SOFTWARE REQUIREMENTS SPECIFICATION

for

Support program for sequence's research

Version 1.0 approved

Prepared by Juan Toca Mateo
Universidad de Cantabria

July 20, 2020

Contents

1 Introduction

1.1 Purpose

This software product aims to make life easier to researchers in the field of sequences with interesting autocorrelation and crosscorrelation properties. For that, we will provide a way of computing those sequences and store them for future use.

1.2 Document Conventions

TODO

1.3 Intended Audience

This document is aimed to each agent involved in the development of this software project. In one hand, the developer will take it as a guide of the client needs and, in the other hand, the client will have a document to check if the final product conforms to the agreement.

In particular, the developer is defined as the person that will present it's bachelor thesis(Juan Toca) and the client is defined as his thesis director (Domingo Gómez).

In case someone wants to build in top of this project, this document provides the starting point to get a general idea of which issues have been tackled and in which ways the project could be improved.

1.4 Project Scope

The main purpose of this project is to provide a graphical framework to assist in the research of the properties of sequences, more precisely their autocorrelation and cross-correlation properties.

This software must be capable of exploiting the computational power of a calculation center to assist in the said research and store the result of this computations in a structured way.

It must be as easy to use as possible, taking into account the characteristics of it's higly specialized public.

1.5 References

TODO

2 Overall Description

2.1 Product Perspective

This product will assist in the research on symbol sequences with different correlation properties to be used in fields such as WI-FI, sonar, etc..

2.2 Product Functions

We will summarize in a general way the functionalities we expect from this software.

- Perform computations in search for sequences.
  - Obtain real-time information of the computation proccess
  - Change dedicated resources to computation
- Manage the set of found sequences
  - Query by different parameters of the sequences
  - Handle found sequences manually
  - Plot sequence properties
- Manage permissions

2.3 User Classes and Characteristics

Users of this project can be divided in 2 main groups:

Researchers that will control the search for new sequences, being capable of starting computations and modifying the database. This researchers are highly specialized personal and are expected to understand common abbreviations in the field as well as knowing which parameters are more promising to get a successful computation.

Engineers that will query the database searching for sequences to apply in a practical use. They are not expected to know about each single detail about how this sequences are obtained, although they know their properties and how can be applied in the real world.

2.4 Operating Environment

The computation module is expected to run in highly paralelized systems such as supercomputers. This is the enviroment in which researchers work so it would be a shame if we didn't design the software to take advantage of that.

Taking his into account, we should target the software installed in those enviroments. This leads us to assume a Linux enviroment(as most supercomputers run it) and a Python interpreter(as it's a popular language in the scientific community).

2.5 User Documentation

As this project has a limited scope, we think there is no need in a complete user manual/wiki appart from a well documented README.md and a man page.

Keeping in mind that the end user is supposed to have knowledge in the topic, contextual help explaining the concepts doesn't need to be too extense.

2.6 Assumptions and Dependencies

DOES NOT APPLY

3 External Interface Requirements

3.1 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 for which a user interface is needed. Details of the user interface design should be documented in a separate user interface specification.>

3.2 Hardware Interfaces

Taking into account that the software module in charge of running the computations might be deployed in a cluster, we must consider their characteristics. Their architecture of highly paralelized systems rely on a problematic choke point, the interconnection network.

Each time a proccess needs to read/write data in a memory address shared with a proccess in other node, this information must be sent through the interconnection network. This will slow down significantly the performance of the nodes involved.We must take this hardware constraint into account when choosing the parallel programming model.

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.>

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 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:

4.2 System Feature 2 (and so on)

5 Other Nonfunctional Requirements

5.1 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.>

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.>

5.3 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.>

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.>

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.>

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.>

6.1 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.>

6.2 Appendix B: Analysis Models

<Optionally, include any pertinent analysis models, such as data flow diagrams, class diagrams, state-transition diagrams, or entity-relationship diagrams.>

6.3 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.>