# SRS Document for Fixit!

March 13, 2018

By Sunny and the Boys

# 1. Introduction

## 1.1 Purpose

The University of California Irvine, UCI, needs two new subsystems to help with (1) campus upkeep and (2) resolve emergency situations. This is outlined by the Repair, Maintain, and Respond (RMR) Project. This system, FixIt!, aims to efficiently help with campus upkeep as it constantly needs maintenance and repair. This is done as the system will be built to facilitate the reporting, repair, and maintenance of UCI facilities.

The FixIt! System's main purpose is to increase the efficiency of receiving and delegating reports, in comparison to the current workflow. The entire process of creating reports, delegating work, scheduling work, and regular maintenance will be streamlined by the new system.

## 1.2 Scope

The FixIt! application is currently only being utilized by the students, faculty, and maintenance employees of the UCI campus. In the future, this app could be expanded for other universities. FixIt! will include both a web and mobile version, allowing it to be available to users with access to the internet. The application can only be used to report within campus, limiting issues to only those pertaining to UCI.

## 1.3 Definition, acronyms, and abbreviations

| **Acronym and abbreviations** | **Definitions** |
|---|---|
| FM | Facilities Management |
| UCI | University of California, Irvine |
| app | Application |
| RMR | Repair, Maintain, and Respond |

## 1.4 References

Emily Navarro's Informatics 113 Lecture.

## 1.5 Overview

FixIt! is a mobile and web application intended to improve the current repair process within the UCI campus. By having a more effective system, FixIt! Will also

contribute towards a cleaner and safer campus. The system is a two part process: rapid reporting and rapid response, where UCI affiliates contribute by reporting issues throughout campus and FM work to resolve them. Affiliates will be able to provide concise and essential information through a convenient process to streamline issues within campus. FM operates by collaboration and delegation of reports into tasks to increase efficiency.

# 2. General description

### 2.1 Product Perspective

"Fixit!" is part of a larger system, namely the *Repair, Maintain, and Respond*, which also contains the "Respond!" application. In terms of related software, the app also makes use of location services and the UCINetID login system.

### 2.2 Product Functions

This system divides into 4 main functions:

a. Efficient way to report necessary repairs on campus
   -The system allows all UCI affiliates to report many issues that needs to be repaired around campus to maintain the attractiveness of the campus. The system also help minimize the duplicates of the report in order to keep it more organized.
b. Efficient way to delegate tasks
   -The system allows the Facilities Management Manager to delegate work more efficiently to each employees, which gives them more time to repair issue instead of spending unnecessary time on assigning tasks.
c. Allow UCI affiliates to keep track of the process of reports
   -Unlike the system-as-is, this system allows the users to keep track of the process of the report users make from when they submit the report.
d. Increase user satisfaction
   -The system will make new prospective students want to attend UCI by keeping the campus the clean.

### 2.3 User Characteristics

The primary stakeholders and users for the system should include UCI affiliates (students, staff, and faculty) and facilities management employees. Since affiliates will be the biggest stakeholder group, they will be expected to know how to create reports in the application, which entails taking pictures, classifying the issue, and providing the location of the issue. This implies that they are familiar with using mobile devices and with filling out forms on these devices. The same sort of expertise should be required for both facilities management employees and managers. However, these non-affiliates should also be familiar with creating operational plans for maintenance and repairs. The

manager should be the most familiar with this type of application, and they will be able to guide their employees through the process of responding to issues.

### 2.4 General Constraints

The system is to be built for the following systems (HW 2):

a. Mobile
    -Android
    -Apple
    -Blackberry

b. Desktop/Browser
    -Firefox
    -Chrome
    -Edge

Project specifics (HW 3 Field Notes):

-Budget: $600,000
-Timeline: 6 months

Record Keeping:
The entire history of the system should be kept. This includes all reports, maintenance schedules, and operational plans (HW 2).

### 2.5 Assumptions

-Minor details of the user interface
    -How the user sees the status of reports such as “completed”
    -Various tabs such as ‘Report Form’
-Report location visualization will be handled by users’ phone’s location services and Google Maps
-All UCI Facilities Management staff have their own UCInetID and UCI email to create their accounts.
-Facilities Management employees can create a new repair event on their schedule
-Report information gets saved
-Facilities Management employee changes the status of report throughout their work process
-Facilities Management Manager can detect duplicate reports with high accuracy
    -Facilities Management Manage will not create duplicate maintenance events
-Facilities Management employees have to create an operational plan for each of the tasks they are assigned to

# 3. Specific Requirements

## 3.1 Essential Requirements

### 3.1.1 Functional Requirements

**3.1.1.1** Sign in through UCINetID

ID: FR1
DESCRIPTION: Users are required to validate their identity as a UCI affiliate through their UCINetID when opening the application for the first time. Likewise, Users must have a UCINetID to have access to the application.
EVENT/USE CASE: Scenarios #2, 3, 4
FIT CRITERION: User enters in their UCINetID credentials and is logged into the system. User interface and authority is based on role of user, ie: FM manager, FM employee, UCI affiliates
RATIONALE: Failure to implement a sign in feature compromises security of system.
SOURCE: Requirements Elicitation Interview
DEPENDENCIES: None
HISTORY: January 24, 2018 : Elicitation Session

**3.1.1.2** Reporting an issue

ID: FR2
DESCRIPTION: UCI affiliates connected to the application will be able to report issues within campus that require attention from FM. Reports are limited to issues within the UCI campus.
EVENT/USE CASE: Use Case “Create a report”
FIT CRITERION: User fills information about issue and submits as a report, FM Manager can then view the report on their end of the application.
RATIONALE: inability to report issues harms the purpose of the application.
SOURCE: Case Study of RMR, Requirements Elicitation Interview
DEPENDENCIES: FR1
HISTORY: January 24, 2018 : Elicitation Session

**3.1.1.3** View report status

ID: FR3
DESCRIPTION: UCI affiliates are able to check the statuses of their reports. Viewing report status is only available to the user that have

reported the issue. Report statuses should include the following: submitted, in progress, and completed
EVENT/USE CASE: Use Case “view report status”
FIT CRITERION: User is able to view the status of their reports which correctly indicates the progress of their issue.
RATIONALE: Based on the case study, users should be kept up to date on their reports, and notified if chosen to do so. Failure to implement will not be critical to system, but reduces functionality to system.
SOURCE: Case Study of RMR, Requirements Elicitation Interview, Goal Model Achieve[StatusChecks]
DEPENDENCIES: FR1, FR2
HISTORY: January 24, 2018 : Elicitation Session

**3.1.1.4** Create and assign task
ID: FR4
DESCRIPTION: FM Manager will view reports and assign them to FM employees as tasks. FM Manager is the only one that can create and assign tasks
EVENT/USE CASE: Use Case Diagram “Assign Task to Employee”
FIT CRITERION: Task is created and designated to a FM Employee. Corresponding FM Employee can then view task on their end.
RATIONALE: Failure to create tasks conflicts with the purpose of the application.
SOURCE: Case Study of RMR, Requirements Elicitation Interview
DEPENDENCIES: FR1
HISTORY: January 24, 2018 : Elicitation Session

**3.1.1.5** View list of assigned tasks
ID: FR5
DESCRIPTION: FM Employees will be able to view all tasks they have been assigned. User will only see tasks that are assigned to them. User will only see tasks that are not yet completed.
EVENT/USE CASE: Use Case Diagram “Check List of all Assigned Tasks”
FIT CRITERION: FM Employee can see all uncompleted tasks they are assigned to
RATIONALE: Failure to implement requirement prevents FM Employee from completing tasks and reduces functionality of system
SOURCE: Requirements Elicitation interview
DEPENDENCIES: FR1, FR4
HISTORY: January 24, 2018 : Elicitation Session

**3.1.1.6** Build Operational Plan

ID: FR6
DESCRIPTION: FM Manager can create an Operational Plan that will allow multiple employees to collaborate and resolve larger issues. Employees are capable of creating subtasks and assigning them to other Employees within FM. Operational Plan can only be initiated by FM Manager
EVENT/USE CASE: Scenario #5, Use Case Build Operational Plan
FIT CRITERION: Operational Plan is created by FM Manager. FM Employees can collaborate on task. Completion of Operational Plan completes corresponding report.
RATIONALE: Failure to implement operational plan makes larger tasks harder to assign. Decreases functionality and usability of system.
SOURCE: Case Study of RMR, Goal Model Achieve[OperationalPlan]
DEPENDENCIES: FR1, FR4, FR5
HISTORY: January 24, 2018 : Elicitation Session

**3.1.1.7** Create Maintenance Event
ID: FR7
DESCRIPTION: The FM manager can create a new maintenance event based on tasks/routine checks listed on the Maintenance Calendar. Maintenance events can only be created by the FM Manager.
EVENT/USE CASE: Use Case Diagram's "Create Maintenance Event"
FIT CRITERION: Maintenance event is created and regularly assigns corresponding FM Employees with tasks.
RATIONALE: Failure to create Maintenance Events prevents usability and functionality of system.
SOURCE: Goal Model "Schedule Maintenance"
DEPENDENCIES: FR1, FR4
HISTORY: January 24, 2018 : Elicitation Session

**3.1.1.8** View Maintenance Calendar
ID: FR8
DESCRIPTION: The FM manager can see a calendar with required maintenance that is coming up.
EVENT/USE CASE: Use Case Diagram's "View Maintenance Calendar"
FIT CRITERION: FM manager views Maintenance Calendar and sees all
RATIONALE: Failure to View Maintenance Calendar makes Maintenance obsolete.
SOURCE: Case Study of RMR, Elicitation Session
DEPENDENCIES: FR1, FR7
HISTORY: Allowed FM employees to also view the maintenance calendar (3/12/18)

**3.1.1.9** Update status of task

ID: FR9

DESCRIPTION: FM employees have the ability to change the status of tasks they are given, to either in progress or completed. Change should be reflected elsewhere in the system wherever status of issue is displayed.

EVENT/USE CASE:Use Case Diagram's "Update Status of Tasks"

FIT CRITERION: On completion, original reporter should also be notified that issue is resolved.

RATIONALE: failure to update status of task prevents reports from being marked as completed, affects the purpose of the system.

SOURCE: Goal Model's "Achieve[StatusChecks],

DEPENDENCIES: FR1, FR3, FR4, FR5

HISTORY: January 24, 2018 : Elicitation Session

**3.1.1.10** Send event notification

ID: FR10

DESCRIPTION: After creating a maintenance event, there is an option to send a department-wide or campus-wide email notification about the maintenance that is being done.

EVENT/USE CASE: Use Case Diagram's "Send event notification"

FIT CRITERION: Completion of task sends reporter of issue a notification

RATIONALE: failure of feature is not critical, but is a feature mentioned by Case Study of RMR

SOURCE: Case Study of RMR, elicitation session

DEPENDENCIES: FR1, FR9

HISTORY: January 24, 2018 : Elicitation Session

### 3.1.2 Non-functional Requirements

**3.1.2.1** Security

The FixIt! App should be safe and secure for all users. The users' data and privacy should be protected from hackers. The FixIt! App should only be accessible to people who report or respond to RMR requests (Goal Model).

**3.1.2.2** Scalability

The FixIt! App should be able to scale well with the increasing amount of faculty and students. The app should also be able to handle a high amount of traffic as well as the potentially large amount of reports. This non-functional requirement is very important to achieve so that the FixIt! App can possibly be used at other universities.

**3.1.2.3** Usability

The FixIt! App should have an easy learning curve. Users should be able to quickly learn how to fill out reports and view the status of their reports. This non-functional requirement is important to achieve so that the app can be used by all students and faculty.

**3.1.2.4** Reliability

The FixIt! App should be able to run without failure for a specific amount of uses or over a specific amount of time. Reliability tests should be met in order to prove that the app can run reliably before it launches. This non-functional requirement is important to achieve so that the app can be reliable.

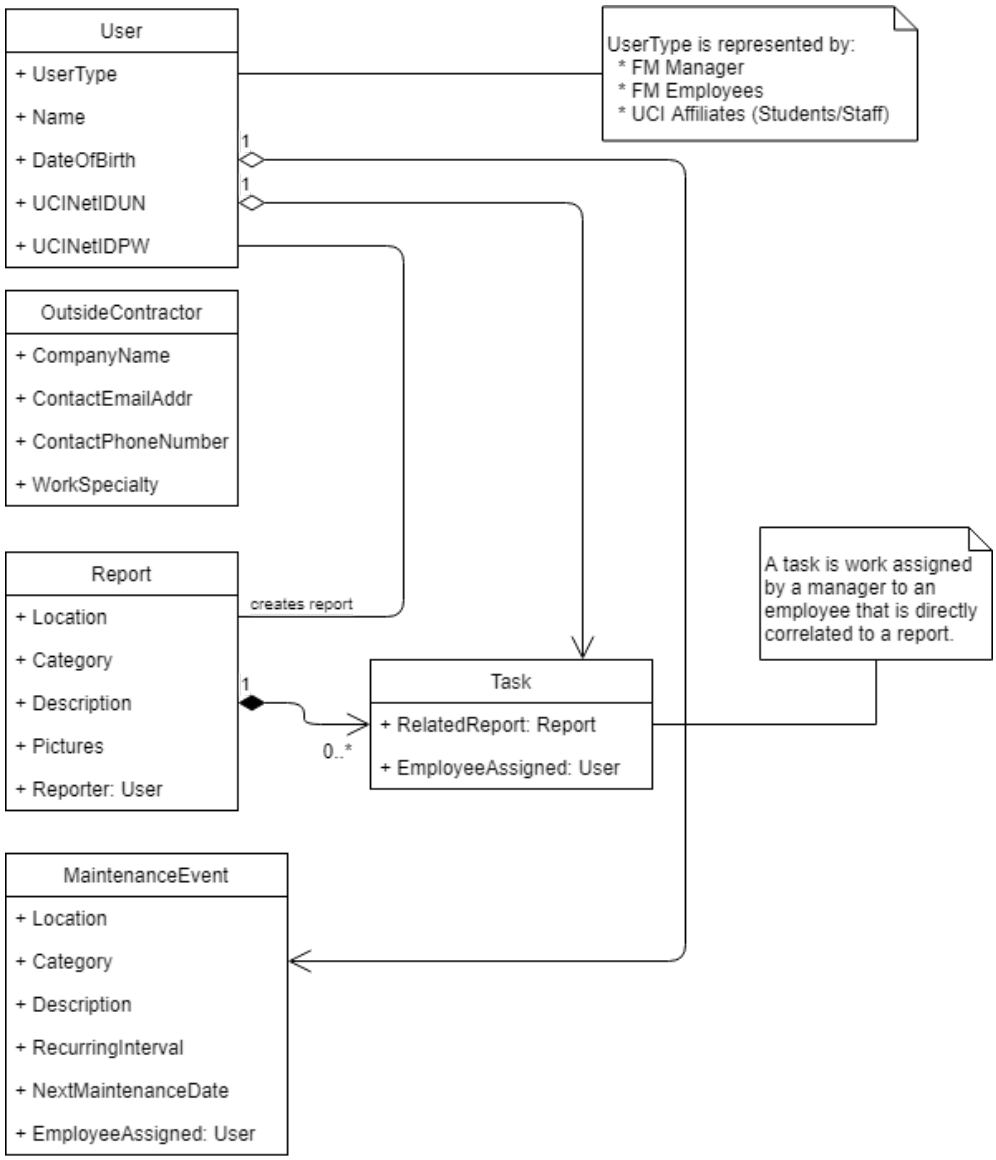
**3.1.2.5** Maintainability

The FixIt! App should be able to be maintained easily by the IT department post-launch. This non-functional requirement is important because maintenance of FixIt! is handled by IT, not the initial development team. Maintenance consists of fixing minor errors, bugs, and crashes, not overhauling the system.

### 3.1.3 External Interface Requirements

-GPS Location Services
-E-Mail Services
-Outside Contractor Communication Protocol
(User Interfaces, Hardware Interfaces, Software Interfaces, Communication Interfaces … as appropriate)

### 3.1.4 Logical Data Model




**User**

A user is any user of the system. Their permissions, usage of application, security, functionality etc. is dependent on what type of user they are. Users can be of three categories, FM Manager, FM Employees, or UCI affiliates.

**Outside Contractor**

Outside contractors information is stored in the system. Primary information is name, contact information, and work specialty (e.g. specializing in plumbing, cementing, etc.).

**Report**

Reports are stored with the primary information of the report, location, category, description, and pictures. Also stored with the report is the reporting user. This is not stored separately in the database.

**Task**
Tasks are different than reports in the database. Based on a single report, there can be multiple tasks for it. This is the basis of the operational plan. Multiple tasks can be assigned for different employees to fix. E.g. one employee unclogs drain, another re-caulks the sink, and last employee verifies caulking.

**Maintenance Event**
A maintenance is similar to a report except it holds two extra values. The interval in which this maintenance needs to be performed, and the next scheduled date for the maintenance.

### 3.1.5 Design Constraints

**3.1.5.1**
Application should be designed to run on devices running iOS 10.0 and later and Android 6.0 and later. (Source: Elicitation Interview)

## 3.2 Extension Requirements

### 3.2.1 Functional Requirements

**3.2.1.1**
The system should have the option to save ID and password information after logging in the first time

**3.2.1.2**
The system should have the option to push notification for report status updates

**3.2.1.3**
The system should limit the report optional information to 150 characters and two photos

### 3.2.2 Non-functional Requirements

**3.2.2.1**
The system should be changed according to appropriate complaint emails from users

**3.2.2.2**
The system should support unlimited amount of reports

**3.2.2.3**

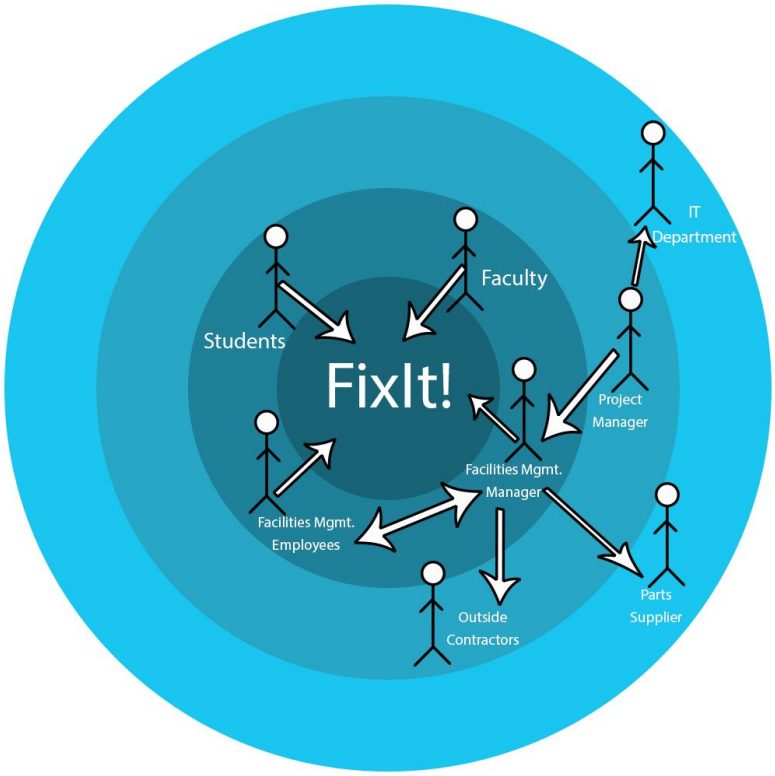
The system should delete information of report/tasks once it has been fixed

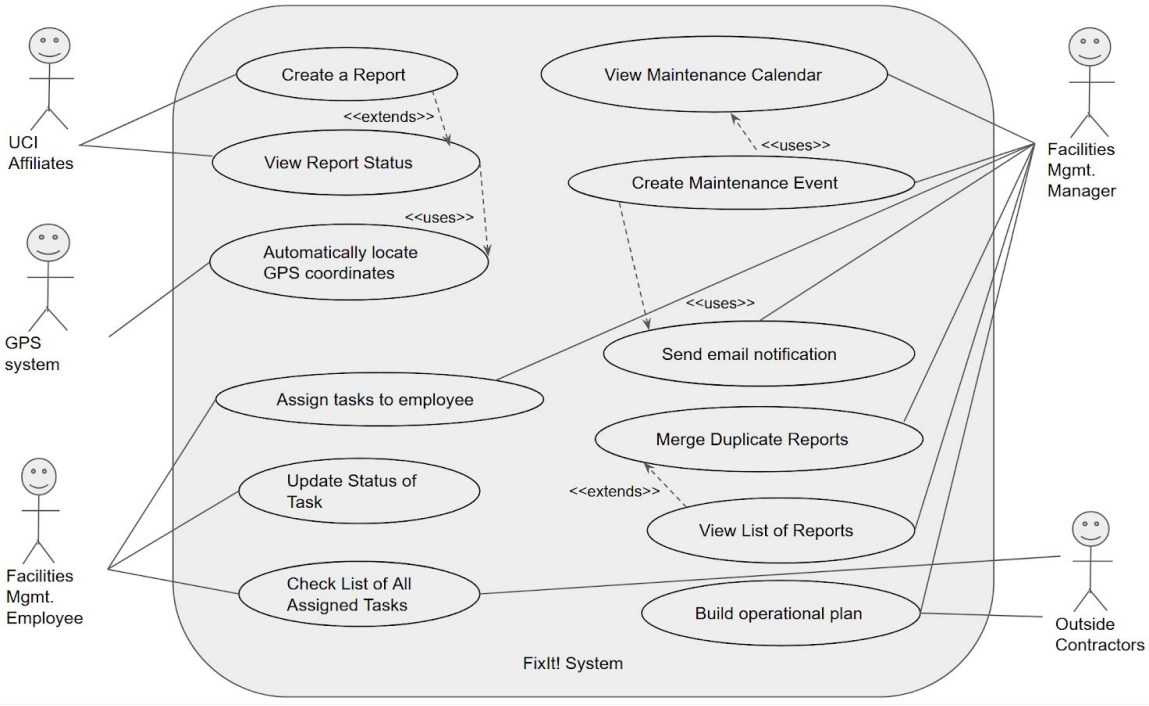
# Appendix

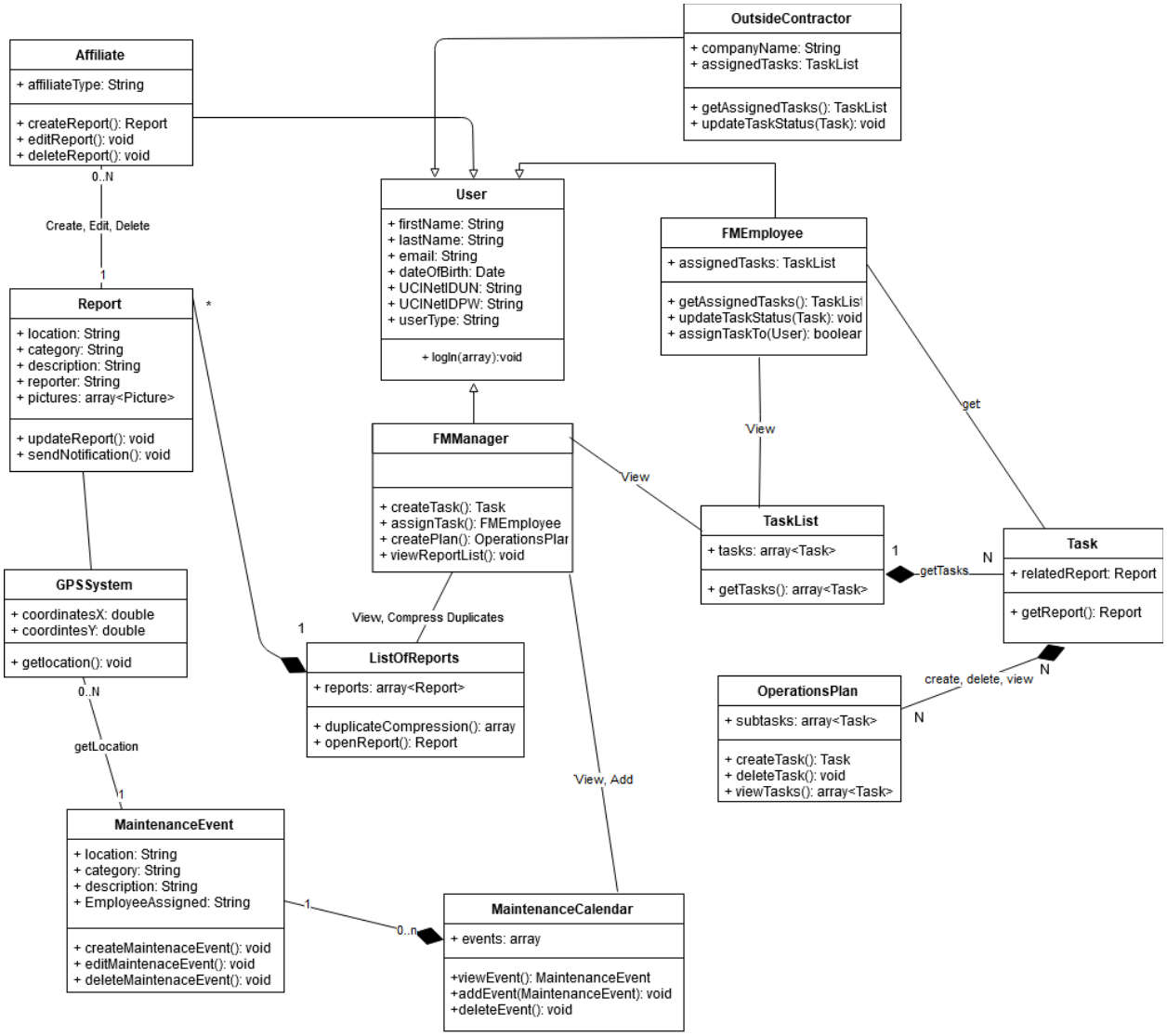
## A.1 Glossary

| **Acronym and abbreviations** | **Definitions** |
|---|---|
| FM | Facilities Management, the department in charge of all maintenance and repairs at UCI |
| UCI | University of California, Irvine |
| app | Application |
| RMR | Repair, Maintain, and Respond, the project that consists of the “Fixit!” And “Respond!” systems |

## A.2 Analysis Models (Stakeholder, Goal, Usage, Domain)






**A.3 Traceability (relationships/links between the requirements and the models)**

## Field notes

- Informed the customers of all the assumptions our group made and asked for their opinions on it. The customers agreed with all our assumptions
- Went over the document with the customer and asked if they wanted any changes before proceeding further with the project
- Informed the customers we will have the prototype of the app for further verification

## Missing Information

- Ask the customers to keep us updated on new buildings in order to keep our maps up to date

## Team Meeting Minutes

| **Team ID: Sunny and the Boys** | **Date: 3/8/18** |
|---|---|

| **Team Members (Name)** | **Role** |
|---|---|
| 1. Patrick Pham | Leader |
| 2. Andrew Wang | Scribe |
| 3. Patrick Tran | Member |
| 4. Matthew Robinson | Member |
| 5. Kavi Mathur | Member |
| 6. Euisun Lee | Member |

| **Agenda for this meeting, List of agenda items** | **Outcomes** |
|---|---|
| 1. Find gaps in information | Looked through past homeworks for missing information. |
| 2. Understand SRS template | Read through SRS template as a group |
| 3. Read through both examples | Read through both examples for guidance |
| 4. Split work up equally | Started to work on SRS document as a group |

| Problems encountered | Resolution |
|---|---|
| 1. Difficulty understanding logical data model | Researched logical data model |
| 2. Splitting up work evenly | Members chose which parts of the SRS document they wanted to complete |
| 3. Had difficulty understanding scope | Discussed the with group on what scope is |

| Plans for next meeting: Activity | Responsibility |
|---|---|
| 1. Finish SRS document | Everyone |
| 2. Find missing information | Everyone |
| 3. Write down field notes | Everyone |
| 4. Complete group member evaluations | Everyone |