System and Software Architecture Description (SSAD)

<Project Name>

<Team number>

<Team members and roles>

<Date>

Version History

Date	Author	Version	Changes made	Rationale
08/25/05	PA	2.0	• Original template for use with Instructional ICM-Sw v1.0	• Initial draft for use with Instructional ICM-Sw v1.0
05/22/09	SK	2.1	• Embedded description in each Table	• To be consistent with ICM EPG template set standard V2.1
10/04/16	BO	2.2	In Technology-Independent Model Design • Added conceptual domain model • Separated "Process Realization Diagram" into "Robustness diagram" and "Sequence Diagram"	• For research data collection

Table of Contents

Table of Tables

Table of Figures

1. Introduction

1.1 Purpose of the SSAD

<<Identify the objectives of this document. >>

1.2 Status of the SSAD

<<Briefly describe about status of this document, key differences from previous version.>>

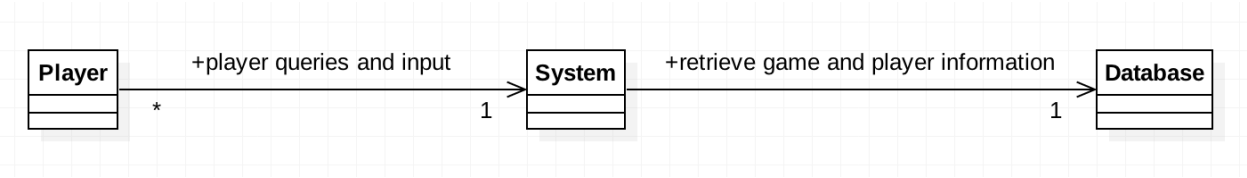
2. System Analysis

2.1 System Analysis Overview

The purpose of this system is to play a game on your mobile device called Spy Fall. This game will be played on your mobile device so that users can play with their friends whenever they want without having to need the actual physical game with them or putting themselves at a desktop to play a web application.

2.1.1 System Context

Figure 1: System Context Diagram



Player: The player is the person who is interacting with the game and will be the one sending information over to the system and then the system will transfer that information to the database.

System: The system is an android application, the systems job is look nice on the outside while hiding all the complicated queries from the player so that the user can focus on having a fun time playing the game, for example making a room for the people joining and starting and ending the game.

Database: The database that we will be using is MySQL database, which will store all the player data and game data. We will call on the database for login information for the player and room information so that the player can join the game.

Table 1: Actors Summary

Actor	Description	Responsibilities
Player	The User	Inputting login information and playing the game
System	The application	To process user input to the database and fetch back information
Database	MySql Database	Sync with the application to send and store back information for the plaer

2.1.2 Artifacts & Information

Figure 2: Artifacts and Information Diagram

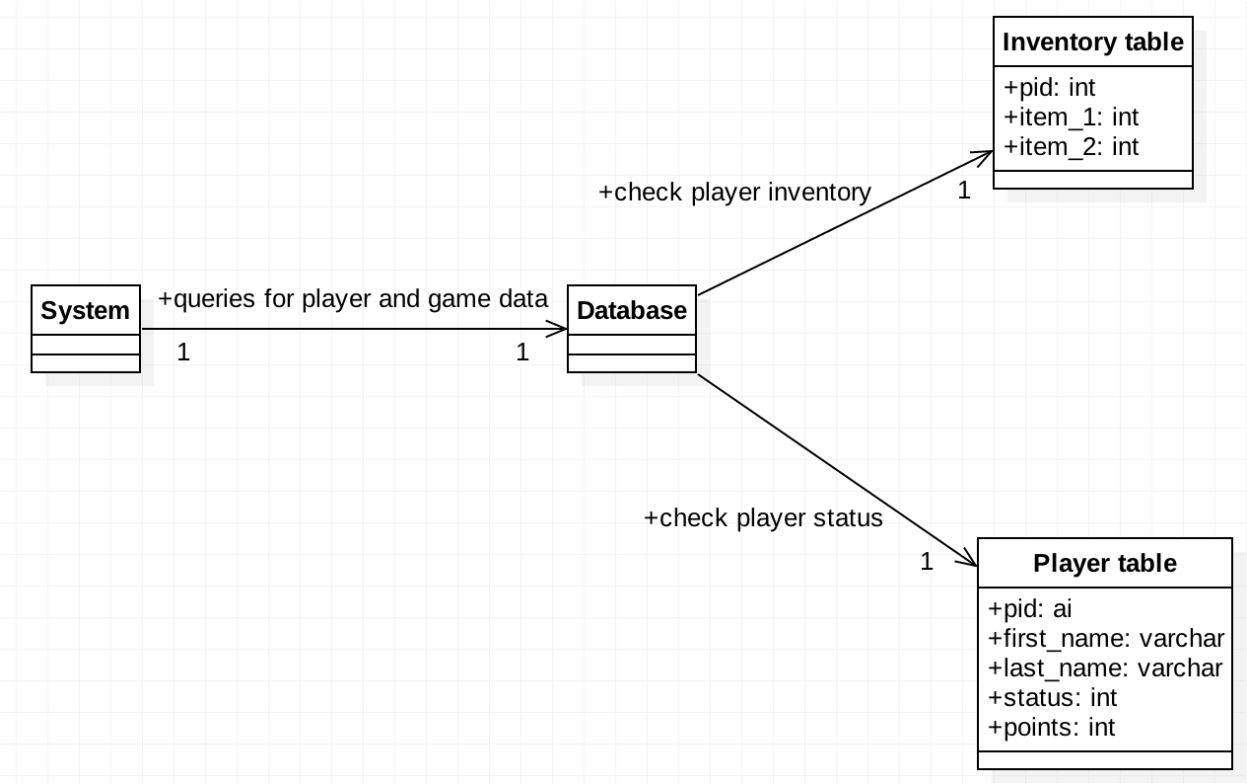


Table 2: Artifacts and Information Summary

Artifact	Purpose
Player table	The player table will keep track of the players username and any game information that we decide to add in for the player
Inventory table	The inventory table will store and keep track of items the player may obtain during their play through of the game

2.1.3 Behavior

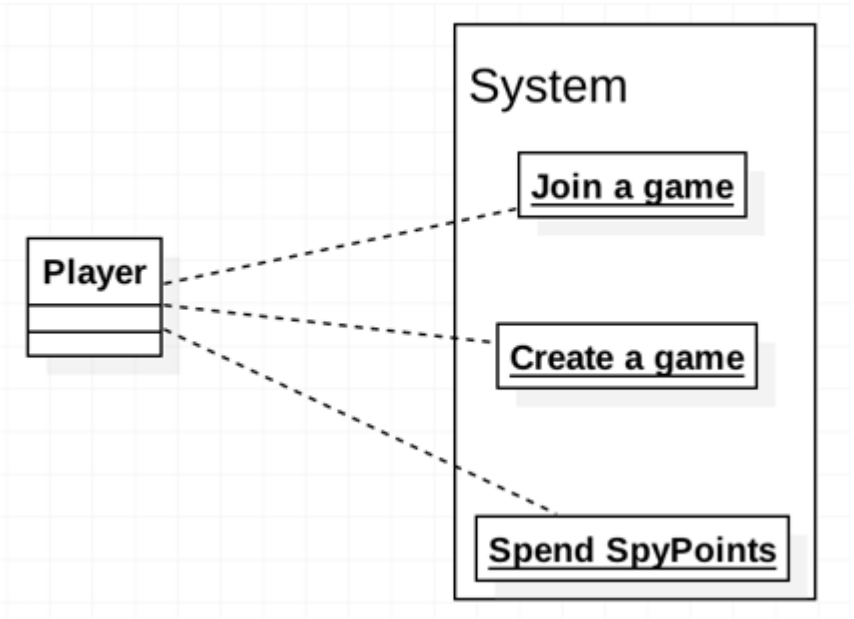


Figure 3: Process Diagram

2.1.3.1.1 Player logs in and plays a game

2.1.3.1.2 Creates or joins a game

Table 3: Logs in and joins/creates a game

Identifier	Logs in to the game
Purpose	For the player to join and create a game
Requirements	The player to have logged in
Development Risks	The database may not be able to store a lot of information if a lot of people play
Pre-conditions	The player logged in
Post-conditions	The database is still running

Table 4: join a game

Seq#	Actor's Action	System's Response
1	Player enters login information	Displays a create and join game button
2	Player enters game room number to join	System looks for the game room and If a match is found enters the game
…		

Table 5: Enters wrong room number

Seq#	Actor's Action	System's Response
1	Player pushes join game	Ask for room number
2	Enters code	After searching for the room and no room is found displays an error message and puts the user back into the create/join screen
…		

3. Technology-Independent Model

3.1 Design Overview

3.1.1 System Structure

<< This section should contain

- a conceptual domain model
- a UML hardware component class diagram
- a UML software component class diagram
- a UML deployment diagram
- If necessary, a class diagram for the system's supporting software infrastructure
- and descriptions of the hardware components, software components, and, if necessary, the supporting software infrastructure components of the technology/platform-independent system architecture

More information and example can be found in **ICM EPG> Task: Define Technology-Independent Architecture >>**

<<Conceptual Domain Model>>

Figure 4: Conceptual Domain Model

<<Hardware Component Class Diagram>>

Figure 5: Hardware Component Class Diagram

<<Software Component Class Diagram>>

Figure 6: Software Component Class Diagram

<<Deployment Diagram>>

Figure 7: Deployment Diagram

<<Optional: Supporting Software Infrastructure Diagram>>

Figure 8: Supporting Software Component Class Diagram

Table 6: Hardware Component Description

Hardware Component	Description

Table 7: Software Component Description

Software Component	Description

Table 8: Supporting Software Component Description

Support Software Component	Description

3.1.2 Design Classes

This section should contain:

- UML class diagrams showing all the boundary, entity, and control classes in the design of the system being developed
- and a description of each class in the diagram

More information and example can be found in **ICM EPG> Task: Define Technology-Independent Architecture** >>

3.1.2.1<Classes n>

<<Design Classes Class Diagram>>

Figure 9: Design Class Diagram

Table 9: Design Class Description

Class	Type	Description

3.1.3 Process Realization

<< This section shows how the proposed architecture can be realized by conducting robustness analysis and constructing sequence diagrams. More information and example can be found in **ICM EPG> Task: Define Technology-Independent Architecture** >>

<<Robustness Diagram>>

Figure 10: Robustness Diagram

<<Sequence Diagram>>

Figure 11: Sequence Diagram

3.2 Design Rationale

<< This section should contain an explanation of how/why the architecture/design described in previous sections was chosen. More information and example can be found in **ICM EPG> Task: Define Technology-Independent Architecture** >>

4. Technology-Specific System Design

<< Once you know specific technology that you team is going to use, design the system and software architecture and document them in this section. >>

4.1 Design Overview

4.1.1 System Structure

<<Hardware Component Class Diagram>>

Figure 12: Hardware Component Class Diagram

<<Software Component Class Diagram>>

Figure 13: Software Component Class Diagram

<<Deployment Diagram>>

Figure 14: Deployment Diagram

<<Optional: Supporting Software Infrastructure Diagram>>

Figure 15: Supporting Software Component Class Diagram

Table 10: Hardware Component Description

Hardware Component	Description

Table 11: Software Component Description

Software Component	Description

Table 12: Supporting Software Component Description

Support Software Component	Description

4.1.2 Design Classes

4.1.2.1<Classes n>

<<Design Classes Class Diagram>>

Figure 16: Design Class Diagram

Table 13: Design Class Description

Class	Type	Description

4.1.3 Process Realization

<<Process Realization Diagram>>

Figure 17: Process Realization Diagram

4.2 Design Rationale

5. Architectural Styles, Patterns and Frameworks

<< Describe any implementation architecture styles (e.g. the Prism style and 3-tier architecture), patterns (e.g. pipe-and-filter and client-server), or frameworks (e.g. Java and CORBA) used to describe the system architecture. >>

Table 14: Architectural Styles, Patterns, and Frameworks

Name	Description	Benefits, Costs, and Limitations