Project Dreamer / Item Module

Design Document

Change History

Version: 0.1
Modifier: Joshua Griffis
Date: 15/02/2022
Description of Change: Design Document started

Version: 0.2
Modifier: Joshua Griffis
Date: 12/03/2022
Description of Change: Modified document to copy modules and created better visuals for the Mid-Level Design, and Detailed Design sections.

Version: 0.3
Modifier: Joshua Griffis
Date: 13/03/2022
Description of Change: Added Process View and a better Detailed View

Version: 0.4
Modifier: Joshua Griffis
Date: 15/03/2022
Description of Change: Update Process View

Introduction

This document describes the architecture and design for the Project Dreamer application being developed by Radical Dreamers. Project Dreamer is a Third Person Role Playing Game where you explore the dream world and interact with its inhabitants and fight bad dreams.

The purpose of this document is to describe the architecture and design of the Project Dreamer application in a way that addresses the interests and concerns of all major stakeholders. For this application the major stakeholders are:

- Developers – they want an architecture that will minimise complexity and development effort.
- Project Manager – the project manager is responsible for assigning tasks and coordinating development work. He or she wants an architecture that divides the system into components of roughly equal size and complexity that can be developed simultaneously with minimal dependencies. For this to happen, the modules need well-defined interfaces. Also, because most individuals specialise in a particular skill or technology, modules should be designed around specific expertise. For example, all UI logic might be encapsulated in one module. Another might have all game logic.
- Maintenance Programmers – they want assurance that the system will be easy to evolve and maintain into the future.

Design Goals

The goals we decided on for the design of the Items was an easily customizable/creatable system where the design team can easily input the variables for the Items. Items are mostly used to store the data we will need for future checks and uses.

System Behaviour

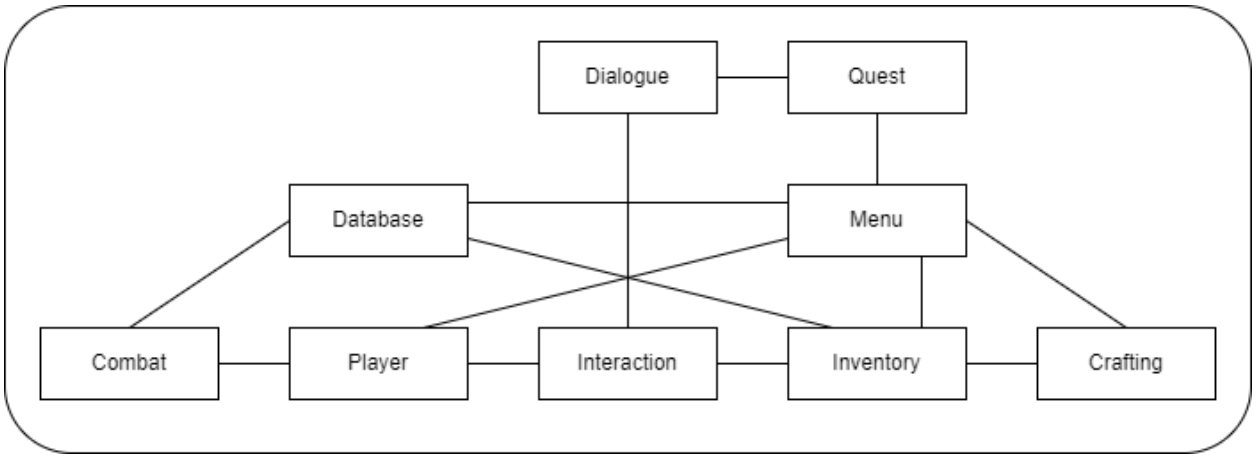
The use case view is used to both drive the design phase and validate the output of the design phase. The architecture description presented here starts with a review of the expected system behaviour in order to set the stage for the architecture description that follows. For a more detailed account of software requirements, see the requirements document (GDD).

An Item is something the player will collect and use alongside the story of Project Dreamers. It is mostly used as a reference for the variables attached to it. The Database Manager will pull the data from the item when it is required and will use those variables to implement different effects like healing characters, equipping the item, or using it on enemies to deal damage.

Logical View

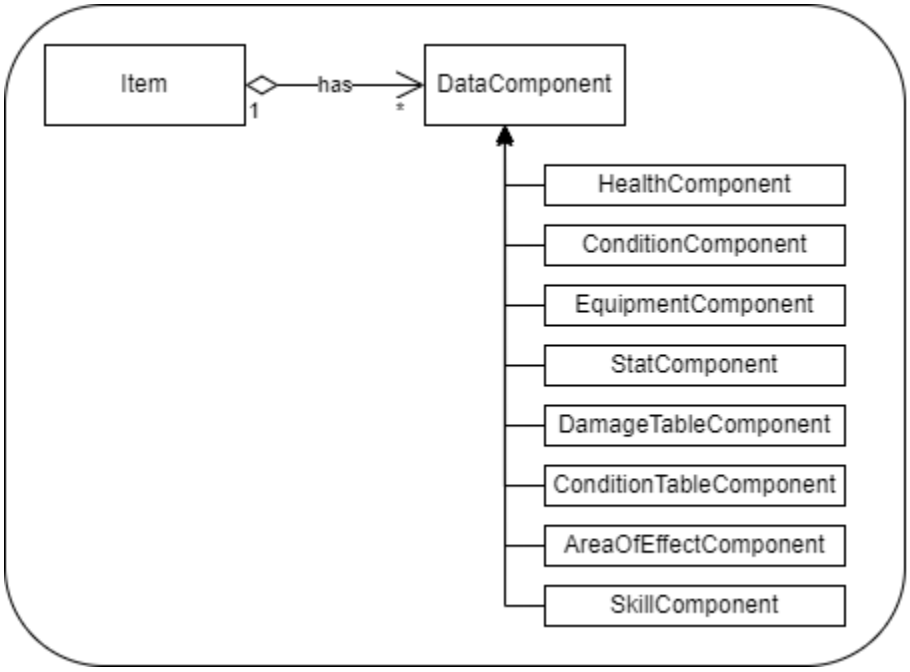
High-Level Design (Architecture of the Entire system)

The high-level view or architecture consists of 9 major components:



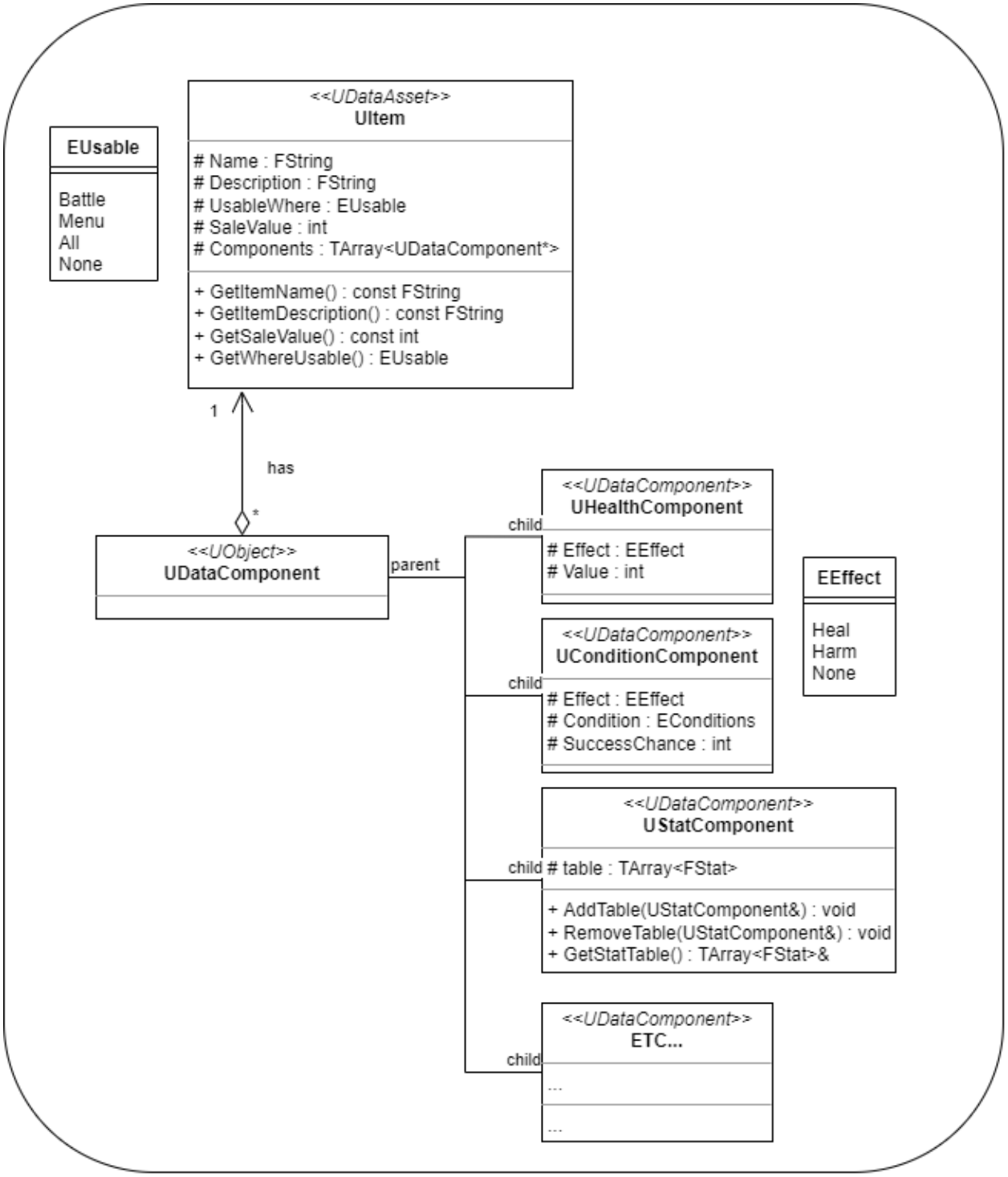
- **Player:** is the main control over the character in the world and allows the user to interact with the world.
- **Database:** stores the data needed for many features to work.
- **Interaction:** handles the objects the player can interact with like talking to characters, opening chests, opening doors, etc.
- **Dialogue:** is responsible for handling the flow of conversations and displaying dialogue to the user.
- **Inventory:** manages the items picked up by the player and money stored. Allowing the player to use them at a later point.
- **Menu:** allows interaction with some modules and features.
- **Combat:** allows interaction between enemies and player characters as they fight. Controls the flow of battle and which battler may act.
- **Quest:** allows the user to have stored data for quests and be able to receive rewards upon completion.
- **Crafting:** allows the player to combine multiple items together to create new items

Mid-Level Design of Item Module

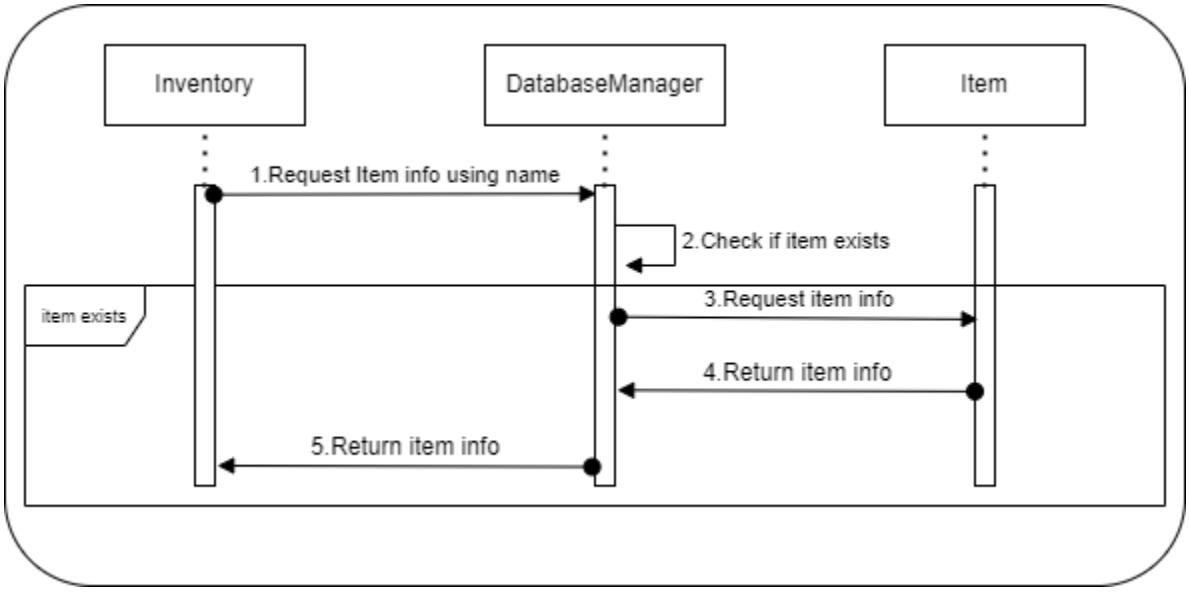


- The Item has a few variables attached to it by default but has an array of components it can have
- You can put whatever Components you want in your design of an Item

Detailed Class Design of the Item Module



Process View of the Item Module



When requested, the database will retrieve the information on an item for us using the name of the item. This is able to happen because all items in Project Dreamer have unique names and no item has the same name as another item.

Use Case View

Setting up an Item

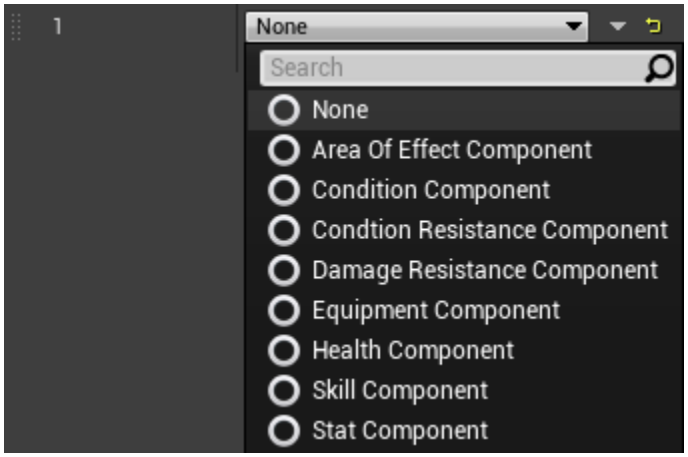
An item will have 5 major variables:

- The **Name**
- The **Description**
- Where it can be **Used**
- How much its **Sale Value** is
- The **Component** list

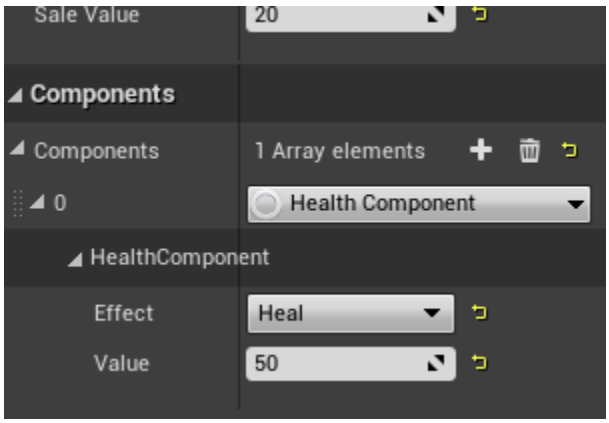
The following images are to show an example of the Item creation process.

- We have named it the Item Potion.
- We have given a description for the item, usually explaining what the item does.
- Then we declare where the Item can be used, Menu, Battle, All, or None. For this Item we have said it can be used anywhere.
- The last value here is how much the item would sell for in a shop.

We can then decide on other features it may have:



We then get a list of possible components to add to the Item. For this example we will add the Health Component:



- We have determined if the item will heal or injure the target.
- We have then set the value it will change it by

With this basic process, any of our designers should be able to create an item they envision.