SADesignerLite

A Structured Analysis Software for Software Engineers

Group G13

Amogha Yalgi	119CS0570
Divyajyoti Panda	119CS0546
Leo Raphael Rodrigues	119CS0787
Rahul Meher	119CS0539

Contents

S.No.	Title	Page
1.	Introduction	2
1.1	Purpose	2
1.2	Scope	2
1.3	Definitions, Acronyms, Abbreviations	2
1.4	Overview	3
2.	Objectives	4
2.1	Product Perspectives	4
2.2	Product Functions	4
2.3	User Characteristics	5
2.4	Constraints	5
2.5	Dependencies	5
3.	Specific Requirements	6
3.1	External Interfaces	6
3.2	Functions	6
3.3	Performance Requirements	7
3.4	Design Constraints	7
3.5	Software System Quality Attributes	8

1. Introduction

1.1. Purpose

To build software to build data flow diagrams (DFD) and convert them into a data dictionary for structured software analysis. The projects target students and working professionals who are working on software analysis.

1.2. Scope

The software allows the user to

- Construct, open, edit and save annotated data flow diagrams graphically
- Decompose processes to generate higher-level processes
- Debug errors in the data flow diagram
- Generate data dictionary or image from the data flow diagram
- Print the image of the data flow diagram

1.3. Definitions, Acronyms, Abbreviations

- Data Flow Diagram (DFD) - A graph that maps out the flow of information between processes.
- Data store - A structure that stores data
- Data process - A function that processes data
- External entity - An entity that interacts with the system but is not a part of it.
- Data flow - A flow of data
- Label - Annotation giving further details on a DFD
- Data Dictionary (DD) - A document that catalogs data entities' structure and communication.
- Pixel (px)
- Operating System (OS)
- JSON (JavaScript Object Notation) - A file to store structured data
- Software Requirement Specifications (SRS) - A document that records the requirements expected from the software.
- Decomposition - Breaking down a process into subprocesses
- Deadlock - A situation in which two or more processes, sharing some resources, prevent one another from executing.

1.4. Overview

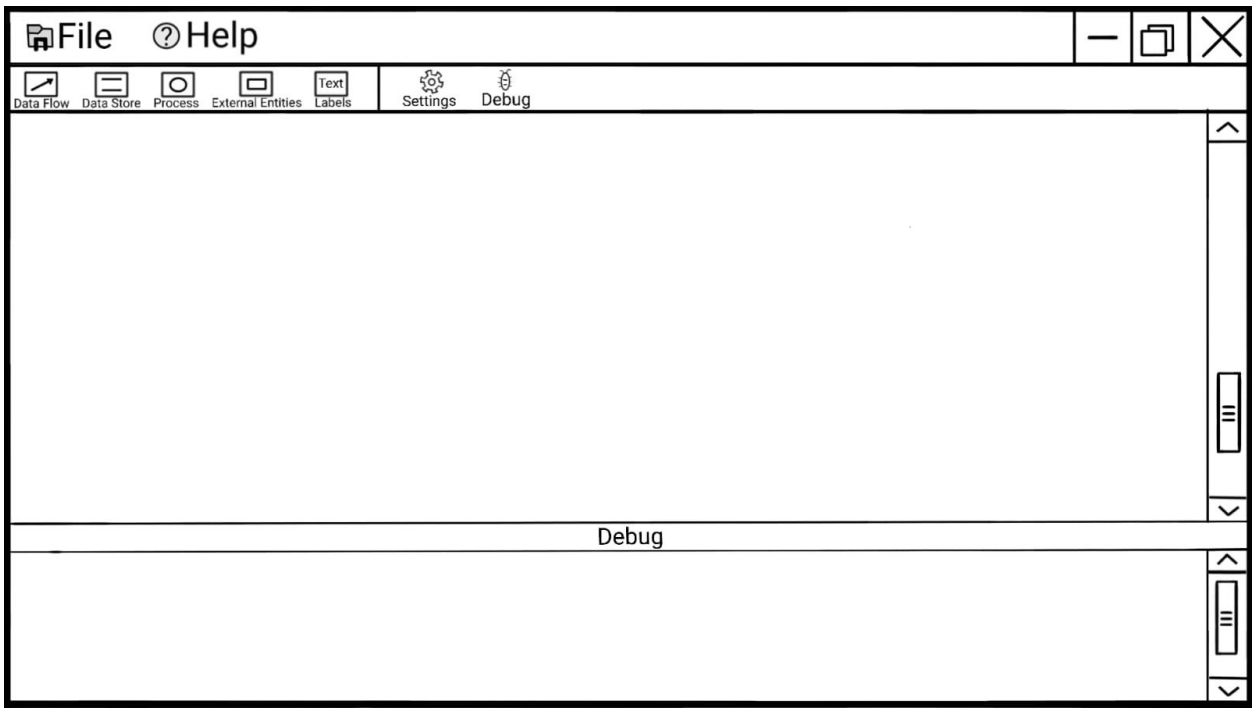
The SRS document is divided into three sections: Introduction, Objectives, and Specific Requirements. In the introduction section, the purpose and scope of the project are listed. The objectives section describes the project's objectives: perspectives, functions, constraints, dependencies, and targeted user characteristics. The specific requirements section records the particular requirements of the product, namely functional and non-functional requirements.

2. Overall Description

2.1. Product Perspective

The software SADesignerLite helps software engineering students and professionals construct DFDs, effectively converting a complex software engineering problem into a simple hierarchical and graphical model. The software represents the different processes and the data flow between them. It automatically checks for different types of errors in a DFD and helps the users debug them. The software also provides specialized functionality to convert DFD into image form or a data dictionary. The users can then print this image or data dictionary.

2.2. Product Functions



- **File Manipulation**: The user can open a new or an existing file and save the changes made.
- **Constructing a DFD**: The software provides different shapes to build various components of DFD, namely data store, process, outputs, and external entities. The user can use arrows to represent the data flow between different parts.
- **Error Handling**: The software checks for different types of errors like balancing errors, unnecessary data flow, data flow between two data stores, and isolated components.
- **Decomposing a DFD**: The software enables the user to decompose DFD processes into sub-processes. The existing data flows are auto-generated in a new workspace, and if they are not utilized, balancing errors are generated. The user cannot edit the lower level

hierarchy before saving and closing the higher level hierarchy. External Entities shall be unavailable.

- **Export**: A DFD is always accompanied by a data dictionary that lists all the data items appearing in the DFD. The “export” option in the file section converts DFD into an image or a data dictionary.
- **Print**: This image and data dictionary can be printed as an image using the print function.

2.3. User Characteristics

- The user must be familiar with basic software engineering concepts.
- The user must exhibit basic proficiency in English.
- There is no user-specific interface.

2.4. Constraints

- The software runs only in Windows operating system.
- It should allow access to modify and create files in the memory.
- It should be allowed to execute .jar files.
- It should allow the user to print the files.

2.5. Assumptions and Dependencies

- It requires Windows 7 or above operating system.
- It requires JDK 8 or above. And Java SE 17.0.2 or above

3. Specific Requirements

3.1. External Interfaces:

- **Hardware Interfaces:** Windows OS
- **Front-end Package:** Java Swing
- **Back-end Package:** JSON Simple

3.2. Functions:

- **Menu Bar:**
    - **Help:** Opens the software documentation for reference on click.
    - **File:**
        - **New:** creates a new file with the extension: .dfd for use. On click, a new empty file is created.
        - **Open:** opens an existing file with the extension supported by the system. The user is prompted to select the file they would like to open on click.
        - **Save:** Save the file with the supported extension. It prompts the user to select the destination folder and saves the file at the specified location on click.
        - **Export:** Generates a data dictionary/image and saves it.
- **Edit:** Resize, annotate, and rotate shapes. The software will assign each shape a system-generated variable name. On click, it opens details of the shapes and allows the user to change details for each shape.
- **Ribbon:**
    - **Data Flow:** on click, it creates a data flow between two processes and datastores and entities.
    - **Data Store:** on click, it creates a data store (represented by horizontal pipelines) that the user can place anywhere on the window.
    - **Process:** on click, it creates a process (as a circle) that the user can place anywhere on the window.
    - **External Entity:** on click, it creates an external entity (as a rectangle) that the user can place anywhere on the window. (Only used in level 0 of the DFD)
    - **Output:** on click, it creates an output (as a parallelogram) that the user can place anywhere on the window.
    - **Labels:** on click, it creates a text field over a shape that the user chooses to name the entity.
- **Debugging:**

    - **Balancing errors:** making sure the external data flow in the lower order hierarchy is connected in the higher-order hierarchy adequately.
    - **Identifying erroneous and unnecessary dataflows.**
        - Data flow between datastore: If a data flow connects two data stores.
        - Unnecessary data flow: If there are two or more paths between two processes irrespective of direction.
    - Identifying isolated components.
- **Decomposition:**
    - A new workspace (window) is created with the system-generated variable name appended to the file name, representing the higher level DFD (The window stays with the same format).
    - The existing data flows are auto-generated in the new workspace; balancing errors will be formed if not utilized.
    - The user cannot edit the lower level hierarchy before saving and closing the higher level hierarchy.
- **Printing:**
    - Generate an image of the DFD and create a print preview of the image.
- **Shortcuts:**
    - Del - deleting a shape
    - Alt+F4 - Close
    - Ctrl+S - Save

3.3. Performance Requirements:

- Time taken for export: < 1 min.
- Time taken for debugging <= 0.1 ms per shape
- Probability of deadlock: < 1%
- Error rate <0.1%
- Maximum storage capacity: proportional to the storage available on the machine.

3.4. Design Constraints:

- Level of DFD <= 6

- The net number of shapes <= 150
- Only one file is open at a time
- The system must save only files without errors
- The system must ask for saving before exit if unsaved.
- Size of workspace = 2048 px x 2048 px
- All objects must remain within workspace

3.5. Software System Quality Attributes

- **Reliability:** The software should be reliable in editing shapes, balancing errors, debugging, and exporting a file.
- **Usability:** The software will be easy to use and not have a steep learning curve. It will satisfy a maximum percentage of people who use it. The aesthetics and design of the software will be well designed and minimalist.
- **Portability:** The software will be easy to install and adaptable to various versions of Windows OS (7 or higher).
- **Maintainability:** The software will be reused for multiple projects and possess reasonable modularity. Updates in certain parts of the software will not affect the rest.
- **Performance Efficiency:** The software will be efficient in exporting DFD models and DD models and will keep export times at a minimum. In addition, the software will minimize stutter during use and keep average performance at an optimal level.