

**Custom Protocol Implementation**

Low Level Design

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

Low-Level Design Version Draft v0.2

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

|  |  |
| --- | --- |
| **Employee ID:** | **Name** |
| **46290132** | **Deepthi J** |
| **46290123** | **Sai Sudheshna Revuri** |
| **46290133** | **Shaik Rasheeda Begum** |
| **46290126** | **Vijaya Jangala** |
| **46290137** | **Varalakshmi Kosana** |

**Document Control:**

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

**I) Low-Level Design**

|  |  |
| --- | --- |
| 1. Introduction |  |
| 1.1 Purpose | 4 |
| 1.2 Document Conventions | 4 |
| 1.3 Intended Audience and Reading Suggestions | 4 |
| 1.4 References | 4 |
| 1. Detailed system design | 5 |
| 2.1 Design Description | 5 |
| 2.2 Flowchart | 6 |
| 2.3 Modules | 7 |
| 2.4 Structure used | 11 |
| 2.5 Design and implementation constraints | 11 |

|  |
| --- |
| **Low-Level Design** |

**1. Introduction**

The aim of this document is to gather, analyze and give an in-depth insight into the complete mechanism of custom protocol implementation between client and server by defining the problem statement in detail. The intended audience includes all stakeholders, Users, and developers. The detailed low-level design of Custom message protocol is provided in this document.

### 1.1 Purpose

This document describes the low-level design flow of the Custom protocol implementation.

### 1.2 Document Conventions

TBD (To be continued).

### 1.3 Intended Audience and Reading Suggestions

The document is primarily intended for team members, which consists of trainees under the **Capgemini** Training Program.

### 1.4 Reference

The references are:

1. System Requirements Specification Document

## 2. Detailed System Design

**2.1 Design Descriptions:**

This project design mainly focuses on implementing some set of message protocol between the sender and receiver. User will be able to send a message request to perform corresponding action. Users must enter the correct message type to ensure the appropriate action occurs. This protocol happens between the client and server. When the client sends a request it becomes user, and the server becomes receiver and vice versa. Then the receiver receives the request and performs the specified action.

**Main menu:**

User gets options to open the message type menu or to exit the program. The client can send requests, the one who sends the request becomes the user while the other one becomes the receiver.

**Message type selection menu:**

User gets option to choose message type and according to that the action should take place. Only the corresponding action to the message type should take place.

1. If message type is 1, then the corresponding action to create a file takes place. It also allows us to write contents to the file

2. If message type is 2, then the corresponding action to update contents of a file takes place.

3. If message type is 3, then the corresponding action to delete a file takes place. It asks for filename and deletes the file.

4. If message type is 4, the corresponding action to perform the functionality of a ls command should be done. Ls command lists the files.

5. If message type is 5, the corresponding action to perform the functionality of a cat command should be done. Cat command lists the content of the files.

6. If message type is 6, then the selection menu exits.

**2.2 Flowchart**

Diagram

Description automatically generated

**2.2.2 Storyboard**

**Diagram

Description automatically generated**

**2.3 Modules**

**2.3.1 ui.c**

**main ()**

| **Name** | main | | | |
| --- | --- | --- | --- | --- |
| **Input** | Parameter Name | NA | Initial value: | - |
| **Output** | Return value type | int | - | - |
| **Description** | Shows the options to open the message type menu or to exit the program. | | | |
| **Pseudo Code** | 1. Display the main menu  2. Wait for user options  3. Based on user input further action takes place | | | |

**2.3.2 ui\_val()**

| **Name** | ui\_val | | | |
| --- | --- | --- | --- | --- |
| **Input** | Parameter Name | int |  | To check whether user input is valid or not. |
| **Output** | Return value type | int |  | NA |
| **Description** | This program is used to set limits to the input values of the User Input. | | | |
| **Pseudo Code** | 1. Validate the inputs based on the menu level.  2. If invalid return error message, else perform certain action | | | |

**2.3.3 Create\_file()**

| **Name** | Create\_file | | | |
| --- | --- | --- | --- | --- |
| **Input** | Parameter Name | Msg\_type 1 |  | - |
| **Output** | Return value type |  |  | - |
| **Description** | This function on giving its message type should create a file whenever the message is inputted. | | | |
| **Pseudo**  **Code** | 1. Asks the user the name of the file  2. creates the file | | | |

**2.3.4 Copy\_file()**

| **Name** | Copy\_file | | | |
| --- | --- | --- | --- | --- |
| **Input** | Parameter Name |  |  | - |
| **Output** | Return value type |  |  | - |
| **Description** | This function on giving its message type should open a certain file and copies the contents to new file | | | |
| **Pseudo**  **Code** | 1. Asks the user the name of the file  2. copies the file contents | | | |

**2.3.5 Delete\_file()**

| **Name** | Delete\_file | | | |
| --- | --- | --- | --- | --- |
| **Input** | Parameter Name |  |  | - |
| **Output** | Return value type |  |  | - |
| **Description** | This function on giving its message type should open a certain file and should delete the file. | | | |
| **Pseudo**  **Code** | 1. Asks the user the name of the file  2. After finding it deletes the file | | | |

**2.3.6 ls\_com()**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | Ls\_com() | | | |
| **Input** | Parameter Name |  |  | - |
| **Output** | Return value type |  |  | - |
| **Description** | This function on giving its message type should perform functionality of ls command. The ls command is used to list all the files in a directory. | | | |
| **Pseudo**  **Code** | 1. Asks the user to the directory  2. List all the files in the directory | | | |

**2.3.7 cat\_com()**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Name** | cat\_com() | | | |
| **Input** | Parameter Name |  |  | - |
| **Output** | Return value type |  |  | - |
| **Description** | This function on giving its message type should perform functionality of cat command. The cat command shows the entire contents of a file | | | |
| **Pseudo**  **Code** | 1. Asks the user to the file as an input  2. shows all the contents present in the file | | | |

**2.4 Structure used**

* **The proc structure was created to store parameters for each entered binary.**

typedef struct

{

msg\_type\_e msg\_type;

char msg\_data[MAX\_DATA\_LEN];

char msg\_add\_data[MAX\_DATA\_LEN];

} msg\_t;

**2.5 Design and Implementation Constraints**

The system is built using the C language using concepts of system programming.