IdentiSky Architectural Design

Group 2 for SFWR ENG 3A04

Alex Guerrero, guerreap, 1133763
Jabrayil Malikov, malikoj, 1302641
Matt Franceschini, francemj, 1310437
Sam Hamel, hamels2, 1321692
Vicky Bilbily, bilbilv, 1317465

March 7th, 2016

1 Introduction

1.1 Purpose

a) The purpose of this documents is to show the architecture of the system and how it will be structured.

b) This document is to be used as a reference for the development team as well as the professor and teaching assistants of SFWR ENG 3A03.

1.2 System Description

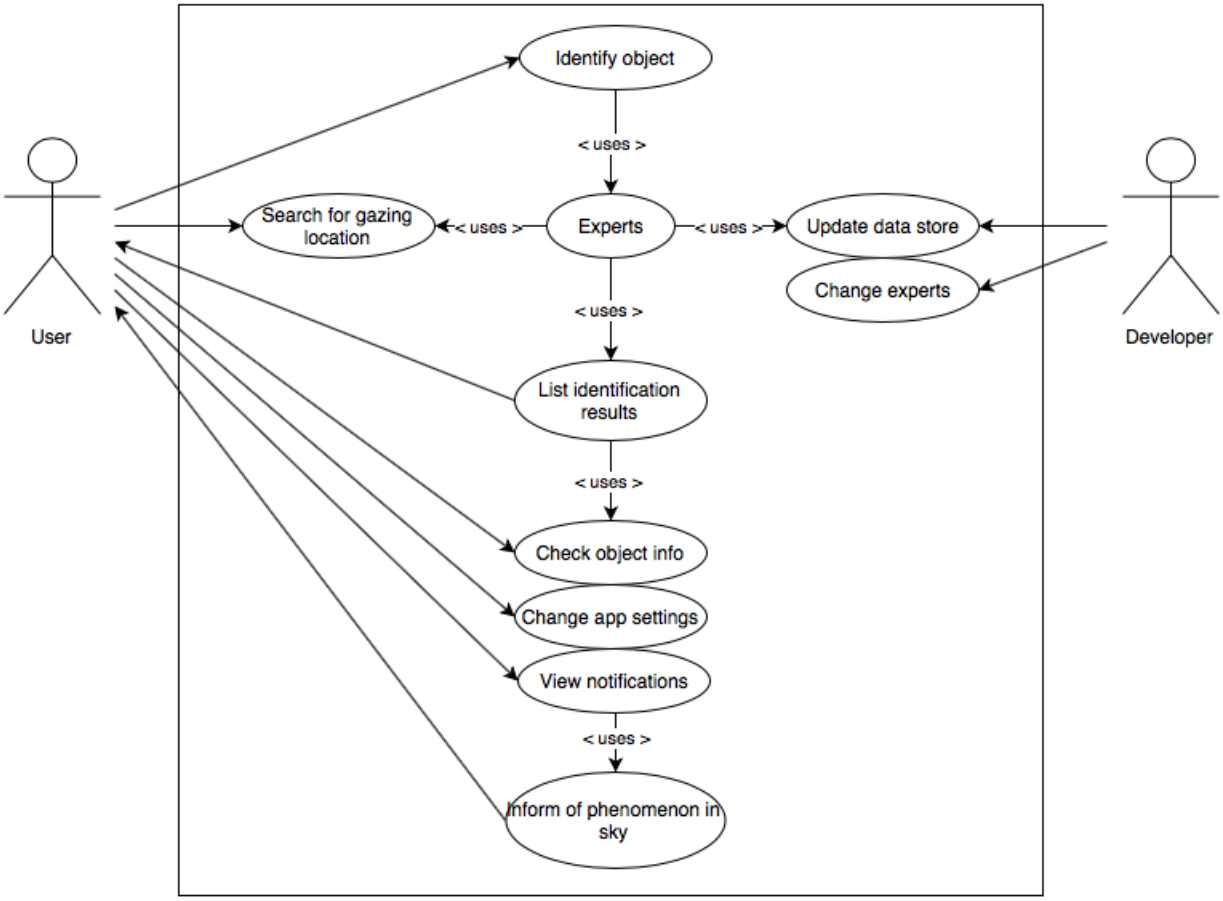
a) IdentiSky is a mobile application that identifies objects in the night sky. These objects are limited to constellations and their associated stars, planets, planes, and noteable satellites. Search results are based on the user's current location, date, time, and the user's answers to questions queried by the application.

1.3 Overview

a) Included in this document will be a couple of diagrams, one will model the functionality of system using actors and use cases. The other will outline the use cases. There will also be an overview of the overall architectural design.

b) The first section will provide an brief overview of the entire document. Section 2 will provide a use case diagram for the application. The next section(section 3) will have an analysis class diagram for the application. In section 4, will be an overview of the overall architectural design of your application. Section 5 will contain the CRC cards.

2 Use Case Diagram



Identify object: The user wants to identify an object in the sky, which prompts a questionnaire. The questionnaire gives the experts a basis to form their individual results.

Experts: When a user requests to identify and object, experts use the information for the questionnaire to determine its identity. Experts will be mutually exclusive, each being knowledgeable on one specific and different domain. This allows for each expert to be modular and easily swappable.

Search for gazing location: The user wants to find near by star gazing spots. The user's location is given to the expert (or experts) to determine these spots.

List identification results: The system will display the results from the information given from the experts. When given the results from the experts, they will be filtered and ranked, only displaying the relevant information to the user. In the case of identifying an object, the top ten results will be displayed. In the case of locating star gazing spots, a map will be displayed with all the suggested points marked. The results will be within a set kilometre radius.

Check object info: From the results of identifying an object, the user can inspect each result, revealing additional information about it.

Change app settings: The user goes to the settings page to adjust general settings of the application, such as setting the kilometre radius for the star gazing spot search.

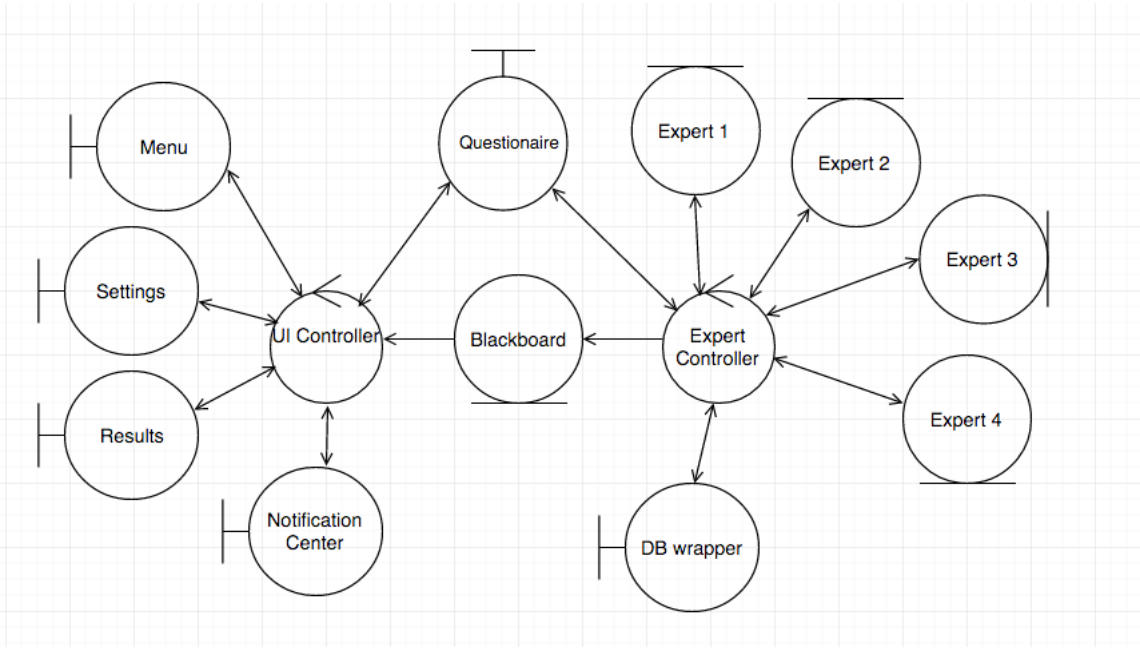
View notifications: The user can check the recent notifications sent to them from the system about local sky phenomenon occurring in the near future.

Inform of phenomenon in sky: The system sends the user notifications about local phenomenon occurring in the near future.

Update data store: The developer wants to update the system's data store. They shall be able to add and/or removing objects as needed.

Change experts: The developer wants to change the experts of the system. Since the experts are modular, the developer shall be add and remove them will little effort.

3 Analysis Class Diagram



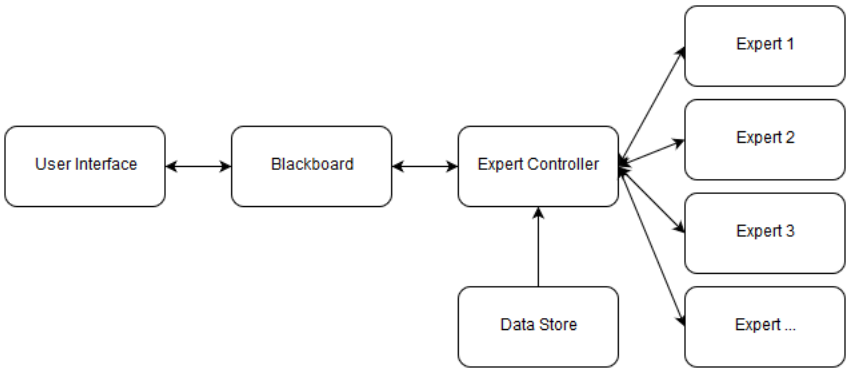
4 Architectural Design

4.1 System Architecture

The overall architecture of IndentiSky is based off of the Blackboard Architecture Style. This architecture style is most appropriate for solving non-deterministic problems, or problems where approximate solutions are acceptable, through the collaboration of some number of distinctive, independent knowledge sources. IdentiSky fits this description, as there are many different realms of knowledge, such as date and time, location, brightness, colour, and motion, that will be considered when attempting to identify some object in the sky based on information given by the user. Since much of this information is descriptive, the application is not expected to give exact results, and estimates are appropriate. As well, depending on the category chosen by the user, different realms of knowledge may be more appropriate for finding a solution– for example, number of stars is a useful property for identifying constellations, but nothing else. The Blackboard Architecture makes it easy to swap out knowledge sources, which is another reason that it has been selected

for IdentiSky.

Using the Blackboard Architecture will ensure low coupling, since all knowledge sources, called *experts*, will not actually communicate directly between themselves, but with the *blackboard*. High cohesion is also attained, since each expert consults itself on its topic of expertise. In addition to these basic components of the Blackboard Architecture, an *experts controller* will manage the experts, interfacing them with the blackboard and the *data store*, and a *user interface* subsystem will manage input and output between the user and the blackboard.



4.2 Subsystems

1. User Interface
   - Purpose: Displays information and options to the user. Collects input data from the user.
   - Relationships: Sends user-information to blackboard, receives questions and results from blackboard to be displayed to the user.
2. Expert Controller
   - Purpose: To interface the experts, and especially to swap in (or "activate") the experts that are currently needed and swap out (or "deactivate") those that our not.
   - Relationships: Connects the appropriate experts to the blackboard, allowing two-way communication. Also connects each expert to the data store.
3. Experts
   - Purpose: Each expert's purpose is to provide a partial solution to the problem based on their particular domain of expertise.
   - Relationships: Gets information through the expert controller.
4. Blackboard
   - Purpose: To contain all the relevant information to be accessed by the experts, including information that may be added or modified by the experts.
   - Relationships: Gets information from the user interface and sends results and questions to the user interface. Communicates with the experts through the expert controller.
5. Data Store
   - Purpose: Contains all possible identifiable objects for the experts to reference.
   - Relationships: Accessed by the experts through the expert controller

5 Class Responsibility Collaboration (CRC) Cards

Class Name: Expert Controller	
Responsibility:	**Collaborators:**
Send expert information to the Blackboard.	Expert 1-4
Send users answers to the experts	Expert 1-4, Questionnaire

Class Name: Expert	
Responsibility:	**Collaborators:**
Narrow down the identification process.	The other experts, expert controller
Access data store to cross reference user answers with known objects	Expert Controller, DB Wrapper

Class Name: Blackboard	
Responsibility:	**Collaborators:**
Determine what the object could be.	Expert Controller, Expert 1-4
Compile all of the expert input and determine which of the answers is the most likely	Expert Controller, Expert 1-4, Questionnaire
Send the compiled answer to the user	UI Controller

Class Name: Questionnaire	
Responsibility:	**Collaborators:**
Send questions for the user to answer	UI Controller
Send users answers to the experts	Expert Controller

Class Name: DB Wrapper	
Responsibility:	**Collaborators:**
Allow access to data stores for each expert. Only the expert which is associated with each data store can access it	Expert Controller
Encrypts the communications of the app and the data stores.	

Class Name: UI Controller	
Responsibility:	**Collaborators:**
Controls the interaction with the user	
Takes the user input and sends it the correct place	
Displays the questions to be answered to aid in the identification process	Questionnaire
Display the answer(s) that the experts have come up with	Blackboard, Results

Class Name: Notification Center	
Responsibility:	**Collaborators:**
Send push notifications to the user to alert them of cool things they can see in the sky based on their location	Location Expert, UI Controller

Class Name: Results	
Responsibility:	**Collaborators:**
Displays the results from the experts	Blackboard, UI Controller

Class Name: Settings	
Responsibility:	**Collaborators:**
Allows user to filter what they want to find	UI controller
Allows user to change general settings	UI Controller

Class Name: Menu	
Responsibility:	**Collaborators:**
Displays at the app startup with prompts for what the user wants to do next	UI Controller