Architecture & Design, B.Tech(CSE) - SD Deptt.,MMEC , MMDU

SuDoku Puzzle Player and Solver

Software Architecture Document

Version <1.00>

Revision History

Date	Version	Description
30-04-2020	1.00	Software architecture document for applying MVC architecture on the existing Sudoku code.

Table of Contents

Scaling - Up (PPP Tool)

1. Introduction

This project Sudoku deals with solving a Sudoku puzzle either form taking the values directly form the user or form a text file. At the beginning we have given a combined code. We made research on MVC architecture and divided the code into three main classes Model, view and controller. Where view deals with the UI part, the model deals with the business logics and the controller sends requests from view to model class.

1.1 Purpose

Basically a SUDOKU Puzzle is defined as a logic based or even we can say that it's a number-placement puzzle. For suppose if we consider a 9*9 grid with digits in such a way that each column and each row and each of the nine 3×3 grids that make up the larger 9×9 grid contains all of the digits from 1 to 9.

The main purpose of this game is:

- It boosts logical thinking
- It improves memory and recalling
- It develops quick thinking skills
- It develops vision

1.2 Scope

The game SUDOKU will be having all the records that you will be performing like the level you choose easy, medium or difficulty. We can also make our own sudoku. We can even make changes within them.

Playing SUDOKU is really a difficult job and also it needs a lot of mental thinking, remaining and recalling.

1.3 Definitions, Acronyms, and Abbreviations

Definition of SUDOKU is a puzzle in which players insert the numbers one to nine into a grid consisting of nine squares subdivided into a further nine smaller squares in such a way that every number appears once in each horizontal line, vertical line, and square.

MVC-Model View Controller.

UI- User Interface.

Business Logic - part of the program that encodes the real-world business rules that

determine how data can be created, stored, and changed.

1.4 References

MVC: https://www.tutorialspoint.com/struts_2/basic_mvc_architecture.htm

Advantages of MVC: http://siyainfo.com/2017/01/16/top-6-important-benefits-mvc-architecture-web-application-development-process/

Disadvantages of MVC: https://www.interserver.net/tips/kb/mvc-advantages-disadvantages-mvc/

Design view: http://www.chambers.com.au/glossary/design_view.php

1.5 Overview

The first section of the document is the introduction containing the purpose, scope and series of definition of terms. Section 2 includes the overview of the system together with the description of its basic functionality. The overall architectural representation is described in section 3. Section 4 is devoted to the description of each architectural component like use-case view and design view. It includes what the component is, what is the purpose and what it does. The size and performance are also mentioned along with system quality issues and concerns.

2. Architectural Goals and Constraints

According to the project, there are following constraints:

1. The complexity is high to develop the application using this pattern.

2. Understanding the flow of application is difficult. It is difficult to implement and is not suitable for small level applications.

3. Its development and final deployment are a difficult task.

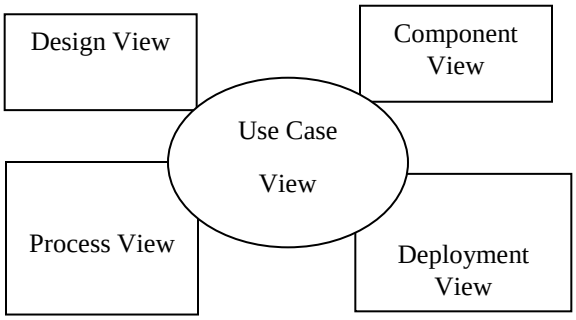
Architectural goals:

1. The code should be divided into three classes Model, view and control.

2. Then View class contains all the UI part of the application, Model class contain the business logic and problem-solving algorithms and the controller class handles the user requests.

3. Architectural Representation

3.1 Architectural Views



Use Case View: The main purpose of the use case view is to define main drivers of the system, which are the system requirements.

Design View: this view contains any system definitions as well as class and object diagrams which depict the services that the system will provide to its end-users.

Process View: this view will display the processes that form the systems' mechanism. These will be represented as collaboration, sequence, and activity diagrams.

Component View: this view will include system and user interface specifications, meaning, the different components that make up the system.

Deployment View: this view will depict how the different systems' hardware nodes will come to life together as well as how each of the hardware nodes will be installed and deployed.

As seen in the above diagram, requirements, or the use case view, are the main driving force to this and any software architecture system.

3.2 Architectural Design Patterns

MVC architecture

We have chosen MVC architecture to design the code. In MVC pattern, application and its development are divided into three interconnected parts, Model, view and controller. The advantage of this is it helps in focusing on a specific part of the application name, the ways information is presented to and accepted from, the user. This helps the application to be maintained in three parts which will be easy to find out the errors in the code. In the model class all the data and the business logic are maintained, in the view class the user interface is done and the user will interact to the application here and in the controller class the request will be send to the model form the user via view class.

MVC Regarding the Sudoku:
Model

In the model class a getter is established which acts as a reference to the view class, we will have a method for checking the input file weather the file is valid or not, we will have the routines establishTheSuDoKuPuzzle() which passed in parameter to define the puzzle from an input file, solveTheSuDoKuPuzzle() automatically solves the SuDoKu puzzle, setPuzzleValue() used when the controller needs to update the current board due to input from the user changing one of the combo-boxes.

View

In the view class we will have attributes for width of the window, height of the window, pane which is root of the GUI elements that form the UI, a comboBoxSelector which will display the puzzle in the screen. In the view() method which is fully parametrized it has parameters reference to the WindowManager parent, width of the window, height of the window, reference to the Controller Object, reference to the Window Pane object. The setupWindowDecoration() will decorate the window with rectangles. The method setMsgInvalid (String) allows others to set and clear the message field value that is displayed by the window manager. The routine resetSelector (int, int) used by the model to reset a comboBox back to the defaults state. The setSingleSelectItem (int, int, int) is used by the model to change a combo box. The getter getComboBoxSelectors () returns a reference to the matrix of comboBoxes so the rest of the systems can access them and update them as needed.

Controller

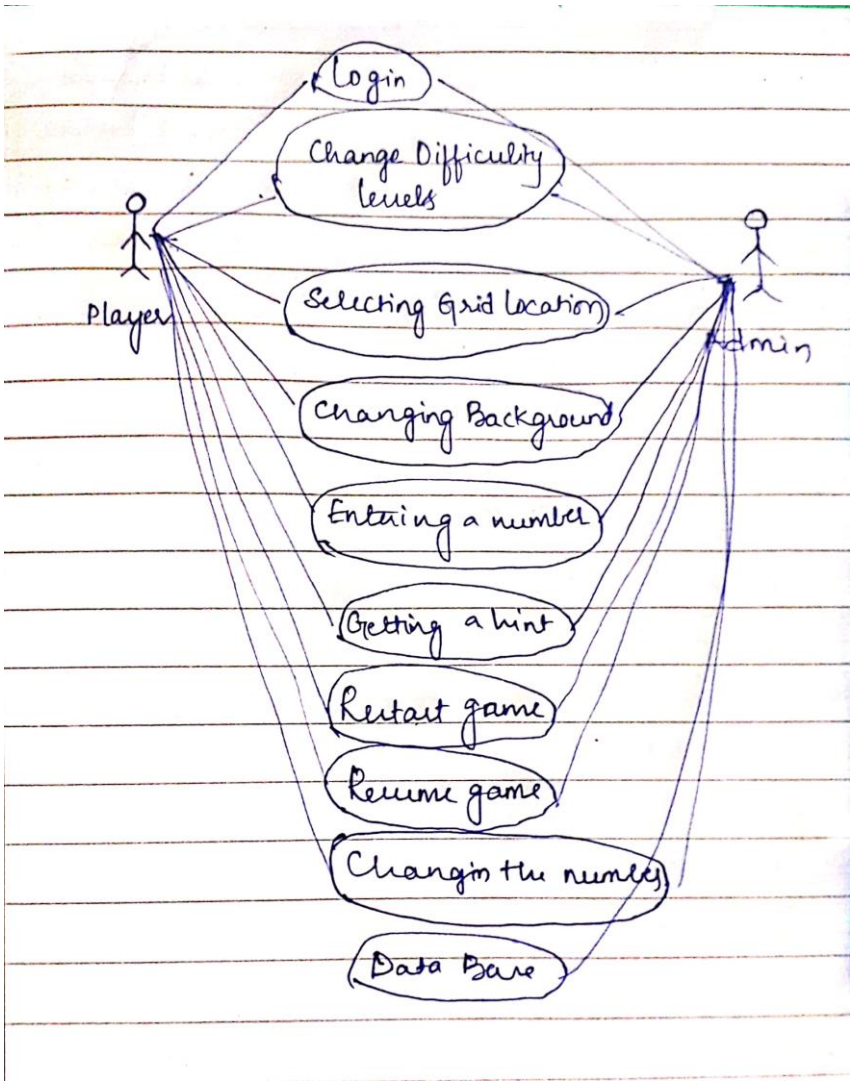
In the controller class the Controller (Model) constructor establishes this controller and gives it a reference to the Model that this controller is supposed to use. The setView (View) is a setter used to connect this controller to the View that manages the user interface. The solveTheSuDoKuPuzzle () class gets the input form the view class and gets it solved form the model class, this method displays message weather the puzzle is solved or not. The method solveTheSuDoKuPuzzle () is setting up the user interface array of comboBoxes to reflect the solution that was found. The performSelection (ActionEvent) is used when the user changes a comboBox select list item. This is not used when the user asked the application to solve the puzzle.

These are the key benefits:

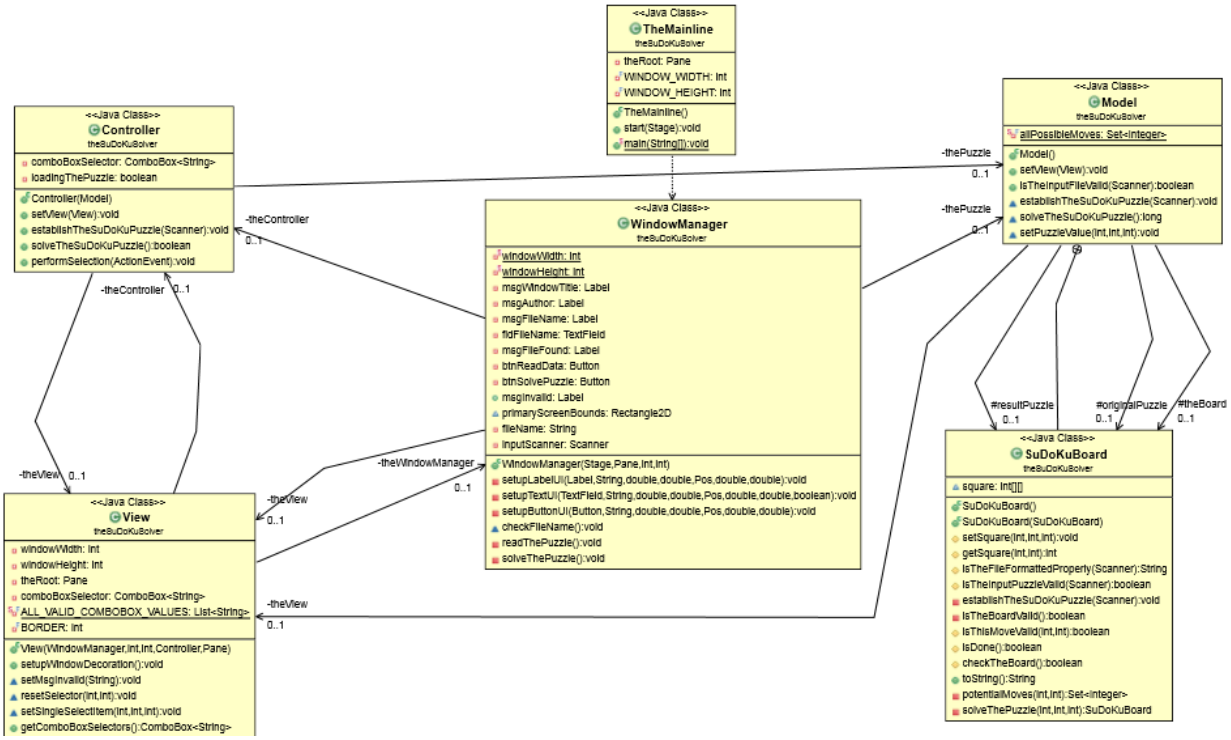
- MVC architecture supports rapid and parallel development. With MVC, one programmer can work on the view while other can work on the controller to create business logic of the web application. The application developed using MVC can be three times faster than application developed using other development patterns.
- MVC also supports asynchronous technique, which helps developers to develop an application that loads very fast.
- Modification does not affect the entire model because model part does not depend on the views part. Therefore, any changes in any of the classes will not affect the entire architecture.
- In the MVC Model, you can create multiple views for a model. Code duplication is very limited in MVC because it separates data and business logic from the display.

4. Architectural View Decomposition

4.1 Use-Case View



4.2 Design View



5. Size and Performance

Size:

The size of the project can be managed using the various types of data structures to reduce the memory space for storing the data, which also increases the performance of the project. How much space the program is occupying while running?

Performance:

How fast the system is able to solve the puzzle and provide the solution in correct manner following all the algorithms.

They have to be quick enough to make changes on run time depending on the file containing puzzle.

6. Quality

Reliability:

The code must solve the puzzle as soon as we click on the solve button.

Performance:

How fast the system is able to solve the given puzzle?

How accurately the system is displaying the error messages?

Usability:

All the labels should have description in order to understand them.

Error messages should be displayed in a appropriate way.

Integrity:

The developer can only be able to change the interface of the tool.

Reusability:

Use the inheritance concept to achieve the code reusability.

7. Bibliography

Frank Armour, Granville Miller. 2000. Advanced MVC modelling Modeling: Software Systems. Upper Saddle River, NJ. 425 pages.

Dean Leffingwell, Don Widrig. 2001. Managing Software Requirements: MVC architecture, IN. 491 pages.

James Rumbaugh, Ivan Jacobson, Grady Booch. 2000. MVC Reference Manual. Massachusetts. 550 pages.