### **SOLVEDOKU**

(SDK)

#### CS 3337 Software Engineering

#### Functional Requirements and Design Document

Prepared by

Kevin Trang

Allen Ruan

Nikita Aggarwal

Brandon Lee

APR 16, 2018

**Table of Contents**

Introduction

* Product Pg.3
* Scope Pg.3
* Context background (DFD Level 0) Pg.4
* Project name architecture Pg.4
* Description and Major Functions of SOLVEDOKU Pg.4
* Hardware and Software Considerations Pg.4
* References Pg.5

Detailed Functional Description of SOLVEDOKU

* Detailed SOLVEDOKU Functional Description. Pg.5,6,7
* Level 1 DFD Pg.5,6,7

SOLVEDOKU Functional Requirements Pg.8,9

SOLVEDOKU Hardware Requirements Pg.9

SOLVEDOKU’s Elements of Implementation

* Introduction Pg.10
* Description of Java Classes Pg.10

# List of Figures

Figure 1-1 Context Diagram (Level 0 Data Flow Diagram)

Figure 2-1 Level 1 Data Flow Diagram

# List of Tables

None.**1.0 INTRODUCTION**

**1.1 Purpose**

The purpose of this document is four-fold:

a) Define a full set of requirements for SOLVEDOKU.

b) Define the design for SOLVEDOKU.

c) Define the implementation of SOLVEDOKU.

The complete definition of all SOLVEDOKU requirements provides the requirements to be used in the subsequent software subsystem documents.

**1.2 Scope**

This documentation was developed as part of a Software Engineering class, CS 3337.

The scope of this document includes the following:

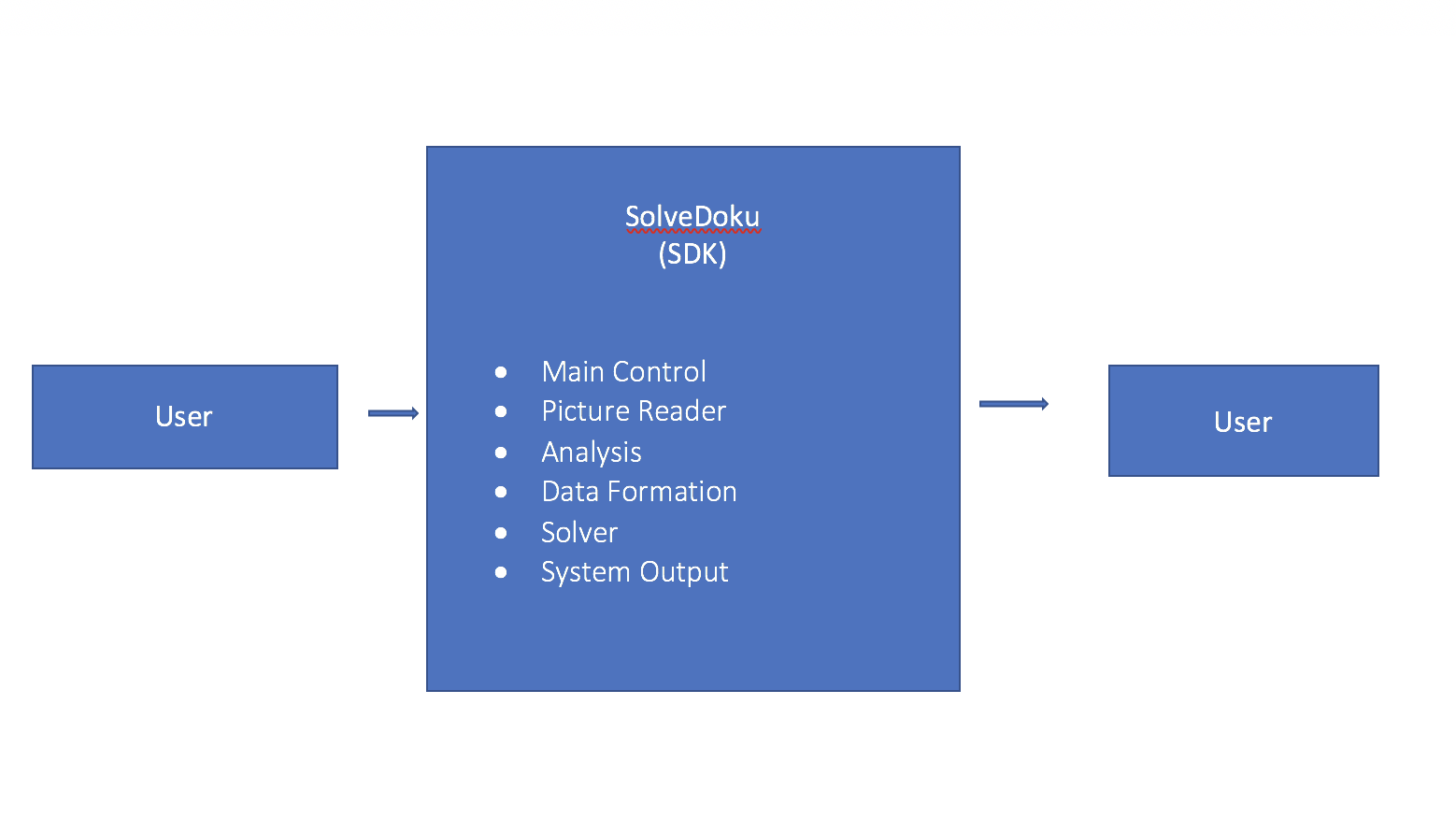
* All functional requirements. These requirements are organized by key SOLVEDOKU functional units shown in DFD 1
* Functional descriptions of each of the modules of SOLVEDOKU.
* General descriptions of hardware necessary for implementation of SOLVEDOKU.

**1.3 Project Name Architecture**

**1.3.1 Context Diagram (DFD Level 0)**

SOLVEDOKU’s architecture is summarized in the Context Diagram (DFD Level 0) given below. A more detailed Functional Description is given in Section 2 of this document.

**Figure 1-1:** **Level 0 DFD**

****

**1.3.2 Description and Major Functions of SOLVEDOKU**

SOLVEDOKU automates the solving process of a Sudoku puzzle. The system offers the ability to upload an image and have it solved using image recognition. SOLVEDOKU recognizes the image and stores all the digits and uses backtracking to solve the 9x9 puzzle. SOLVEDOKU is controlled by a GUI. The name SOLVEDOKU is a combination between the famous Sudoku puzzle and the word solve.

**1.3.3 Hardware and Software Considerations**

SOLVEDOKU requires standard computer hardware. Other software requirements include the latest build of Oracle’s Java Runtime Environment.

**1.5 References**

All references used in the creation of this document are listed below.

1.5.1 OLYMPIA – PowerPoint

1.5.2 OLYMPIA – SRS

1.5.3 J.O.S.E – SRS

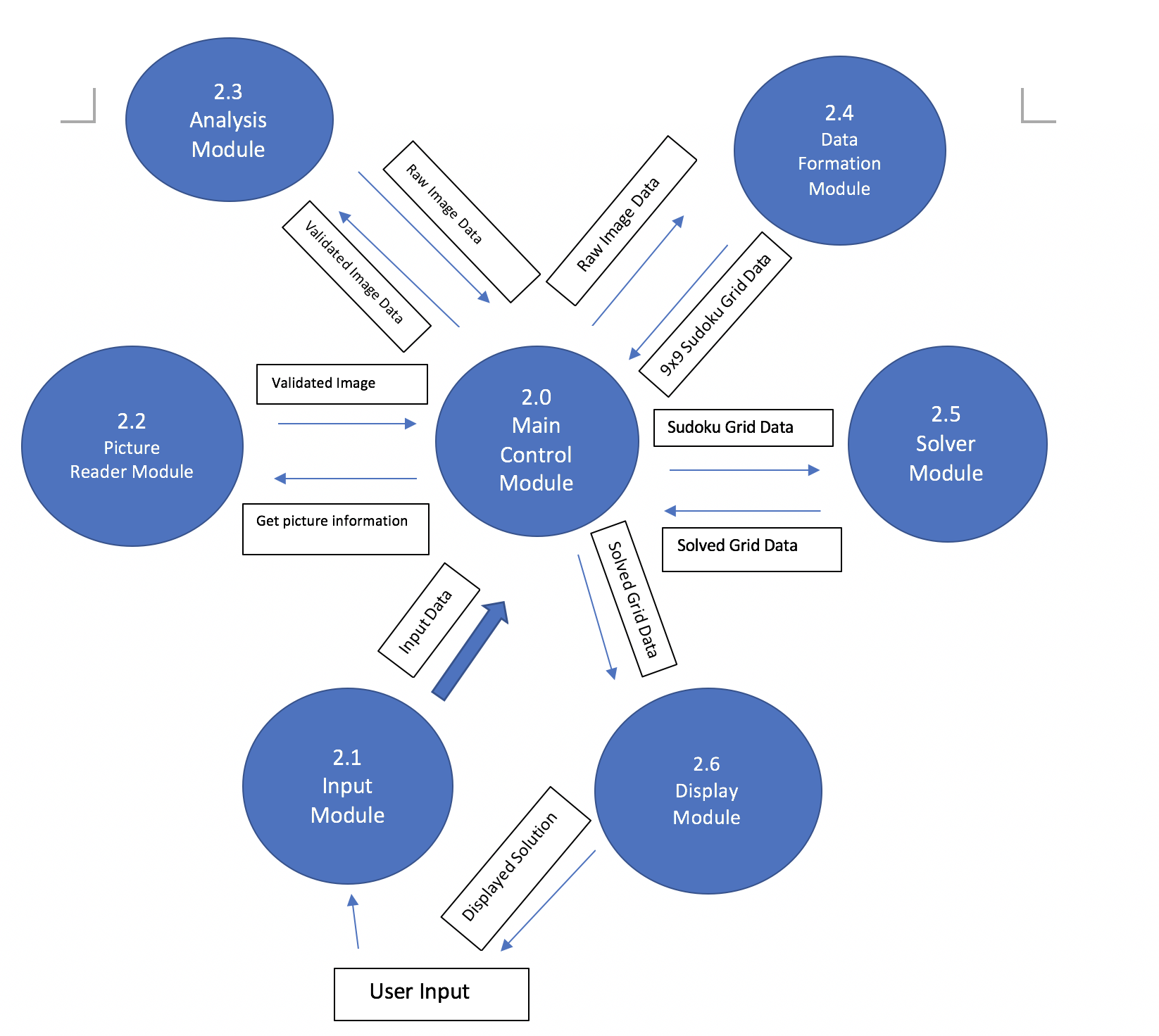
**2.0 DETAILED FUNCTIONAL DESCRIPTION OF SOLVEDOKU**

**2.1 Detailed SOLVEDOKU Functional Description**

The major tool used to design SOLVEDOKU is the Data Flow Diagram, DFD. The rationale for the selection of DFDs as the preferred design tool is its simplicity and versatility. In the future additional tools may be used if a stronger correlation from Design to Requirement to Implementation and Testing is required.

**2.1.1 Level 1 DFD**

SOLVEDOKU’s major functional subunits are shown in the DFD Level 1 shown below:

****

**Figure 2.1: Level 1 DFD**

**2.1.2 Detailed Functional Description of SOLVEDOKU’s Major Units.**

The description of SOLVEDOKU’s major functional units shown in Figure 2.1 follows. Most of SOLVEDOKU’s operations depend on the image provided by the user. Image provided can only be a 9x9 grid.

Input - Module 2.1

The Input Module (IM) allows the user to insert the image file for processing.

Picture Reader - Module 2.2

The Picture Reader Module (PRM) validates the image that is entered. If an invalid picture is entered into the program, it rejects the picture. Also, the grid that is entered must contain 17 digits already pre-entered.

Analysis - Module 2.3

The Analysis Module (AM) analyzes the image’s 17 pre-filled digits. The Analysis Module will take the 17 or more pre-filled digits will be extracted from the image and stored.

Data Formulation - Module 2.4

The Data Formulation Module (DFM) processes the extracted image data and stores the data into a data structure.

Solver Module - Module 2.5

The Solver Module (SM) uses backtracking to fill the blank spaces as random digits first. As it fills the puzzle with random digits, collisions shall occur. Through collisions, the Solver Module shall find the digits that do not belong in certain spaces and reject the digit.

Display - Module 2.6

The Display Module (DM) will take the solution of the solved puzzle. Through the solved puzzle, the Display Module will create a visual representation for the user.

**3.0 SOLVEDOKU REQUIREMENTS**

**3.1 SOLVEDOKU Functional Requirements**

This Section lists SOLVEDOKU’s functional requirements. This section includes the complete set of functional requirements, along with explanations.

|  |  |
| --- | --- |
| **Requirements Related to Design Module 2.2: Input Module** | |
| Requirement No. | Requirement Description |
|  | **Input Module (IM)** |
| 3.1.1 | IM shall process inserted data |

|  |  |
| --- | --- |
| **Requirements Related to Design Module 2.2: Picture Reader Module** | |
| Requirement No. | Requirement Description |
|  | **Picture Reader Module (PRM)** |
| 3.2.1 | PRM shall validate the image file |
| 3.2.2 | PRM shall reject non 9x9 image grids |

|  |  |
| --- | --- |
| **Requirements Related to Design Module 2.3: Analysis Module** | |
| Requirement No. | Requirement Description |
|  | **Analysis Module (AM)** |
| 3.3.1 | AM shall collect Sudoku data from the validated image |
| 3.3.2 | AM shall recognize blank spaces as zeros |

|  |  |
| --- | --- |
| **Requirements Related to Design Module 2.4: Data Formation Module** | |
| Requirement No. | Requirement Description |
|  | **Data Formation Module (DFM)** |
| 3.4.1 | DFM shall store Sudoku grid data in a data structure |
|  |  |

|  |  |
| --- | --- |
| **Requirements Related to Design Module 2.5: Solver Module** | |
| Requirement No. | Requirement Description |
|  | **Solver Module (SM)** |
| 3.5.1 | SM shall use backtracking |
| 3.5.2 | SM shall populate grid using random digits from 1-9 |
| 3.5.3 | SM shall test the random digits against the rules of a Sudoku puzzle |
| 3.5.4 | SM shall reject incorrect digits entered into the puzzle |

|  |  |
| --- | --- |
| **Requirements Related to Design Module 2.6: Display Module** | |
| Requirement No. | Requirement Description |
|  | **Display Module** |
| 3.6.1 | DM shall be able to process the populated data for displaying it |
| 3.6.2 | DM shall be able to create a visual representation for the user’s interpretation |

**3.3 SOLVEDOKU Hardware Requirements**

This Section collects all SOLVEDOKU’s electronic hardware requirements. All hardware requirements are numbered “H – n” where “n” indicates the nth requirement.

H - 1 SOLVEDOKU will run on standard microcomputer hardware

**4.0 SOLVEDOKU’sELEMENTS OF IMPLEMENTATION**

**4.1 Introduction**

This section descries the implementation of the requirements described above. We include brief descriptions of each of the coded classes. Java code files are included separately.

**4.2 Descriptions of Java Classes**

**Tess4JDigit GUI:**

*Tess4JDigit* uses optical character recognition to read the individual images of digits and outputs the integer value of the inserted digit.

**DetectSudoku**

*DetectSudoku* grabs the puzzle and cuts out unnecessary border. This class also crops the puzzle into individual 81digit squares.

**FeatureDetecter**

*FeatureDetecter* detects the digit inside the individual squares. This class then determines if the square contains a digit or not. If not, then this class enters 0 as a substitute for a blank digit.

**Board**

*Board* orients the output as a 9 by 9 grid during testing.

**DancingLinks**

*DancingLinks* is a row and column creator. It creates either a vertical or horizontal strip 9 grids. These grids are combined to create a 9x9 grid for the Board class to process

**Main/Driver**

These two classes create graphical user interface for the user to interact with. It also displays the correct solution to the Sudoku puzzle after it has been solved.