

**Simulation of Different Dead lock Scenarios and Avoiding the Deadlocks**

**High Level Design**

**Document Control:**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Project Revision History** | | | | | | |
|  |  |  | |  |  |  |
| **Date** | **Version** | **Author** | **Brief Description of Changes** | | | |
|  |  |  |  | | | |
|  |  |  |  | | | |
|  |  |  |  | | | |
|  |  |  |  | | | |
|  |  |  |  | | | |

|  |
| --- |
| **Team Members** |

|  |  |
| --- | --- |
| **Employee ID:** | **Name** |
| 46282103 | Saranya Munigolla |
| 46282105 | Shwetha Julappayyanapalya Parashuramareddy |
| 46290160 | Lakshmi Durga Pippara |
| 46282106 | A.P.Senthamil Selvi |
| 46290166 | Yalavarthi Sravya |
| 46290161 | Geethanjali Sambana |

|  |
| --- |
| **Table of Contents** |

1. **High Level Design**

|  |  |  |
| --- | --- | --- |
| 1. Introduction | | 5 |
| 1.1 | Purpose | 5 |
| 1.2 | Scope | 5 |
| 1.3 | Overview | 5 |
| 2. General Description | | 5 |
| 2.1 | Product Perspective | 5 |
| 2.2 | Tools used | 6 |
| 2.3 | Special Design aspects | 6 |
| 3. Design Details | | 6 |
| 3.1 | Main Design Features | 6 |
| 3.2 | Application Architecture | 7 |
| 3.3 | Standards | 7 |
| 3.4 | Data Flow Level-0 | 7 |
| 3.5 | Data Flow Level-1 | 8 |
| 3.6 | User Interface | 8 |
| 3.7 | Error Handling | 8 |
| 3.8 | Help | 8 |
| 3.9 | Performance | 8 |
| 3.10 | Reliability | 8 |
| 3.11 | Maintainability | 9 |
| 3.12 | Portability | 9 |
| 3.13 | Reusability | 10 |
| 3.14 | Application compatibility | 10 |

|  |
| --- |
| **High Level Design** |

**1. Introduction**

**1.1 Purpose**

The purpose of this High-Level Design (HLD) Document is to add the necessary detail to the current project description to represent a suitable model for coding. This document is also intended to help detect contradictions prior to coding and can be used as a reference manual for how the modules interact at a high level.

**1.2 Scope**

This document provides a comprehensive high level design overview of the different dead lock scenarios and deadlock detection scope of the project is to identify the conditions for deadlocks, distinguish the different circumstances that lead to this undesirable state, and identify the methods for detection, prevention, and recovery.

It highlights the high-level flow / use cases in simulation of different dead lock scenarios and serves as an input to the low level design documents that would further elaborate on the proposed system design.

**1.3** **Overview**

This HLD Document is arranged in the following format:

-Section1: Introduction

A brief explanation about the purpose, aim, scope, and design format of the proposed project.

- Section 2: General Description

This section is all about the general constraints, assumptions, and design aspects associated with the proposed project. The product perspective will give an overall description of the simulation and detection of dead locks.

- Section 3: Design Details

This section documents the detailed design of all modules associated with the development of the proposed simulation and detection of dead locks.

1. **General Description**

**2.1 Product Perspective**

Deadlock in operating system is a situation which occurs when a process or thread enters a waiting state because a resource requested is being held by another waiting thread, which in turn is waiting for another resource held by another waiting thread. In a deadlock state a thread is unable to change its state(waiting) indefinitely because the resources requested by it are being used by another waiting thread.

**2.2** **Tools used**

1. C tools

2. C language is used.

**2.3 Special Design aspects**

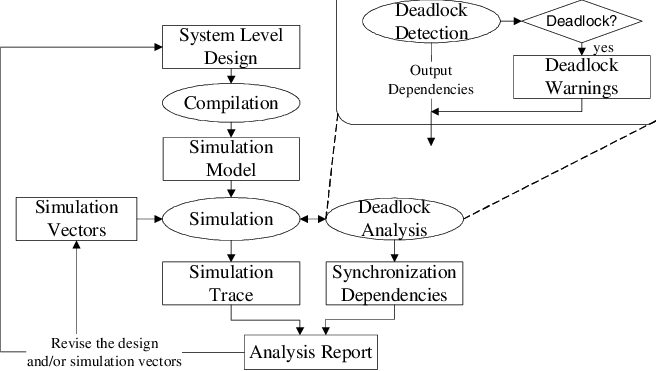
One of the design aspects is that the system will work with a single user at a time.

**3. Design Details**

**3.1 Main Design Features**

The main design features include four major parts: the architecture, the user interface design, the files, process relation, and automation. In order to make these designs easier to understand, the design has been illustrated in attached diagrams (Use Case, Data flow diagrams).

**3.2 Application Architecture**

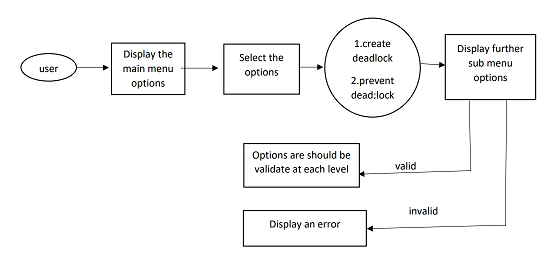


**FIG: Simulation based of deadlock analysis**

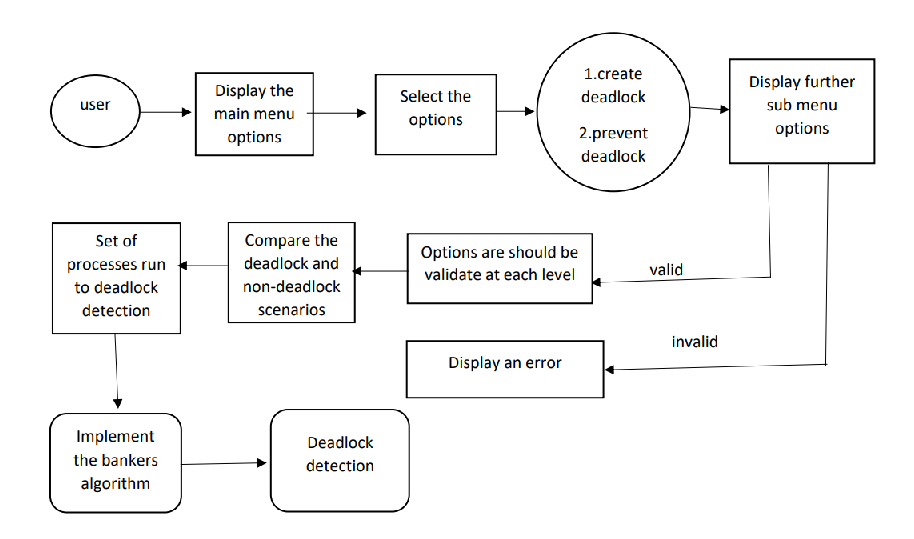
**3.3 Standards**

* Security –NA
* Quality – by keeping the interface simple and direct, quality should be kept at a maximum.

**3.4 Data Flow Diagram (Level - 0)**



**3.5 Data Flow Diagram (Level - 1)**



**3.6 User Interface**

Command Line interface.

**3.7 Error Handling**

Should errors be encountered, an explanation will be displayed as to what went wrong.

An error will be defined as anything that falls outside the normal and intended usage.

**3.8 Help**

Help will come in the form of all the documentation created prior to coding, which explains the intended uses. Should time allow, detailed instructions will be written on how to create and implement the system with the intention of publishing as an Open Source solution.

**3.9** **Performance**

Performance is going to be very important for this project. For everything to run smoothly for this project, this will work on the user terminal, performance depends upon the hardware component of the user and efficiency is dependent on the algorithm selected by the user.

**3.10 Reliability**

The simulation can schedule processes flawlessly and provide a nice overview to the user about the algorithms implemented.

**3.11 Maintainability**

NA

**3.12 Portability**

This system should have the ability that, once it is together, the entire system should be able to be physically moved to any location. Code and program portability should be possible between kernel-recompiled Linux distributions. For everything to work properly, all programs should be in one folder.

**3.13 Reusability**

The code has the ability to be reused with no problems. Everything will be completely reusable to anyone.

**3.14 Application compatibility**

This was designed as an independent system. As it is not connected to any other components or interfaces, application compatibility is not a concern.