Software Requirements Specification

for

Fraction Worksheet Creator

**Version 1.0 approved?**

**Prepared by**

**Olga Sheehan, Eric Holm, Sara Hakkoum, Ethan Fiero**

**Elementary Engineers**

**2/9/2016**

**Table of Contents**

**Table of Contents**

**Revision History**

**1. Introduction**

1.1 Purpose

1.2 Product Scope

1.3 Intended Audience and Reading Suggestions

1.4 References

1.5 Overview

**2. Overall Description**

2.1 Product Perspective

2.2 Product Functions

2.3 User Classes and Characteristics

2.4 Operating Environment

2.5 Design and Implementation Constraints

2.6 User Documentation

2.7 Assumptions and Dependencies

**3. User Requirements**

3.1 Functional Requirements

3.2 Non-Functional Requirements

3.3 Software Interfaces

3.4 Communications Interfaces

**4. System Requirements**

4.1 System Feature 1

4.2 System Feature 2 (and so on)

**5. Other Nonfunctional Requirements**

5.1 Performance Requirements

5.2 Safety Requirements

5.3 Security Requirements

5.4 Software Quality Attributes

5.5 Business Rules

**6. Other Requirements**

**Appendix A: Glossary**

**Appendix B: Analysis Models**

**Appendix C: To Be Determined List**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Name** | **Date** | **Reason For Changes** | **Version** |
| Initial Draft | 02/09/2016 | Initial Draft | 0.0.1 |
|  |  |  |  |

# Introduction

This section gives a scope description and overview of everything included in this SRS document. Also, the purpose for this document is described and a list of abbreviations and definitions is provided.

## Purpose

The purpose of this document is to give a detailed description of the requirements for the “Fraction Worksheet Creator” (FWC) software. It will illustrate the purpose and complete declaration for the development of system. It will also explain system constraints, interface and interactions with other external applications. This document is primarily intended to be proposed to a customer for its approval and a reference for developing the first version of the system for the development team.

## Product Scope

The “Fraction Worksheet Creator” (FWC) is an off-line worksheet generation tool designed by Elementary Engineers to help elementary school teachers to create a lot of exercises for students to study and practice fractions.The fractions worksheets are randomly created and never repeated so the teachers have an endless supply of quality fractions worksheets to use in the classroom or at home. The generated worksheets can contain fraction problems of various difficulty levels, from basic addition and subtraction problems with visuals and images suitable for small children, to quite advanced fraction equations. The worksheets created with “Fraction Worksheet Creator” are not pre-designed but are randomly generated based on a complex set of algorithms corresponding to the specific mathematical structure of each type of worksheet. This allows for virtually unlimited original worksheets. Furthermore, FWC will also generate an answer sheet. The worksheets will be created as PDF documents which are automatically opened on the computer once the generating process is done.

The “Fraction Worksheet Creator” worksheets are free to print, easy to use, and very flexible. This project is being produced by students at Framingham State University with help from Dr. Andrew Jung.

## Intended Audience and Reading Suggestions

This document is to be read by the customer, development team, the project managers, testers, documentation writers and end users. The SRS has been organized approximately in order of increasing specificity. The developers and project managers need to become intimately familiar with the SRS.

Others involved need to review the document as such:

*Overall Description* – The customer and end users have to become accustomed to the various product features in order to effectively advertise the product.

*System features* – Testers need an understanding of the system features to develop meaningful test cases and give useful feedback to the developers.

*External Interface Requirements* – The software developers need to know the requirements of the device they need to build.

*Nonfunctional and Functional Requirements* – The software developers.

## References

[1] IEEE Software Engineering Standards Committee, “IEEE Std 830-1998, IEEE Recommended Practice for Software Requirements Specifications”, October 20, 1998.

[2]

[3]

***1.5 Overview***

The remainder of this document includes three chapters and appendixes.

The second one provides an overview of the system functionality and system interaction with other systems. Further, the chapter also mentions the system constraints and assumptions about the product.

The third chapter provides the requirements specification in detailed terms and a description of the different system interfaces. Different specification techniques are used in order to specify the requirements more precisely for different audiences.

The fourth chapter deals with the prioritization of the requirements. It includes a motivation for the chosen prioritization methods and discusses why other alternatives were not chosen. The fifth chapter describes other nonfunctional requirements.

The Appendixes in the end of the document include the all results of the requirement prioritization and a release plan based on them.

# Overall Description

## Product Perspective

*<Describe the context and origin of the product being specified in this SRS. For example, state whether this product is a follow-on member of a product family, a replacement for certain existing systems, or a new, self-contained product. If the SRS defines a component of a larger system, relate the requirements of the larger system to the functionality of this software and identify interfaces between the two. A simple diagram that shows the major components of the overall system, subsystem interconnections, and external interfaces can be helpful.>*

## Product Functions

*<Summarize the major functions the product must perform or must let the user perform. Details will be provided in Section 3, so only a high level summary (such as a bullet list) is needed here. Organize the functions to make them understandable to any reader of the SRS. A picture of the major groups of related requirements and how they relate, such as a top level data flow diagram or object class diagram, is often effective.>*

## User Classes and Characteristics

*<Identify the various user classes that you anticipate will use this product. User classes may be differentiated based on frequency of use, subset of product functions used, technical expertise, security or privilege levels, educational level, or experience. Describe the pertinent characteristics of each user class. Certain requirements may pertain only to certain user classes. Distinguish the most important user classes for this product from those who are less important to satisfy.>*

## Operating Environment

*<Describe the environment in which the software will operate, including the hardware platform, operating system and versions, and any other software components or applications with which it must peacefully coexist.>*

## Design and Implementation Constraints

*<Describe any items or issues that will limit the options available to the developers. These might include: corporate or regulatory policies; hardware limitations (timing requirements, memory requirements); interfaces to other applications; specific technologies, tools, and databases to be used; parallel operations; language requirements; communications protocols; security considerations; design conventions or programming standards (for example, if the customer’s organization will be responsible for maintaining the delivered software).>*

## User Documentation

*<List the user documentation components (such as user manuals, on-line help, and tutorials) that will be delivered along with the software. Identify any known user documentation delivery formats or standards.>*

## Assumptions and Dependencies

*<List any assumed factors (as opposed to known facts) that could affect the requirements stated in the SRS. These could include third-party or commercial components that you plan to use, issues around the development or operating environment, or constraints. The project could be affected if these assumptions are incorrect, are not shared, or change. Also identify any dependencies the project has on external factors, such as software components that you intend to reuse from another project, unless they are already documented elsewhere (for example, in the vision and scope document or the project plan).>*

# User Requirements

## Functional Requirements

*These are the non-technical versions of the Functional Requirements for the Fraction Worksheet Creator.*

* The user shall be able to select various difficulty levels based on student skill.
* The user shall be able to select various problem types at each difficulty level.
* The user shall be able to modify problem parameters such as denominator range.
* The user shall be able to modify worksheet parameters such as problem spacing, font type and font size.
* The user shall be able to customize the header at the top of each worksheet to include class name, date and title).
* The system shall generate random exercises for each worksheet.
* The system shall generate a tutorial / instructions for each problem type.
* The system shall allow saving via PDF.
* The system shall allow the user to preview the worksheet before printing.
* The system shall allow printing via PDF.
* The user shall be able to select the option to print an answer sheet for each worksheet.

## Non-Functional Requirements

*These are the non-technical versions of the Non-Functional Requirements for the Fraction Worksheet Creator*

* The system shall run on a Windows or MacOS device running Java 8 or higher.

## 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.>*

## 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.>*

# System Requirements

*<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.>*

## 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 a placeholder to indicate when necessary information is not yet available.>*

*<Each requirement should be uniquely identified with a sequence number or a meaningful tag of some kind.>*

REQ-1:

REQ-2:

## System Feature 2 (and so on)

# Other Nonfunctional Requirements

## Performance Requirements

*<If there are performance requirements for the product under various circumstances, state them here and explain their rationale, to help the developers understand the intent and make suitable design choices. Specify the timing relationships for real time systems. Make such requirements as specific as possible. You may need to state performance requirements for individual functional requirements or features.>*

## 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.>*

## Security Requirements

*<Specify any requirements 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.>*

## 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.>*

## 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.>*

# 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.>*

1. Three levels of difficulty for worksheets:
2. Beginner (addition and subtraction problems with visuals and images)
3. Intermediate (addition and subtraction without visuals, maybe introductory multiplication)
4. Advanced ( multiplication and division, word questions and problems, mixed numbers, least common multiple, greatest common factor)

*! This could go to our requirements (Olga)*