# Software Design Description

Doner

## Table Of Contents

1 Introduction 3

1.1 System Objectives 3

1.2 Hardware, Software, and Human Interfaces Section 3

1.2.1 Hardware Interfaces 3

1.2.2 Software Interfaces 3

1.2.3 Human Interfaces 3

2 Architectural Design Section 4

2.1 Major Software Components Section 4

2.2 Major Software Interactions Section 4

2.3 Architectural Design Diagrams Sections 4

3 CSC and CSU Descriptions Section 6

3.1 Detailed Class Descriptions section 6

3.1.1 Tools (Tools.h and Tools.cpp) 6

3.1.2 texture (structure located inside Tools.h) 6

3.1.1 Doner (Doner.h, Doner.cpp and Doner.rc) 6

3.2 Detailed Interface Descriptions Section 7

3.3 Detailed Data Structure Descriptions Section 7

3.4 Detailed Design Diagrams Section 7

4 Database Design and Description Section 8

# 1 Introduction

This document presents the architecture and detailed design for the software for the Doner project. This project performs functions that allows users to create visual novels without programming knowledge and allow them to view changes instantly.

## 1.1 System Objectives

The objective of this application is to provide a game engine that serves solely for visual novels. It will provide a frame of visual novel and allows users to insert and edit texts, images and audios for it. This project will save what's been changed, and a export function is provided for the user to get their own game.

## 1.2 Hardware, Software, and Human Interfaces Section

### 1.2.1 Hardware Interfaces

This project requires the user to have a computer, a monitor, and both a mouse or a keyboard. Downloading this program requires internet connection, but using this program does not.

### 1.2.2 Software Interfaces

This project runs on the basis of Windows32 system and Windows vista, it requires OpenGL, c++14 compiler, and directX 11.

### 1.2.3 Human Interfaces

This project includes input from keyboard, and can be controlled with either a mouse or a touchpad.

# 2 Architectural Design Section

This application will have two part, the engine part and the game part. By creating a project, this application will create a program that could run as a visual novel game. The user will use the engine to modify a file that saves settings and data, and the game could read the data file to perform the game the user created.

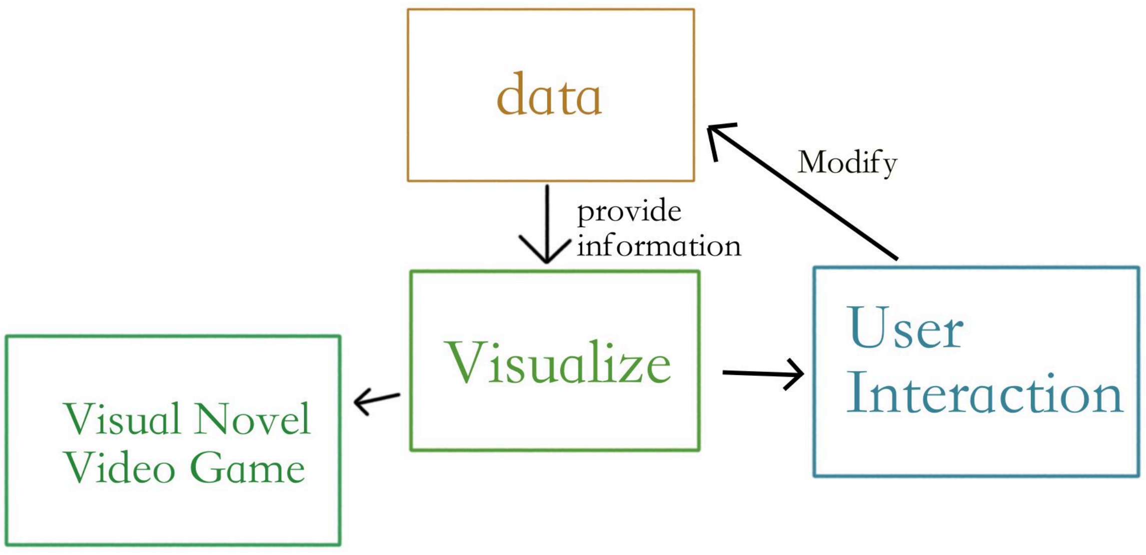
## 2.1 Major Software Components Section

This application should have the ability to read and write into files, as presenting their contents in the window. This application should also accept user's input on editing files.

## 2.2 Major Software Interactions Section

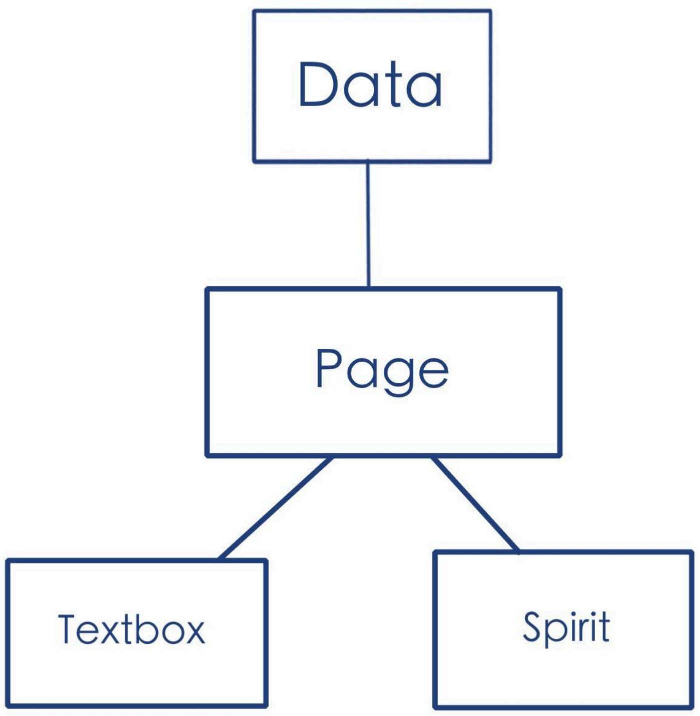
In this application, the function about file, including reading, writing and storing data, should be in one segment. It should pass the data it reads to the modification segment, as these functions are presented to the user with the visualizing section which collaborate with ImGUI. On the other hand, the data that's being read could also be accepted by the section that presented the data in the form of game.

## 2.3 Architectural Design Diagrams Sections

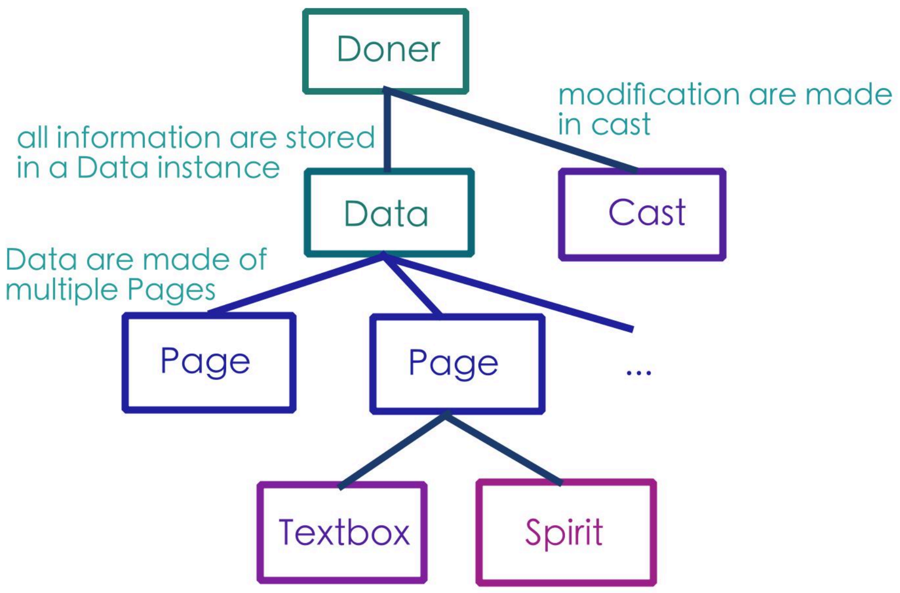


Component Diagram

说明



Data Structure Diagram


Doner
modification are made in cast
all information are stored in a Data instance
Data
Cast
Data are made of multiple Pages
Page
Page
...
Textbox
Spirit

说明

## 3 CSC and CSU Descriptions Section

The components of this CSCI are showed in the graph above. More explanations will be made below.

## 3.1 Detailed Class Descriptions section

### 3.1.1 Tools (Tools.h and Tools.cpp)

Tools is a helper class containing some methods that can be generally used in other classes.
Method:
    static void setBackground(std::string background_name)

### 3.1.2 texture (structure located inside Tools.h)

### 3.1.1 Doner (Doner.h, Doner.cpp and Doner.rc)

Doner is the main file for the application to run. Its main use is to communicate with the system to create a window which supports all other functions to run on it.

Field:
    std::string PathLoc
        The address of the application
    std::string Path
        The address of the application's "project storage" file
    std::string DefaultBackground
        The address of the background picture that will be show when there is no user's project opened
    data gameData
        The data of the user's project, empty in default
    std::vector<texture> textureList(20)
        The list of texture the application is loading every frame. It is set to 20 because it is a fair amount that is either too little for the user or too much and take more memory than needed.
    static ID3D11Device* g_pd3dDevice
    static ID3D11DeviceContext* g_pd3dDeviceContect
    static IDXGISwapChain* g_pSwapChain
    static UINT g_ResizeWidth, g_ResizeHeight
    static ID3D11RenderTargetView* g_mainRenderTargetView
        The above are some global variable for the window

Method:

## 3.2 Detailed Interface Descriptions Section

## 3.3 Detailed Data Structure Descriptions Section

## 3.4 Detailed Design Diagrams Section

## 4 Database Design and Description Section