Software Requirements Specification

ODE Grapher - Version 1.0

Prepared by:

1. **Pradeep Kumar R (220001058)**
2. **M. Sai Varshith (220001044)**
3. **Pavan Kumar A (220001011)**
4. **Aadish Jain (220001001)**
5. **Saket Thamke (220001067)**

Submitted to: Dr. Abhisheik Srivastava

[CSE / IIT INDORE]

Date: January 21, 2024

Revision History

Date	Description	Author	Comments
21/04/24	V1 :- Initial SRS	Pradeep Kumar	Initial SRS Approved

Document Approval

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

Signature	Printed Name	Title	Date
Pradeep	Pradeep Kumar R	Team Lead.	21/04/24
M Sai Varshith	M Sai Varshith	Member	21/04/24
Pavan Kumar A	Pavan Kumar A	Member	21/04/24
Aadish Jain	Aadish Jain	Member	21/04/24
Saket Thamke	Saket Thamke	Member	21/04/24

Table of Contents

Software Requirements Specification (SRS)

1. Introduction

1.1 Purpose

This Software Requirements Specification (SRS) defines the requirements for the development of a sophisticated simulation and analysis system focused on "Tiered Synchronization in Kuramoto Oscillators with Adaptive Higher-Order Interactions." It serves as a comprehensive guide for software engineers and developers involved in designing and implementing this advanced research tool.

1.2 Scope

1.2.1 Software Product

The software product under development is a cutting-edge simulation and analysis system tailored for researchers and engineers investigating tiered synchronization phenomena in Kuramoto oscillators with adaptive higher-order interactions.

1.2.2 Functionality

The software will offer the following key functionalities:

- Accurate simulation of Kuramoto oscillators with adaptive higher-order interactions.
- In-depth analysis of tiered synchronization patterns.
- Real-time visualization of simulation results.

1.2.3 Application Description

Targeting researchers and engineers, this application will facilitate the study of adaptive higher-order interactions in Kuramoto oscillators, particularly focusing on the intriguing phenomenon of tiered synchronization.

1.3 Definitions, Acronyms, and Abbreviations

Please refer to the glossary provided in [Appendix A.1] for detailed definitions of terms, acronyms, and abbreviations used throughout this document.

1.4 References

1.4.1 Document List

1. "Tiered Synchronization in Kuramoto Oscillators with Adaptive Higher-Order Interactions" - Research Paper, XYZ Journal, January 2024
2. "Kuramoto Oscillators: Principles and Applications" - Book, ABC Publishers, 2022

1.4.2 Sources

1. All documents referenced in this SRS can be obtained from XYZ Journal and ABC Publishers.

1.5 Overview

1.5.1 Content

This SRS is organized into distinct sections, including Introduction, General Description, Specific Requirements, Analysis Models, Change Management Process, and Appendices.

1.5.2 Organization

Carefully structured for clarity, this document provides a systematic overview of the requirements, ensuring ease of comprehension for all stakeholders.

2. General Description

2.1 Product Perspective

The system stands as an independent, sophisticated research tool designed for the simulation and analysis of tiered synchronization in Kuramoto oscillators with adaptive higher-order interactions. It will seamlessly interface with other analysis tools to facilitate comprehensive research endeavors.

2.2 Product Functions

2.2.1 Simulation of Kuramoto Oscillators

The system will simulate Kuramoto oscillators with adaptive higher-order interactions using input parameters provided by the user, such as k and n.

2.2.2 Analysis of Tiered Synchronization Patterns

In-depth analysis capabilities will be provided to discern and understand tiered synchronization patterns emerging from the simulation results.

2.2.3 Real-Time Visualization

Real-time visualization tools will accompany the simulation, enabling users to observe and analyze the evolving patterns immediately.

2.3 User Characteristics

The target users for this software include researchers and engineers with expertise in Kuramoto oscillators, adaptive interactions, and simulation tools.

2.4 General Constraints

The system is designed to operate seamlessly on Windows, macOS, and Linux platforms, providing a wide range of accessibility to the user base.

2.5 Assumptions and Dependencies

The software assumes the availability of a standard computer with adequate processing power and compatibility with Python 3.7 and above.

3. Specific Requirements

3.1 External Interface Requirements

3.1.1 User Interfaces

The user interface shall feature:

- Input parameters for Kuramoto oscillators simulation.
- Visualization tools for tiered synchronization patterns.

3.2 Functional Requirements

3.2.1 Simulation and Analysis

3.2.1.1 Introduction

The system shall provide an accurate and comprehensive simulation of Kuramoto oscillators with adaptive higher-order interactions.

3.2.1.2 Inputs

Users shall be able to input specific parameters, such as O and w, to tailor the simulation according to their research needs.

3.2.1.3 Processing

The system shall process the simulation in real-time, ensuring swift and accurate results based on the user-defined parameters.

3.2.1.4 Outputs

Simulation outputs shall include detailed tiered synchronization patterns and visual representations for enhanced analysis.

3.2.1.5 Error Handling

A robust error-handling mechanism will be implemented to provide users with meaningful feedback in case of invalid input parameters or unforeseen errors.

3.2.2 Real-Time Visualization

The system shall provide real-time visualization tools, allowing users to observe and analyze simulation results as they unfold.

3.3 Use Cases

3.3.1 Research Scenario

In a typical research scenario, a user aims to study the impact of adaptive higher-order interactions on tiered synchronization in Kuramoto oscillators.

3.3.2 Comparative Analysis

Users may engage in comparative analysis, comparing simulation results under different sets of input parameters to draw meaningful conclusions.

3.4 Classes / Objects

3.4.1 SimulationEngine

3.4.1.1 Attributes

- Parameters (O, w)
- Simulation Results

3.4.1.2 Functions

- RunSimulation()
- AnalyzeResults()

3.4.2 VisualizationModule

3.4.2.1 Attributes

- Simulation Data
- Visualization Options

3.4.2.2 Functions

- DisplaySimulationResults()

3.5 Non-Functional Requirements

3.5.1 Performance

The system must strive to complete 95% of simulations within a specified time frame, aiming for efficiency and user satisfaction.

3.5.2 Reliability

To ensure a dependable user experience, the system shall have a downtime of no more than one hour per week for routine maintenance.

3.5.3 Availability

The system's availability target is set at an impressive 99.9%, promoting consistent and reliable access for users.

3.5.4 Security

Data integrity and user privacy shall be prioritized and maintained through robust security measures.

3.5.5 Maintainability

To facilitate future updates and enhancements, the software shall be designed with maintainability in mind, ensuring smooth transitions between versions.

3.5.6 Portability

The system shall be designed to be easily portable across different operating systems, enhancing accessibility for users on diverse platforms.

3.6 Inverse Requirements

The system should not impose limitations on the number of Kuramoto oscillators or adaptive parameters, allowing users flexibility in their research endeavors.

3.7 Design Constraints

The system's design shall adhere to established Python coding standards, ensuring code quality and maintainability.

3.8 Logical Database Requirements

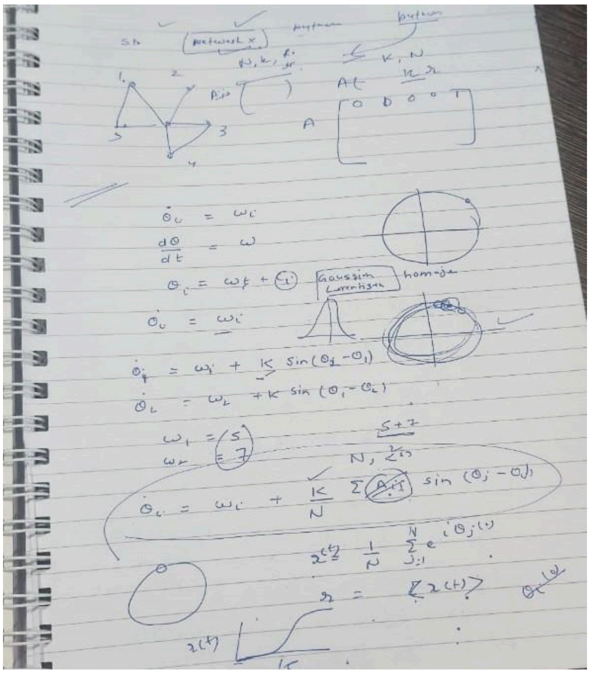
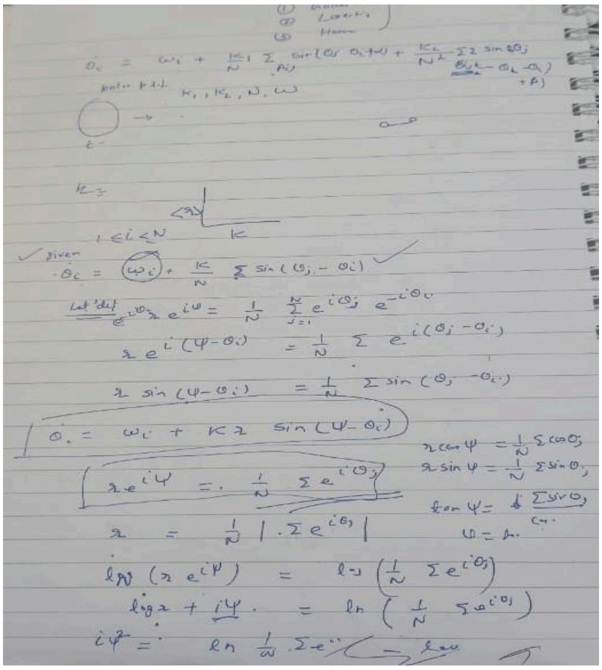
No database integration is necessary for the core functionality of this standalone research tool.

3.9 Other Requirements

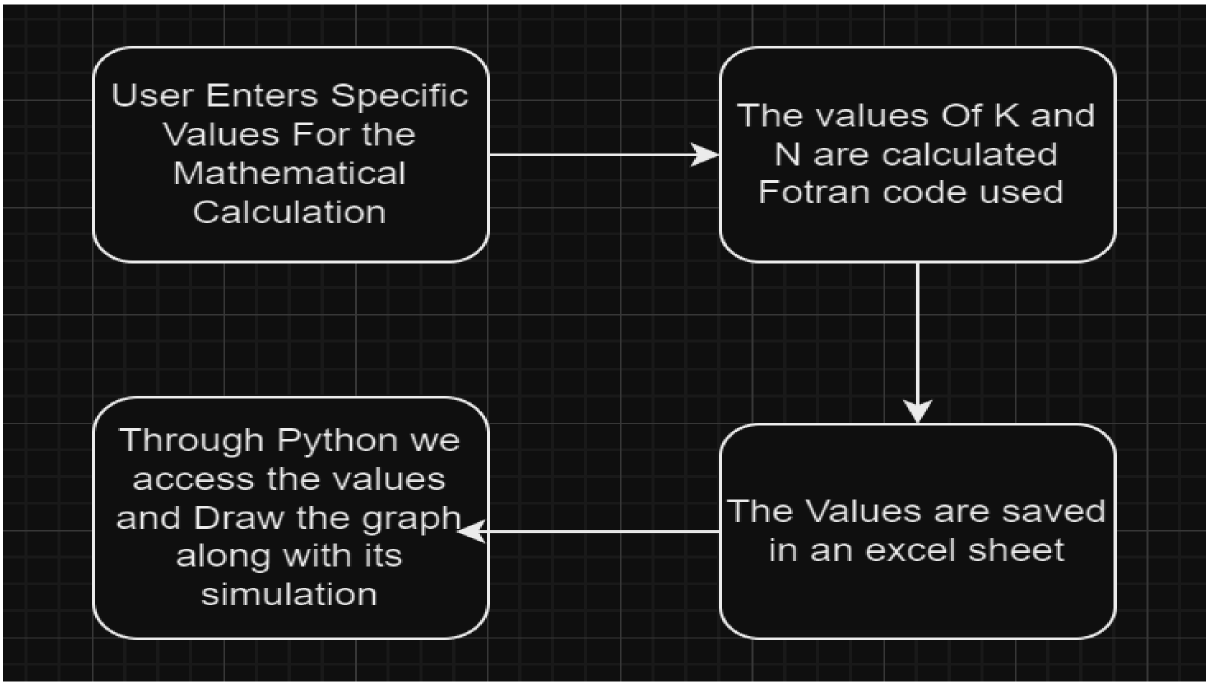
- The system shall provide a comprehensive user manual for the software's effective utilization (refer to [Appendix A.3]).

4. Analysis Models

4.1 Sequence Diagrams (Flow Of Data (Maths - Code))

4.2 Data Flow Diagrams (DFD)



5. Change Management Process

Changes to the SRS shall adhere to the Change Management Process outlined in [Appendix A.2], providing a systematic approach to handling updates and modifications.

A. Appendices

A.1 Glossary

This glossary provides definitions for terms, acronyms, and abbreviations used throughout the Software Requirements Specification (SRS).

- **Kuramoto Oscillators:** A mathematical model used to describe the synchronization of a large ensemble of coupled oscillators.
- **Adaptive Higher-Order Interactions:** Dynamic interactions between oscillators that adjust based on certain conditions or parameters during simulation.
- **Tiered Synchronization:** A phenomenon observed in oscillatory systems where synchronization occurs in hierarchical patterns.
- **Simulation Engine:** The core module responsible for executing Kuramoto oscillator simulations with adaptive higher-order interactions.
- **Visualization Module:** The component that handles real-time visualization of simulation results, aiding team members in the analysis process.
- **Downtime:** The duration during which the system is not available for use, typically reserved for maintenance and updates.
- **Change Management Process:** A systematic approach for handling changes to the SRS, involving communication with the team lead.
- **User Manual:** Comprehensive documentation providing team members with instructions and information on the functionalities and usage of the software.

A.2 Change Management Process

This section outlines the Change Management Process to guide team members through the submission, review, and approval of changes to the Software Requirements Specification (SRS).

A.2.1 Communication

1. **Initiation of Change:**
    - Any team member may propose a change to the SRS by directly communicating with the team lead.
    - The communication should include details such as the nature of the change, reasons for the change, and potential impact.
2. **Informal Review:**
    - The team lead reviews and evaluates the proposed change informally, considering feasibility, impact, and relevance.

A.2.2 Discussion

3. **Team Discussion:**
   - The proposed change is discussed within the team to gather input on potential implications and benefits.
   - Team members share their perspectives and insights.
4. **Clarification:**
   - If needed, team members seek clarification from the proposer or provide additional information for a clearer understanding.

A.2.3 Decision

5. **Decision-Making:**
   - The team lead makes a decision on whether to approve, reject, or defer the proposed change.
   - The decision is based on team discussions and considerations.
6. **Informal Documentation:**
   - Approved changes are informally documented and communicated to team members.
   - Rejected changes are accompanied by clear explanations.

A.2.4 Implementation

7. **Implementation Planning:**
   - If the change is approved, an informal implementation plan is discussed within the team.
8. **Execution:**
   - The approved change is implemented into the SRS, and progress is communicated informally within the team.

A.2.5 Verification

9. **Verification:**
   - The team collectively verifies that the implemented change aligns with the approved specifications.
   - Adjustments are made as necessary.

A.2.6 Documentation

10. **Updated SRS:**
    - The updated SRS, reflecting the approved changes, is communicated informally to team members.
    - The team maintains an awareness of the changes made to the document.

A.3 User Manual

1. **Higher order interactions in complex networks**
   - **https://drive.google.com/file/d/1nlUqntzCKLBpBf7UcW58exhq-MpvHHeB/view?usp=sharing**

2. **Tiered synchronization in Kuramoto oscillators**
    - https://drive.google.com/file/d/1J3q4WO-zWB-nIOWoUp0eEI3t5IANJtuU/view?usp=sharing

---