Team P5T

Citation Crawler

Software Requirements Specification

1.0

01/29/2015

# **Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Description** | **Author** | **Comments** |
| 01/29/15 | Version 1 | James Fitzgerald | First Revision |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

# **Document Approval**

The following Software Requirements Specification has been accepted and approved by the following:

|  |  |  |  |
| --- | --- | --- | --- |
| **Signature** | **Printed Name** | **Title** | **Date** |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

**Table of Contents**

**REVISION HISTORY**

**DOCUMENT APPROVAL**

**1. INTRODUCTION**

1.1 Purpose

1.2 Scope

1.3 Definitions, Acronyms, and Abbreviations

1.4 References

1.5 Overview

**2. GENERAL DESCRIPTION**

2.1 Product Perspective

2.2 Product Functions

2.3 User Characteristics

2.4 General Constraints

2.5 Assumptions and Dependencies

**3. SPECIFIC REQUIREMENTS**

3.1 External Interface Requirements

*3.1.1 User Interfaces*

*3.1.2 Hardware Interfaces*

*3.1.3 Software Interfaces*

*3.1.4 Communications Interfaces*

3.2 Functional Requirements

*3.2.1 <Functional Requirement or Feature #1>*

*3.2.2 <Functional Requirement or Feature #2>*

3.3 Use Cases

*3.3.1 Use Case #1*

*3.3.2 Use Case #2*

3.4 Classes / Objects

*3.4.1 <Class / Object #1>*

*3.4.2 <Class / Object #2>*

3.5 Non-Functional Requirements

*3.5.1 Performance*

*3.5.2 Reliability*

*3.5.3 Availability*

*3.5.4 Security*

*3.5.5 Maintainability*

*3.5.6 Portability*

3.6 Inverse Requirements

3.7 Design Constraints

3.8 Logical Database Requirements

3.9 Other Requirements

**4. ANALYSIS MODELS**

4.1 Sequence Diagrams

4.3 Data Flow Diagrams (DFD)

4.2 State-Transition Diagrams (STD)

**5. CHANGE MANAGEMENT PROCESS**

**A. APPENDICES**

A.1 Appendix 1

A.2 Appendix 2

# **1. Introduction**

## **1.1 Purpose**

This Software Requirement Specification document is intended to provide adequate information for current and future developers to implement and maintain the Crawler framework, and any modules to be used by the crawler framework. It is also intended for the sponsor to ensure that the project team has correctly understood the intentions of the sponsor.

## **1.2 Scope**

*This subsection should:*

*(1) Identify the software product(s) to be produced by name; for example, Host DBMS, Report Generator, etc*

*(2) Explain what the software product(s) will, and, if necessary, will not do*

*(3) Describe the application of the software being specified. As a portion of this, it should:*

*(a) Describe all relevant benefits, objectives, and goals as precisely as possible. For example, to say that one goal is to provide effective reporting capabilities is not as good as saying parameter-driven, user-definable reports with a 2 h turnaround and on-line entry of user parameters.*

*(b) Be consistent with similar statements in higher-level specifications (for example, the System Requirement Specification) , if they exist.What is the scope of this software product.*

The team will produce the framework for the Crawler, and a module that allows the crawler to scrape the CiteSeerX database and provide citation information for user research and suggest similar research when a user searches the database.

## **1.3 Definitions, Acronyms, and Abbreviations**

Crawler - an automated program that scrapes information from available APIs and sites

API - application program interface. A set of routines, protocols and tools for building software applications.

## **1.4 References**

## **1.5 Overview**

The rest of this SRS contains information regarding product description, functions, constraints, and contains use cases and diagrams relating to users and product functions. It is organized according to the table of contents contained at the beginning of the document.

# **2. General Description**

*This section of the SRS should describe the general factors that affect 'the product and its requirements. It should be made clear that this section does not state specific requirements; it only makes those requirements easier to understand.*

## **2.1 Product Perspective**

Currently, there are existing databases with citation information related to scientific research and research papers and publications. However, many of these services exist behind a paywall or require fees. The intent of this crawler is to gather information from public-facing interfaces or services that legally obtain the content necessary to provide adequate and accurate citation information.

## **2.2 Product Functions**

*This subsection of the SRS should provide a summary of the functions that the software will perform.*

The Crawler will enable users to check citation information for their research, as well as suggest related research contained in the Crawler database. The Crawler will be separate from, but integrate with the existing web interface.

## **2.3 User Characteristics**

*This subsection of the SRS should describe those general characteristics of the eventual users of the product that will affect the specific requirements. (See the IEEE Guide to SRS for more details).*

The intended users of the system are researchers looking to find citation information for their research, and developers looking to build new add-ons or modules for the framework.

## **2.4 General Constraints**

*This subsection of the SRS should provide a general description of any other items that will*

*limit the developer’s options for designing the system. (See the IEEE Guide to SRS for a partial list of possible general constraints).*

Given the previous requirements for the user interface, the code must be done using C# MVC and .NET. The database used must also be in MS SQL 2008.

## **2.5 Assumptions and Dependencies**

*This subsection of the SRS should list each of the factors that affect the requirements stated in the SRS. These factors are not design constraints on the software but are, rather, any changes to them that can affect the requirements in the SRS. For example, an assumption might be that a specific operating system will be available on the hardware designated for the software product. If, in fact, the operating system is not available, the SRS would then have to change accordingly.*

We make the assumption that there are publicly available interfaces that can be used to obtain citation information and that future developers will be able to use these and have the knowledge to use the interfaces. As such, the Crawler framework is dependent on using these interfaces to obtain citation information.

# **3. Specific Requirements**

*This will be the largest and most important section of the SRS. The customer requirements will be embodied within Section 2, but this section will give the D-requirements that are used to guide the project’s software design, implementation, and testing.*

*Each requirement in this section should be:*

* *Correct*
* *Traceable (both forward and backward to prior/future artifacts)*
* *Unambiguous*
* *Verifiable (i.e., testable)*
* *Prioritized (with respect to importance and/or stability)*
* *Complete*
* *Consistent*
* *Uniquely identifiable (usually via numbering like 3.4.5.6)*

*Attention should be paid to the carefuly organize the requirements presented in this section so that they may easily accessed and understood. Furthermore, this SRS is not the software design document, therefore one should avoid the tendency to over-constrain (and therefore design) the software project within this SRS.*

## **3.1 External Interface Requirements**

### **3.1.1 User Interfaces**

### **3.1.2 Hardware Interfaces**

### **3.1.3 Software Interfaces**

### **3.1.4 Communications Interfaces**

## **3.2 Functional Requirements**

*This section describes specific features of the software project. If desired, some requirements may be specified in the use-case format and listed in the Use Cases Section.*

### **3.2.1 <Functional Requirement or Feature #1>**

3.2.1.1 Introduction

3.2.1.2 Inputs

3.2.1.3 Processing

3.2.1.4 Outputs

3.2.1.5 Error Handling

### **3.2.2 <Functional Requirement or Feature #2>**

…

## **3.3 Use Cases**

### **3.3.1 Use Case #1**

### **3.3.2 Use Case #2**

…

## **3.4 Classes / Objects**

### **3.4.1 <Class / Object #1>**

3.4.1.1 Attributes

3.4.1.2 Functions

<Reference to functional requirements and/or use cases>

### **3.4.2 <Class / Object #2>**

…

## **3.5 Non-Functional Requirements**

### **3.5.1 Performance**

 The Crawler shall pull updates from the associated sites and update its own database once per day. The database shall perform transactions in less than a second.

### **3.5.2 Reliability**

 Transactions to the Crawler database shall be ACID, and the information contained within the database will be correct as of the last update.

### **3.5.3 Availability**

 The crawler shall have 5 9’s of uptime.

### **3.5.4 Security**

 The information contained within is largely already publicly available, and the sponsor has dictated that security is not a major concern.

### **3.5.5 Maintainability**

 The crawler shall have a modular structure so it is easy to understand and implement further modules for adding new sites to be scraped by the crawler.

## **3.6 Inverse Requirements**

*State any \*useful\* inverse requirements.*

## **3.7 Design Constraints**

*Specify design constrains imposed by other standards, company policies, hardware limitation, etc. that will impact this software project.*

## **3.8 Logical Database Requirements**

*Will a database be used? If so, what logical requirements exist for data formats, storage capabilities, data retention, data integrity, etc.*

The database shall be of the same MS SQL 2008 format as the web interface, but will be separate from the existing web interface database to prevent any conflicts in transactions. It is possible that the database for the crawler will be downloaded from the CiteSeerX database for the first module.

## **3.9 Other Requirements**

*Catchall section for any additional requirements.*

# **4. Analysis Models**

*List all analysis models used in developing specific requirements previously given in this SRS. Each model should include an introduction and a narrative description. Furthermore, each model should be traceable the SRS’s requirements.*

## **4.1 Sequence Diagrams**

## **4.3 Data Flow Diagrams (DFD)**

## **4.2 State-Transition Diagrams (STD)**

# **5. Change Management Process**

Changes may be submitted by members of the development team and must be approved by the project sponsor and faculty sponsor. Changes will be approved by the team in the next team meeting, and then by the sponsor at the next sponsor meeting, or via email if a sponsor meeting is not available in a timely manner.

# **A. Appendices**

*Appendices may be used to provide additional (and hopefully helpful) information. If present, the SRS should explicitly state whether the information contained within an appendix is to be considered as a part of the SRS’s overall set of requirements.*

*Example Appendices could include (initial) conceptual documents for the software project, marketing materials, minutes of meetings with the customer(s), etc.*

## **A.1 Appendix 1**

## **A.2 Appendix 2**