this document and shall describe any security or privacy considerations associated with its use.

This document is for limited release only to UI CS personnel working on the project and [state others who will receive the document].

Section 2 of this document describes the system under development from a holistic point of view. Functions, characteristics, constraints, assumptions, dependencies, and overall requirements are de ned from the system-level perspective.

Section 3 of this document describes the speci c requirements of the system being developed. Interfaces, features, and speci c requirements are enumerated and described to a degree su cient for a knowledgeable designer or coder to begin crafting an architectural solution to the proposed system.

Section 4 provides the requirements traceability information for the project. Each feature of the system is indexed by the SSRS requirement number and linked to its SDD and test references.

Sections 5 and up are appendices including original information and communications used to create this document.

2 OVERALL DESCRIPTION

The sQuire project is an answer to the lack of real-time collaborative programming experiences. By bringing programmers together in a more social environment, this program aims to improve collaboration between programmers in a much more fast paced and agile methodology. Furthermore, for programmers who seek others to help with their projects, sQuire aims to provide a simple social platform for engaging with other programmers and start working on a project together.

2.1 PRODUCT PERSPECTIVE

This program will either be a standalone executable with many linked DLLs or a web application that can run on most modern web-browsers with internet access.

2.2 PRODUCT FUNCTIONS

This subsection of the document should provide a summary of the major functions that the software will perform. For the sake of clarity, the functions should be organized in a way that makes the list of functions understandable to the customer or to anyone else reading the document for the rst time. Textual or graphical methods can be used to show the di erent functions and their relationships. Such a diagram is not intended to show a design of a product, but simply shows the logical relationships among variables.

[insert your text here]

2.3 USER CHARACTERISTICS

This subsection of the document should describe those general characteristics of the intended users of the product including educational level, experience, and technical expertise. It should not be used to state speci c requirements, but rather should provide the reasons why certain speci c requirements are later speci ed in Section 3 of this document.

[insert your text here]

2.4 CONSTRAINTS

This subsection of the document should provide a general description of any other items that will limit the developer's options. These include: a) Regulatory policies; b) Hardware limitations (e.g., signal timing requirements); c) Interfaces to other applications; d) Parallel operation; e) Audit functions; f) Control functions; g) Higher-order language requirements; h) Signal handshake protocols; i) Reliability requirements; j) Criticality of the application; k) Safety and security considerations.

[insert your text here]

2.5 ASSUMPTIONS AND DEPENDENCIES

This subsection of the document should list each of the factors that a ect the requirements stated in the document. These factors are not design constraints on the system and/or software but are, rather, any changes to them that can a ect the requirements in the document. For example, an assumption may be that a speci c operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not available, the document would then have to change accordingly.

[insert your text here]

2.6 SYSTEM LEVEL (NON-FUNCTIONAL) REQUIREMENTS

This subsection of the document should identify system level (whole, not functional) requirements that impact the construction, operation, packaging and delivery of the system and software.

2.6.1 Site dependencies

This paragraph shall specify site-dependent operational parameters and needs (such as parameters indicating operation-dependent targeting constants or data recording). The requirements shall include, as applicable, number of each type of equipment, type, size, capacity, and other required characteristics of processors, memory, input/output devices, auxiliary storage, communications/ network equipment, and other required equipment or software that must be used by, or incorporated into, the system. Examples include operating systems, database management systems, communications/network software, utility software, input and equipment simulators, test software, and manufacturing software. The correct nomenclature, version, and documentation references of each such device or software item shall be provided.

1. Central SQL Server
2. Host-side Project Server
3. Collaborator-side Client

The Central server stores user credentials, project descriptions, and user pro le and achievement data.
Requirements for Central SQL Server

1. Host with high uptime percentage
2. SQL capable
3. E-mail capable for password resets
4. Fast enough connection to prevent login timeout, even while handling multiple requests
5. Prefer host with multiple backups

The Host-side server stores the project les, project access list, hosts the editing environment, runs chat channels, and serves les to collaborators for compiling.
Requirements for Host-side Server

1. Java Capable (http://java.com/en/download/help/sysreq.xml)
2. SQL Capable (WAMP/LAMP)
3. 4 GB RAM
4. Hard drive space for server + project les

The client side application connects to the host server, renders GUI elements, stores connection pro les, stores server les, and compiles the project.
Requirements for Collaborator-side Client

1. Java Capable (http://java.com/en/download/help/sysreq.xml)
2. Project speci ed Java installed
3. 2 GB RAM
4. Hard drive space for project les

2.6.2 Safety, security and privacy requirements

This paragraph shall specify the system requirements, if any, concerned with maintaining safety, security and privacy. These requirements shall include, as applicable, the safety, security and privacy environment in which the system must operate, the type and degree of security or privacy to be provided, and the criteria that must be met for safety/security/privacy certi cation and/or accreditation.

The collaborative nature of sQuire includes several concerns for security and privacy. The program will include in the license agreement the following stipulations:

1. sQuire is a free development environment, and may be used for commercial purposes
2. No guarantee of code con dentiality is implied by use of sQuire
3. Clients assume the risk of downloading, compiling, and running project les
4. Email addresses are visible as part of a user pro le
5. Host assume the risk of allowing peers to connect to their server

However, the program will provide the following minimum features to address security and privacy concerns:

1. All SQL servers will include input sanitization and appropriate anti-injection safeguards
2. Project hosts may turn o guest access to their project
3. Uploads for assets will be limited to folders within the project directory
4. Visibility to host le structure will be limited to project folders only

2.6.3 Performance requirements

This paragraph should specify both the static and the dynamic numerical performance requirements placed on the software or on human interaction as a whole. Static numerical requirements may include the following: a) The number of terminals to be supported; b) The number of simultaneous users to be supported; c) Amount and type of information to be handled. Dynamic numerical requirements may include, for example, the numbers of transactions and tasks and the amount of data to be processed within certain time periods for both normal and peak workload conditions. All of these requirements should be stated in measurable terms. For example, 95% of the transactions shall be processed in less than 1msec.

1. Up to 33 concurrent connections will be supported
2. Edits will be visible to all connected collaborators within 10 seconds
3. Login and server connections will report success or failure within 45 seconds

2.6.4 System and software quality

This paragraph shall specify the requirements, if any, concerned with hardware and software quality factors identi ed in the contract. Examples include quantitative requirements regarding the system's functionality (the ability to perform all required functions), reliability (the ability to perform with correct, consistent results), maintainability (the ability to be easily corrected), availability (the ability to be accessed and operated when needed), exibility (the ability to be easily adapted to changing requirements), portability (the ability to be easily modi ed for a new environment), reusability (the ability to be used in multiple applications), testability (the ability to be easily and thoroughly tested), usability (the ability to be easily learned and used), and other attributes.

Adaptability

1. The program will allow selection of di erent compiling programs and command line arguments.
2. The program will allow importing of les of key words to allow other development languages to be used.

2.6.5 Packaging and delivery requirements

This paragraph shall specify the requirements, if any, for packaging, labeling, handling and delivery of the system being developed to the customer.

The executable system and all associated documentation (i.e., SSRS, SDD, code listing, test plan (data and results), and user manual) will be delivered to the customer on CD's and/or via email, as speci ed by the customer at time of delivery. Although document drops will occur throughout the system development process, the nal, edited version of the above documents will accompany the nal, accepted version of the executable system.

2.6.6 Personnel-related requirements

This paragraph shall specify the system requirements, if any, included to accommodate the number, skill levels, duty cycles, training needs, or other information about the personnel who will use or support the system under development. These requirements shall include, as applicable, considerations for the capabilities and limitations of humans; foreseeable human errors under both normal and extreme conditions; and speci c areas where the e ects of human error would be particularly serious. Examples include requirements for color and duration of error messages, physical placement of critical indicators or keys, and use of auditory signals.

The system under development has no special personnel-related characteristics.

2.6.7 Training-related requirements

This paragraph shall specify the system requirements, if any, pertaining to training. Examples include training software, tutorials, or help information to be included in the system.

No training materials or expectations are tied to this project other than the limited help screens built into the software and the accompanying user manual.

2.6.8 Logistics-related requirements

This paragraph shall specify the system requirements, if any, concerned with logistics considerations. These considerations may include: system maintenance, software support, system transportation modes, supply-system requirements, impact on existing facilities, and impact on existing equipment.

[Insert a description of the minimum hardware requirements and OS and application software dependencies here]

2.6.9 Other requirements

This paragraph shall specify additional system level requirements, if any, not covered in the previous paragraphs.

[insert your text here]

2.6.10 Precedence and criticality of requirements

This paragraph shall specify, if applicable, the order of precedence, criticality, or assigned weights indicating the relative importance of the requirements in this speci cation. Examples include identifying those

requirements deemed critical to safety, to security, or to privacy for purposes of singling them out for special treatment. If all requirements have equal weight, this paragraph shall so state.

[insert your text here]

3 SPECIFIC REQUIREMENTS

3.1 Functional Requirements

1. The sQuire software shall support editing, compilation, and execution of Java programs in the shared project area or in their local system.
2. Java programs shall be composed as projects and support multiple source code les across multiple directories within a shared, top-level directory.
3. Users shall be able to import les to the shared project area.
4. Users shall be able to export les from the shared project area to their local le system.
5. The shared project area shall support up to 32 users in a shared IDE session.
6. The project owner shall host the shared IDE service.
7. The sQuire software shall be able to transfer hosting to other online users.
8. Users shall perform compilation individually.
9. Users shall execute compiled programs individually.
10. The sQuire editor shall provide syntax coloring for the Java language.
11. The sQuire editor shall provide a visual indication to see the author(s)and date(s) of code.
12. Users shall be able to move around individually in the shared project space.
13. Users shall be able to quickly jump to where other users are looking.
14. Users shall be able to text chat with other users.
15. Users shall be able to edit their pro le and customize various settings.
16. The sQuire software shall provide users with a project browser that displays information about all proposed and currently being worked on projects.
17. Users shall be able to vote, comment, and request to join projects.

3.2 EXTERNAL INTERFACE REQUIREMENTS

This subsection should be a detailed description of all inputs into and outputs from the software system. It should complement the constraints and dependencies de ned in earlier sections, but not repeat that information. Hardware, software, user, and other communication interfaces need to be speci ed. Use the four subsections listed below or the table on the next page, or some combination of both.

3.2.1 Hardware Interfaces

[insert your text here]

3.2.2 Software Interfaces

[insert your text here]

3.2.3 User Interfaces

[insert your text here]

3.2.4 Other Communication Interfaces

[insert your text here]

External Interface Requirements

Hardware Interfaces

Name	Source/Destination	Description	Type/range	Dependencies

Software Interfaces

Name	Source/Destination	Description	Type/range	Dependencies

User Interfaces

Name	Source/Destination	Description	Type/range	Dependencies

Other Communication Interfaces

Name	Source/Destination	Description	Type/range	Dependencies

3.3 SYSTEM FEATURES

Functional requirements should de ne the fundamental actions (i.e., features) that must take place in the software in accepting and processing the inputs and in processing and generating the outputs. These requirements are given in the form ofUse Cases where possible, denoting a concrete use (discrete user-performable task) of the system. Use case diagrams are followed by use case descriptions, followed by any non-task features. Non-task features are generally listed as shall statements starting with The system shall... These include: a) Validity checks on the inputs; b) Exact sequence of operations; c) Responses to abnormal situations, including error detection, handling and recovery; d) Parameter speci cation and usage; e) Relationship of outputs to inputs, including formulas for input to output conversion.

It may be appropriate to partition the functional requirements into sub functions or subprocesses, but that decomposition (here) does not imply that the software design will also be partitioned that way. You should repeat subsections 3.3.i for every speci ed feature de ned for the system or software.

3.3.1 Use Case Diagrams

[insert 1+ use case diagrams here]

3.3.2 System feature 1: [insert feature name here]

For each feature, you should either provide a Use Case Description or a Non-task feature description, whichever is more appropriate.

Use Case Description	Non-task feature description
Name Actors Goals Preconditions Summary Related use cases Steps 1. ... 2. ... Alternatives Postconditions	Introduction/Purpose [insert your text here] Input/Output sequence [insert your text here] Design constraints [insert your text here] Performance requirements [insert your text here] Detailed functional requirements Functional requirement 1.1 [insert your text here] Functional requirement 1.2 [insert your text here] . . . Functional requirement 1.[n] [insert your text here]

3.3.3 System feature 2: [insert feature name here]

Write another feature description, as either a use case or a non-task feature description

. . . do as many as necessary, probably a lot

3.3.4 System feature [m]: [insert feature name here]

Write a nal feature description, as either a use case or a non-task feature description

4 REQUIREMENTS TRACEABILITY

This section shall contain traceability information from each system requirement in this specification to the system (or subsystem, if applicable) requirements it addresses. A tabular form is preferred, but not mandatory.

Feature Name	Req No.	Requirement Description	Priority	SDD	Alpha Release		Beta Release	
					Test Case(s)	Test Res.	Test Case(s)	Test Res.
	1.1							
	1.2							
	. . .							
	1.[n]							
	2.1							
	2.2							
	. . .							
	2.[n]							
	3.1							
	3.2							
	. . .							
	3.[n]							
. . .	. . .							
	[m].1							
	[m].2							
	. . .							
	[m.n]							

Priorities are: Mandatory, Low, High

SDD link is version and page number or function name.

Test cases and results are file names and Pass/Fail or % passing.

5 APPENDIX A. [insert name here]

Include copies of speci cations, mockups, prototypes, etc. supplied or derived from the customer. Appendices are labeled A, B, . . . n. Reference each appendix as appropriate in the text of the document.

[insert appendix A here]

6 APPENDIX B. [insert name here]

[insert appendix B here]