PDF Active-Reading Assistant
Software Requirements Specification

A. Hornof (ajh) - 4-4-2024 - v1.02

There is one correction in red on page 4 under "Login 8.".
A CS systems website with mysql instructions was added near the top of page 6.

Table of Contents

1. SRS Revision History

Date	Author	Description
4-4-2024	A.H.	Created the initial document using the template from https://classes.cs.uoregon.edu/24S/cs422/Templates.html

2. The Concept of Operations (ConOps)

This document is the Software Requirements Specification (SRS) for the Active-Reading Assistant (ARA). ARA is a computer program that is designed to support the SQ3R reading method for PDF documents.

ARA is a software tool that guides students through the use of the SQ3R active reading technique. SQ3R stands for Survey, Question, Read, Recite, Review. SQ3R provides students (and other readers) with a structured approach for reading textbooks and other technical material. Learning such material is best accomplished by first skimming the high-level structure of a

chapter (or other section of text), and then generating questions, and then reading the material primarily to answer the initial questions (McKeachie & Svinicki, 2013).

The goal of SQ3R is to motivate students to engage in *active learning* by (a) generating questions, (b) moving through the text with the goal of answering those questions, and then (c) quizzing yourself on what you learned. The goal is to encourage and support "deep processing" of the text (McKeachie & Svinicki, 2013, p.30).

In the first pass through a chapter, the *survey*, the student should skip over a lot of text, such as to only (a) read the headings and topic sentences and (b) study the figures and diagrams. It is expected that students will eventually read the entire chapter, but only after first constructing a mental understanding of the structure of the chapter, and how its major pieces fit together (ibid.).

The basic concept of the ARA system is that it will provide a structured, hierarchical, note-taking facility that guides students through the use of SQ3R active reading technique, by providing (a) appropriate prompts for the next step in following the SQ3R, (b) a structured organization of text fields in which the user can type their notes, and (c) an easy means of quizzing yourself on what you read.

ARA should store the user's notes on a server that can be accessed from anywhere with full internet access. This permits the user to use multiple different computers to use the SQ3R method to read a textbook, and always add to their previous notes rather than starting fresh.

ARA should provide, on the server, sample PDFs of textbook chapters, with all of the important *survey* content highlighted, such as headings, topic sentences, and diagrams. This should help new SQ3R users to understand how much of the chapter they should be reading during the *survey* phase.

2.1. Current System or Situation

There are substantial digital-document paper-based resources available to assist students in the use of SQ3R. For example: Virginia Tech (2024), McKeachie & Svinicki (2013), Robinson (1946), and University of Oregon (2024). A brief internet search suggests that there are few if any pieces of software created specifically for helping students learn and use SQ3R.

2.2. Operational Features of the Proposed System

The key operational features of ARA include: (a) guiding the user through the use of SQ3R, (b) the display of "starter" PDF files that visually highlight what should be looked-at during the *survey* phase, (c) a means to capture hierarchical notes in which the user can do SQ3R tasks (such as writing questions and answers), (d) supporting self-quiz, and (e) storing all of the data on a server.

The system will specifically follow the SQ3R technique that is described by many university academic academic guidance centers, including here at the University of Oregon (2024), such as at https://engage.uoregon.edu/learning-resources/.

2.3. User Classes

There are two user classes:

1. A "student" who is attempting to use SQ3R, with the assistance of ARA, to engage in active learning while reading. The "student" could be anyone, not necessarily someone in school, but this SRS will refer to this user as the student.

2. A system administrator who sets up the server. The administrator's duties will be to set up the MySQL (or other) server that the ARA system will use. This would be a moderately-skilled computer user familiar with installing software, editing text files, and using the Unix command prompt. The system should provide instructions for this user on how to set up and test the server.

2.4. Modes of Operation

The system has one primary mode of operation, in which the server is running, and a student runs ARA which gain client access to the server.

2.5. Operational Scenarios (Also Known as "Use Cases")

(These will be developed.)

3. Specific Requirements

The basic functionality must include the following:

Externally-Visible User Interactions

1. A visual field for taking notes.
2. Automatic storage on the server.
3. The system should support a hierarchy of notes such as:
    Chapter Title (single line, entered by user)
    Section Heading (single line of text, entered by user)
    Notes for that section (scrolling text, with some kind of separation between lines.)
4. The system should support the hiding of notes so that users can quiz themselves on what they wrote down. This could be accomplished, for example, with a single mouse-click (such as, on a button) to hide the notes, and a single mouse-click to unhide notes.

5. Pre-loaded PDFs that have all of the important *survey* content highlighted, such as headings, topic sentences, diagrams. (At least three chapters from Sommerville should be included.)
6. Pre-loaded complete notes from one entire Sommerville chapter, to provide sample data to see how the system can work.

SQ3R Assistance ("Scaffolding")

7. The program should guide the user through the use of SQ3R with prompts such as: "SURVEY: Glance over the headings in the chapter to see the few big points."
    4a. "Prompts" should guide the user through the use of SQ3R. Prompts should be non-modal: They should not block any text the user types, should not require the user to read them or provide any input, and should not be interactive. For example, the prompts could appear as headings above the text fields.
    4b. The user should be able to turn the prompts on or off, making the prompts either present or absent.

Login

8. There should be *NO* need to set up a username and password for this initial version of the software. Any login information should be built in to the software.
   [Correction made on 4-16-2024.]

Target Platform

9. The software should run on a laptop or desktop machine. The program should be designed for use with a real keyboard, not a smartphone. The system should run on macOS 12.

Data Storage

10. Data should be stored on a server using either mysql or mongo.
11. The initial version of the system can have the server and ARA client running on the same machine. The ideal final version of the system should permit the administrator to set up the server on a remote machine, and have ARA access that server using the Internet.
12. The system should save all notes that the user enters, and should never delete user data without a warning.
13. The system should provide a warning, at startup, if the server connection cannot be established, and data will not get saved. (The ideal system would save the changes locally until the server connection could be established, and then save the changes. But this will not be a requirement with this system.)

Some Aspects of the System Can be "Mocked Up".

The system should initially have three predefined users, with no passwords necessary. It should be very easy to select which user you are logging on as, such as with a single click, or selecting usernames from a list. (Yes, this would of course create security problems if the system were to be deployed on the Internet, but these seems like a "solved problem" that would be easy to implement at a later time.)

Build-Related Constraints

14. System Document File Formats
All system-related and system-development-related documents that are intended for human reading must be in either plain text or PDF. For example, Microsoft Word, Microsoft Excel, or markdown language documents must be converted into plain text or PDF.

15. Programming Constraints

Any of the following languages may be used:

- C/C++, the C++ standard library, Cocoa, and no other components. C++ code must comply with C++11. (Note that an XCode command line tool could fulfill many of the requirements.)

- Python 3 along with the Python Standard Library https://docs.python.org/3/library/index.html, but no other imports except for *mysql* or *pymongo* unless written permission is requested and granted. (Please send brief email requests explaining what libraries you would like to use, provide a link to the library website, and explain why you would like to use them.)
- The GUI framework tkinter is part of the Python Standard Library.
- Python code must run in 3.12.

- Java along with Java Standard Edition modules, but no other imports, in Java 17, 21, or 22.

- Other languages with instructor permission. (Please send brief email requests explaining what language you would like to use, and explain why you would like to use it.)

16. Installation Constraints

- Instructions must be provided for how to compile/run the code and install the system.l
- There can be at most 20 commands required to set up the server, compile the code, and run the program.
- An experienced computer programmer should not require more than 20 minutes working alone with the submitted materials to set up the server, and compile and run the code.

- All code, except for approved library imports, should be submitted. Please do not direct the person installing the software to a location to download the code.
- No server connections may be required for either installing or running the software, except for setting up mysql on ix (or ix-dev), and you must provide instructions on how to set this up. The instructions should be as simple as the MongoDB and MySQL guidance at https://systems.cs.uoregon.edu/wiki/index.php?n=Help.Tools (along with whatever files are needed to create your tables and such). [Added on 4-19-2024.]
- No virtual environments may be used.
- No gaming engines such as Unity may be used.

4. References

McKeachie, W., & Svinicki, M. (2013). McKeachie's teaching tips. Cengage Learning.

Robinson, F. P. (1946). Effective study.

University of Oregon Tutoring and Academic Engagement Center. (2024). Reading Strategy: SQ3R. Accessed from this website: https://engage.uoregon.edu/learning-resources/. Direct link: https://engage.uoregon.edu/files/2014/09/SQ3R_Worksheet.pdf. Accessed 4-4-2024.

Virginia Tech Cook Counseling Center. (2024). SQ3R: Reading/Study System. https://ucc.vt.edu/academic_support/study_skills_information/sq3r_reading-study_system.html. Accessed 4-4-2024.

Michigan State University. (2022). Reading a Textbook Effectively. https://natsci.msu.edu/students/current-students/student-success-resources/academic-success/habits-to-develop-outside-of-class/study-strategies/reading-a-textbook-effectively/. Accessed 4-4-2024.

5. Acknowledgements

This SRS builds on the template from https://classes.cs.uoregon.edu/24S/cs422/Templates.html.