Software Engineering Team 5

Cyber-Energy Operation Management System
Subsystems Report

Version <1.5>
3/15/2013

Document Control

Approval

The Guidance Team and the customer shall approve this document.

Document Change Control

Initial Release:	1.0
Current Release:	1.5
Indicator of Last Page in Document:	@
Date of Last Review:	3/15/2013
Date of Next Review:	To Be Determined
Target Date for Next Update:	To Be Determined

Distribution List

This following list of people shall receive a copy of this document every time a new version of this document becomes available:

Guidance Team Members:
Dr. Yoonsik Cheon
Dr. Irbis Gallegos
Aditi Barua

Customer:
Dr. Ralph Martinez

Software Team Members:
Gabriel Arellano
Chris Duran
Crystal Lopez
John McKallip
Ramon Vega
Matthew Wojciechowski

Change Summary

The following table details changes made between versions of this document

Version	Date	Modifier	Description
1.0	03/09/2013	Crystal Lopez	Initial Document
1.1	03/09/2013	Crystal Lopez	Introduction
1.2	03/13/2013	Gabriel Arellano, Chris Duran, Crystal Lopez, Ramon Vega, Matthew Wojciechowski	CRC Cards
1.3	03/13/2013	John McKallip	Subsystem Cards
1.4	03/13/2013	Chris Duran	Class Diagram
1.5	03/13/2013	Gabriel Arellano	Collaboration Graphs

TABLE OF CONTENTS

DOCUMENT CONTROL.. II

APPROVAL II
DOCUMENT CHANGE CONTROL II
DISTRIBUTION LIST.. II
CHANGE SUMMARY.. III

1. INTRODUCTION..1

1.1. PURPOSE AND INTENDED AUDIENCE 1
1.2. SCOPE OF PRODUCT 1
1.3. DEFINITIONS, ACRONYMS AND ABBREVIATIONS 2
1.4. OVERVIEW...2
1.5. REFERENCES 2

2. CRC CARDS 3

3. SUBSYSTEM CARDS...10

4. DIAGRAMS 12

4.1. CLASS DIAGRAM...12
4.2. COLLABORATION GRAPHS 12
4.2.1. High Level...12
4.2.2. Full Detail..14
4.2.3. Graphics Subsystem..15
4.2.4. User Interface Subsystem..15
4.2.5. Math Subsystem 16
4.2.6. Observation Subsystem...16

1. Introduction

1.1. Purpose and Intended Audience

The primary focus of this document is to provide methods to simplify the patterns of communication between the classes in the Cyber-Energy Operation Management System (C-EOMS). The document will demonstrate how classes can be grouped together and show how they collaborate among themselves in order to support a set of cohesive responsibilities, which can be depended on by a client, otherwise known as contracts. Furthermore the contracts identified here are used as an abstraction tool, for refining class hierarchies and detecting subsystems. [2] The target audiences for this document are the system developers (Team 5), the client and any other person(s) involved in the development of the C-EOM system.

1.2. Scope of Product

The Cyber-Energy Operation Management System (C-EOMS) will be used to gather information from several different energy-management sites situated around the UTEP campus. Presently, UTEP is becoming more energy efficient with the introduction of solar cells to harness the sun's clean, natural energy, as well as other green energy solutions. The C-EOMS system will not only read the energy received by these new pieces of technology (along with existing resources), but will also keep records on the energy consumption within UTEP itself. By creating a system with the capabilities to oversee and integrate all of this information, UTEP will be able to better assess issues and problems related to energy consumption, along with a guide to help point out areas for further improvement in energy management.

The C-EOMS will contain two main categories of user interfaces with which to view data, a generalized one for public viewing, and a private interface for viewing live data and statistics about the campus energy system. Additionally, the private interface will also provide a means to focus on relevant data sets during a system malfunction. Employees registered with the C-EOMS will also be able to receive system status summaries, either by voluntary request or automatically during a system malfunction.

By offering all of this information on the campus' energy system, we hope to provide non-employees a chance to see how their campus' energy is being distributed overall, as well as support employees with an invaluable tool to help transition our university into an environmentally sound energy consumer and energy producer.

1.3. Definitions, Acronyms and Abbreviations

Below is a list of acronyms and abbreviations along with their definitions.

Acronym/Abbreviation	Definition
C-EOMS	Cyber-Energy Operation Management System
FD	Facilities Director
Tech	Technician
UI	User Interface

1.4. Overview

The Subsystems report is divided into the following sections: Introduction, CRC Cards, Subsystem Cards and Diagrams. The Introduction is divided into five subsections, which contain the Purpose and Intended Audience of the document; the Scope of the product we are building (C-EOMS); a table containing the all acronyms and abbreviations along with their definitions; followed by an overview of the document, ending with a list of references that were used. The next section, CRC Cards, includes a list of "index" cards, which now include contracts and their collaborations. The following section, Subsystem cards, is composed of "index" cards that include classes grouped together that cooperate among themselves and support a set of contracts (also identified in the CRC cards). The document concludes with a section devoted to the displaying of a UML class diagram and a high-level collaboration graph as well as multiple lower level collaboration graphs.

1.5. References

[1] Gabriel Arellano, Chris Duran, Crystal Lopez, John McKallip, Ramon Vega, and Matthew Wojciechowski. *Cyber-Energy Operation Management CRC Report*.

[2] Cheon, Yoonsik & Gallegos, Irbis. *Contracts* [Microsoft Power Point] El Paso : s.n., 2013.

[3] Cheon, Yoonsik & Gallegos, Irbis. *Subsystems* [Microsoft Power Point] El Paso : s.n., 2013.

2. CRC Cards

Below is an alphabetized list of CRC Cards that are composed of the Candidate Classes, Responsibilities, Contracts and their respective collaborations.

<table>
<tr><td colspan="2"><b>Class Name:</b> Alert</td></tr>
<tr><td colspan="2"><b>Description:</b> Manage communication and creation of new malfunctions</td></tr>
<tr><td><b>Contracts:</b>
10. Generate Alert
- Generate an Alert with given parameters
- Send Alert to FD Interface
<b>Private Responsibilities:</b>
- Generate SMS/ e-mail
- Update FD Interface</td><td><b>Collaborations:</b>
FD Interface(11)
Malfunction(7)</td></tr>
<tr><td colspan="2"><b>Comments:</b>
Known attributes: Building(s), Data, Time, Sensor(s)</td></tr>
</table>

<table>
<tr><td colspan="2"><b>Class Name:</b> Building</td></tr>
<tr><td colspan="2"><b>Description:</b> Abstraction of each building on campus</td></tr>
<tr><td><b>Contracts:</b>
5. Calculate Consumption
- Calculate total Building consumption values
<b>Responsibilities:</b>
- Manage Sensors
- Calculate consumption of Building objects</td><td><b>Collaborations:</b>
Consumption(4)
Map Graphic(13)
Sensor(6,9)</td></tr>
<tr><td colspan="2"><b>Comments:</b>
Known Attributes: Name, Location, and Sensor(s)</td></tr>
</table>

Class Name: Consumption	
Description: Class used to compute consumption data	
Contracts: 4. Generate Graph data - Compute graph data by calculated consumption values. **Responsibilities:** - Compute Consumption	**Collaborations:** Building(5) Graph Graphic(3)
Comments: Used with the Sensor and Building classes to compute consumption statistics of sensor(s) and/or building(s)	

Class Name: Facility Director Interface (FD Interface)	
Description: Handles the dynamic data requests between the facility director and the system	
Contracts: 11. Display Alerts - Display given Alerts to FD Interface **Responsibilities:** - Display FD interface	**Collaborations:** Interface Alert(10)
Comments: Requires successful login by a user with facility director credentials	

Class Name: Graph	
Description: Create various graphs for consumption visualization	
Contracts: 2. Draw graph - Request data with specified parameters - Return image of a graph displaying requested data **Private Responsibilities:** - Draw Graph	**Collaborations:** Graph Graphic (3) Graphic Interface
Comments: Known attributes: Start time of content, end time of content, type(s) of content displayed (electricity, gas, water, etc.)	

Class Name: Graph Graphic	
Superclass: Graphic	
Description: Will be used to get data	
Contracts: 3. Draw Graph - Gather data - Construct graph based on data gathered **Private Responsibilities**: - Validate selection - Criteria	Collaborations: Consumption (4) Graphic Building Sensor Reading
Comments: None at this time.	

Class Name: Graphic	
Description: Encapsulate data in various formats as requested for all major graphic content	
Contracts: **Private Responsibilities:** - Retrieve and package data for use within another object	**Collaborations:** Consumption (4) Graph Graphic Malfunction (7) Map Graphic
Comments: PHP allows for associative arrays, which can hold various types of data concurrently	

Class Name: Interface	
Description: Abstract class of the user view interfaces	

Contracts:
 	
Private Responsibilities:
- Access Database
- Initialize web page
- Initialize graphics
	Collaborations:
	Graph(2)
	Map (12)
	User(1)

Comments: None at the time	

<table>
<tr><td colspan="2"><b>Class Name:</b> Malfunction</td></tr>
<tr><td colspan="2"><b>Description:</b> Class used to identify and manage a malfunction</td></tr>
<tr><td><b>Contracts:</b>
7. List current malfunctions based on parameters given
- prepare a list for Graphics class
- provide locations and status data
15. Provide Malfunction status transactions
- return current condition of a specific malfunction
- update malfunction data
<b>Private Responsibilities:</b>
<b>-</b> Manage malfunction status
- Instantiate Alert</td><td><b>Collaborations:</b>
Alert
Graphic
Reading
Task</td></tr>
<tr><td colspan="2"><b>Comments:</b>
Known attributes: Time, Type, Technician Assigned</td></tr>
</table>

<table>
<tr><td colspan="2"><b>Class Name:</b> Map</td></tr>
<tr><td colspan="2"><b>Description:</b> Class used to generate maps in the user interface</td></tr>
<tr><td><b>Contracts:</b>
12. Draw map
- Display the campus map
<b>Responsibilities:</b>
- Draw map</td><td><b>Collaborations:</b>
Graphic
Interface
Map Graphic (13)</td></tr>
<tr><td colspan="2"><b>Comments:</b>
Will implement by using Google maps API</td></tr>
</table>

<table>
<tr><td colspan="2"><b>Class Name:</b> Map Graphic</td></tr>
<tr><td colspan="2"><b>Superclass</b>: Graphic</td></tr>
<tr><td colspan="2"><b>Description:</b> Will be used to compose buildings and check for malfunctions</td></tr>
<tr><td><b>Contracts</b>:
13. Draw Map
- Gather map data
- Construct image of map based on data gathered
<b>Private Responsibilities:</b>
- Compose buildings
- Check for malfunctions</td><td><b>Collaborations:</b>
Graphic
Building
Malfunction (15)
Consumption (4)</td></tr>
<tr><td colspan="2"><b>Comments:</b>
None at this time.</td></tr>
</table>

Class Name: Mobile Interface	
Description: Handles the dynamic data requests between mobile user and the system	
Contracts: **Responsibilities:** - Display Mobile Interface	**Collaborations:** Tech Interface
Comments: Requires successful login by user with technician credentials	

Class Name: Public Interface	
Description: Handles the dynamic data requests between the general user and the system	
Contracts: **Responsibilities:** - Display Public Interface	**Collaborations:** Interface
Comments: None at this time	

Class Name: Reading	
Description: Representation of a consumption reading generated from a sensor	
Contracts: 8. Read data from sensors - Get sensor data - Record sensor data - Calculate consumption for sensor **Responsibilities:** - Calculate Consumption - Store Reading	**Collaborations:** Sensor (9)
Comments: Known attributes: Data, and Timestamp	

Class Name: Sensor	
Description: Retrieves specific sensor’s history or metadata	
Contracts: 6. Provide reading data - Assist relative buildings in calculating total consumption values 9. Store reading to sensor object - Periodically update database about new values **Private Responsibilities:** - Retrieve Sensor Data	**Collaborations:** Building Reading
Comments: Known attributes: Time, Expected Value, Sensor Data Type, Current data being read	

Class Name: Task	
Description: Handles extra data about a malfunction	
Contracts: 14. Manage task data - Tech users modify/update data as needed **Private Responsibilities:** - Manage Tasks	**Collaborations:** Malfunction (15) Tech Interface
Comments: Known routines: Add, Edit, and Delete	

Class Name: Tech Interface	
Description: Handles the dynamic data requests between Techs and the system	
Contracts: **Private Responsibilities:** - Display Tech Interface - Update Tech Availability	**Collaborations:** Interface Mobile Interface Task (14)
Comments: Requires successful login by user with technician credentials	

Class Name: User	
Description: Helps validate users and set up contents of interface	
Contracts: 1. Validate user credentials - Accept credentials as parameters - Return denial of access or redirect/load into personal workflow **Private Responsibilities:** - Validate user credentials - Generate personal workspace and connect to user interface	**Collaborations:** Interface
Comments: None at this time	

3. Subsystem Cards

Below is a list of “index” cards, which define the subsystems in the C-EOMS and their contracts.

Subsystem Name: Graphics Subsystem
Classes: Graphic, Graph Graphic, Map Graphic, Graph, Map
Collaboration Graph: See figures 4.2.1, 4.2.2, and 4.2.3
Description: Responsible for retrieving data, packaging data, and drawing maps and graphs to user interface.
Contracts: 2. Draw graph - Server: Graph 3. Retrieve and package graph data - Server: Graph Graphic 12. Draw map - Server: Map 13. Retrieve and package map data - Server: Map Graphic

Subsystem Name: User Interface Subsystem
Classes: Interface, Public Interface, FD Interface, Tech Interface, Mobile Interface, User
Collaboration Graph: See figure 4.2.1, 4.2.2, and 4.2.4
Description: Responsible for validating a user’s credentials and displaying alerts to the facility director user interface.
Contracts: 1. Validate user credentials - Server: User 11. Display alerts - Server: FD Interface

Subsystem Name: Math Subsystem

Classes: Building, Consumption, Sensor

Collaboration Graph: See figures 4.2.1, 4.2.2, and 4.2.5

Description: Responsible for storing readings data, providing readings data, calculating consumption data, and generating graph data.

Contracts:

4. Generate graph data
   - Server: Consumption
5. Calculate consumption
   - Server: Building
6. Provide reading data
   - Server: Sensor
9. Store reading to sensor object
   - Server: Sensor

Subsystem Name: Observation Subsystem

Classes: Alert, Task, Malfunction, Reading

Collaboration Graph: See figures 4.2.1, 4.2.2, and 4.2.6

Description: Responsible for reading data from sensors, generating alerts, listing current malfunctions, providing malfunction status updates, and managing task information.

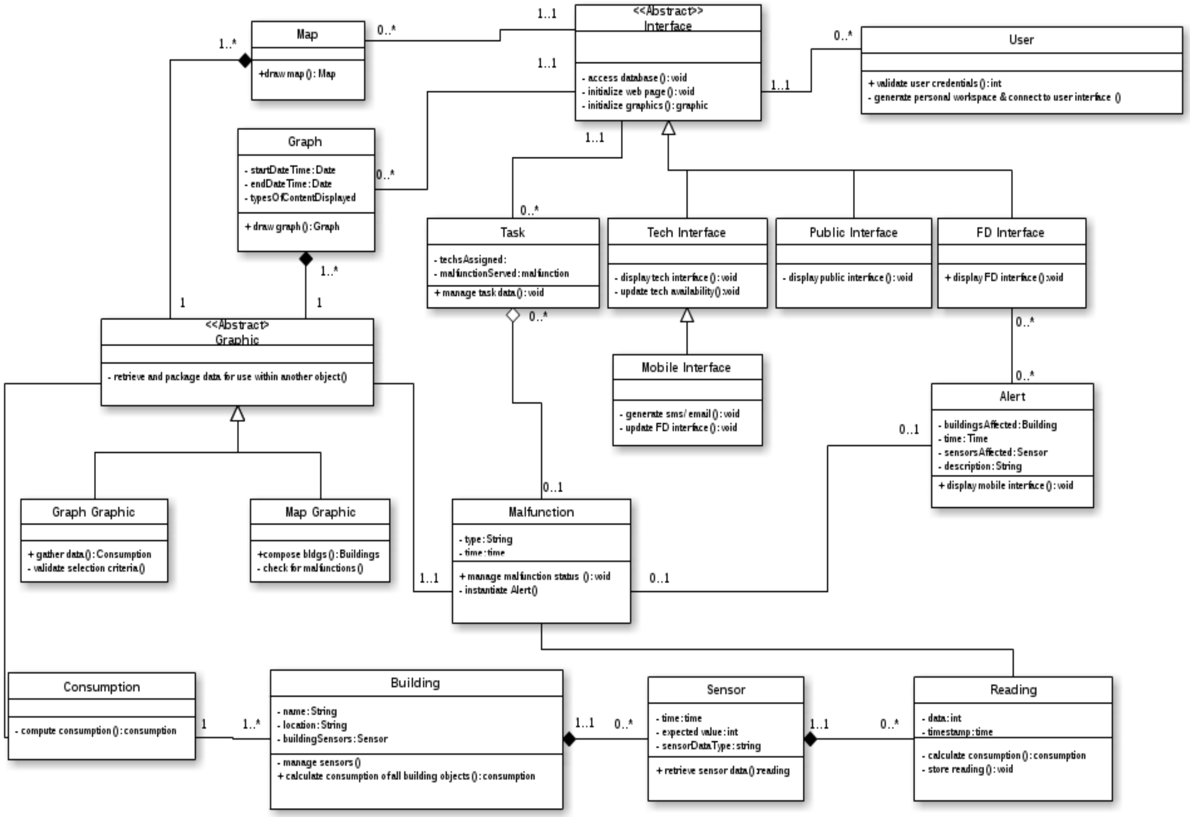
Contracts:

7. List current malfunctions based on parameters given
   - Server: Malfunction
8. Read data from sensors
   - Server: Reading
10. Generate alert
   - Server: Alert
14. Manage task data
   - Server: Task
15. Provide malfunction status updates
   - Server: Malfunction

4. Diagrams

4.1. Class diagram

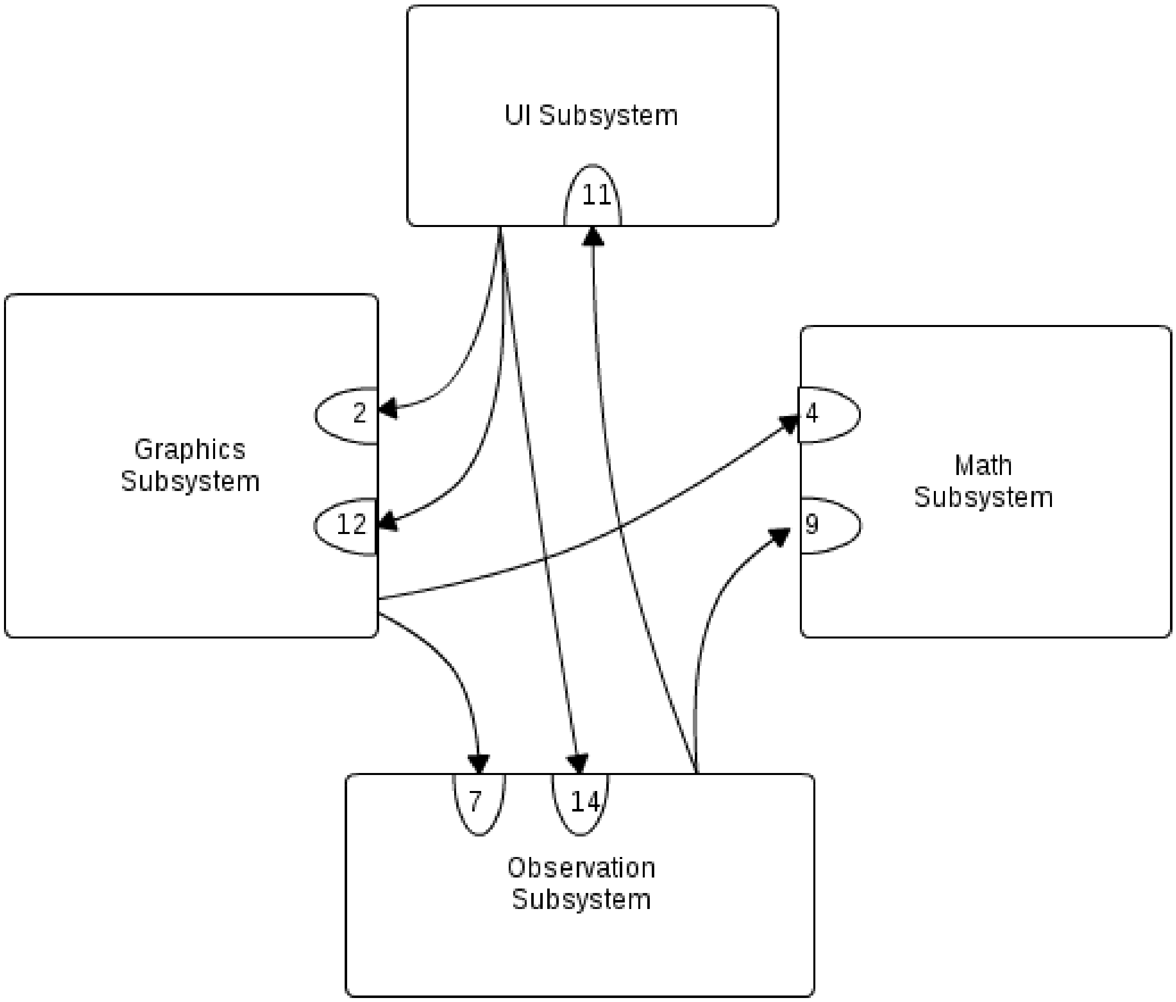
Below is a representation of the classes in UML form.


Map
+draw map (): Map
<<Abstract>>
Interface
- access database () : void
- initialize web page () : void
- initialize graphics () : graphic
User
+ validate user credentials () : int
- generate personal workspace & connect to user interface ()
Graph
- startDateTime : Date
- endDateTime : Date
- typesOfContentDisplayed
+ draw graph () : Graph
Task
- techsAssigned :
- malfunctionServed : malfunction
+ manage task data () : void
Tech Interface
- display tech interface () : void
- update tech availability () :void
Public Interface
- display public interface () : void
FD Interface
+ display FD interface () :void
Mobile Interface
- generate sms/ email () : void
- update FD interface () : void
<<Abstract>
Graphic
- retrieve and package data for use within another object()
Alert
- buildingsAffected : Building
- time : Time
- sensorsAffected : Sensor
- description : String
+ display mobile interface () : void
Graph Graphic
+ gather data () : Consumption
- validate selection criteria ()
Map Graphic
+compose bldgs () : Buildings
- check for malfunctions ()
Malfunction
- type : String
- time : time
+ manage malfunction status () : void
- instantiate Alert()
Consumption
- compute consumption () : consumption
Building
- name : String
- location : String
- buildingSensors : Sensor
- manage sensors ()
+ calculate consumption of all building objects () : consumption
Sensor
- time : time
- expected value : int
- sensorDataType : string
+ retrieve sensor data () :reading
Reading
- data : int
- timestamp : time
- calculate consumption () : consumption
- store reading () : void
1..*
0..*
1..1
1..1
0..*
1..1
0..*
1..1
0..*
1..*
1
1
0..*
0..*
0..*
0..1
0..1
0..1
1..1
0..1
1
1..*
1..1
0..*
1..1
0..*

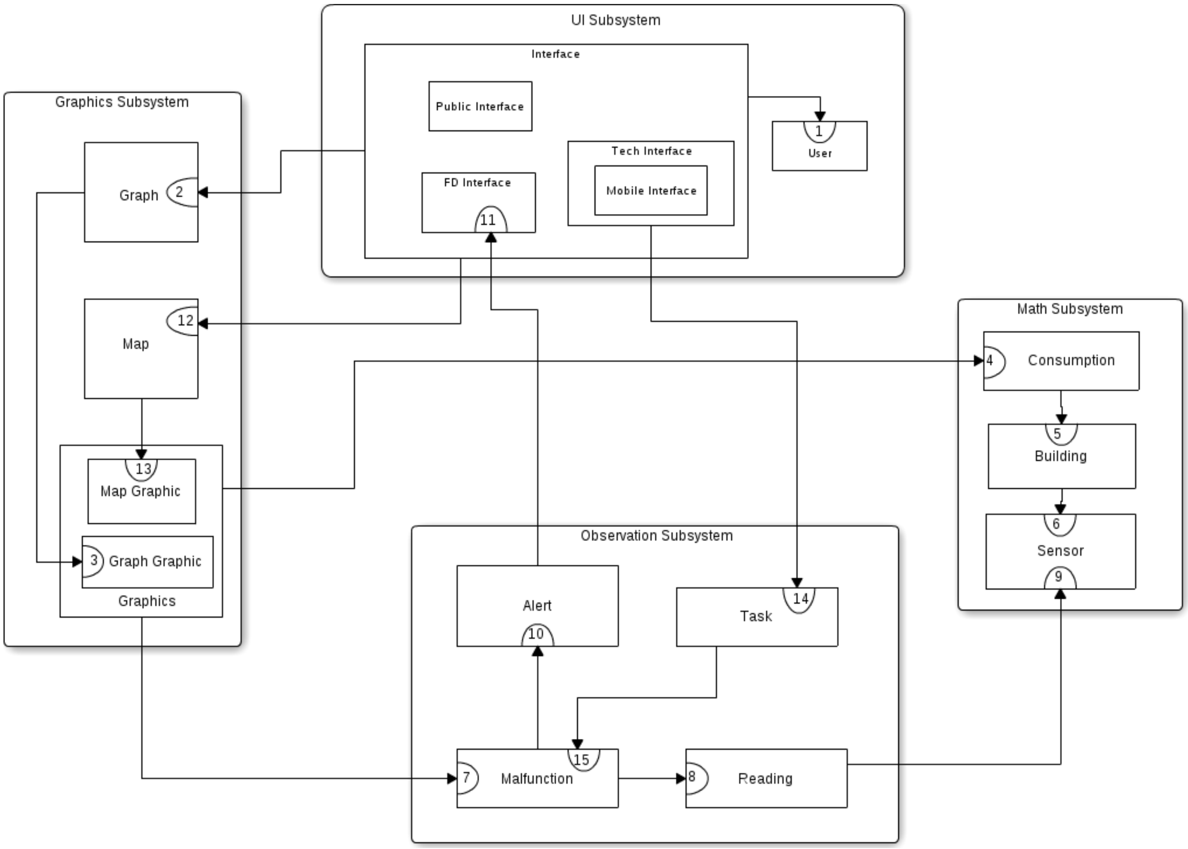
4.2. Collaboration graphs

Below is a set of collaboration graphs. The first graph represents the high level collaboration graph for simplicity. The second is the full representation of the collaboration graph. To further simplify the systems, they have been listed individually following the full collaboration graph

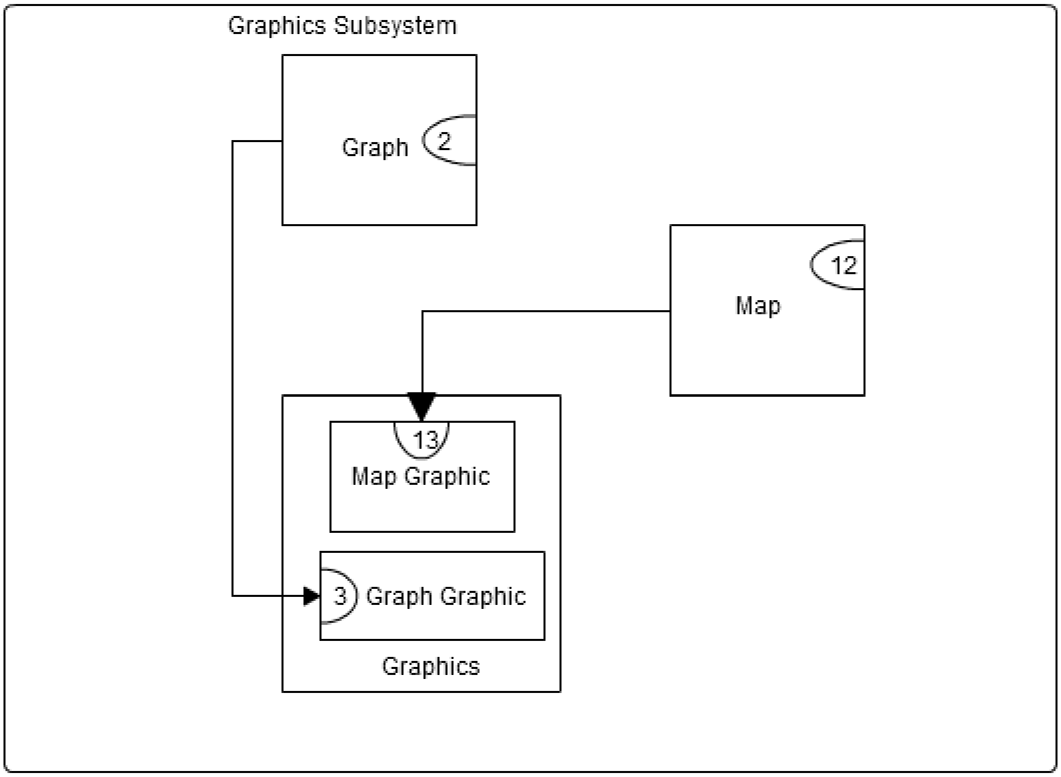
4.2.1. High Level


UI Subsystem
11
Graphics Subsystem
2
12
Math Subsystem
4
9
7
14
Observation Subsystem

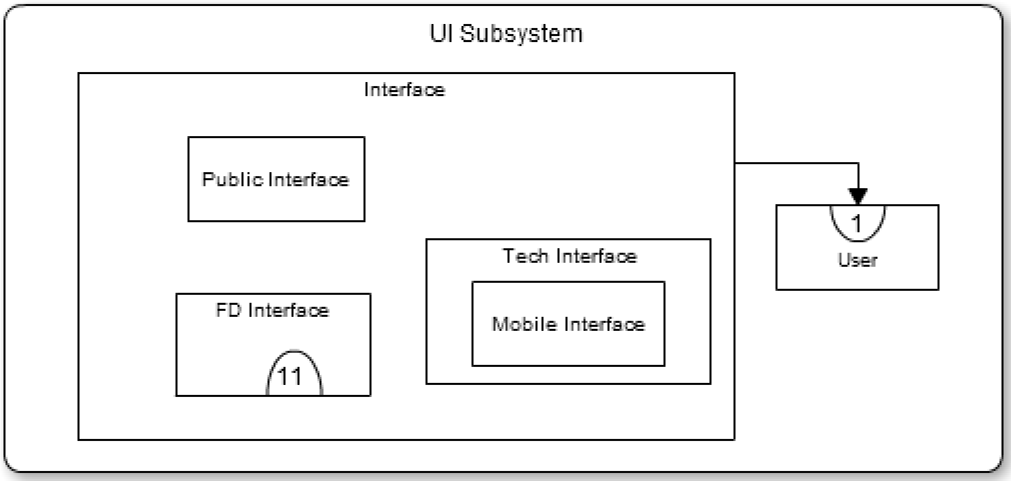
4.2.2. Full Detail



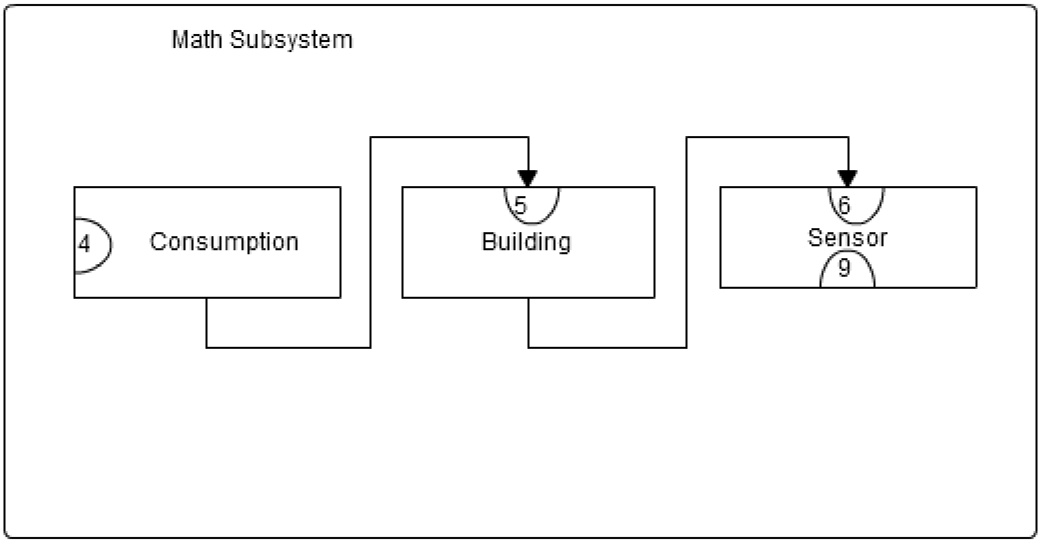
4.2.3. Graphics Subsystem



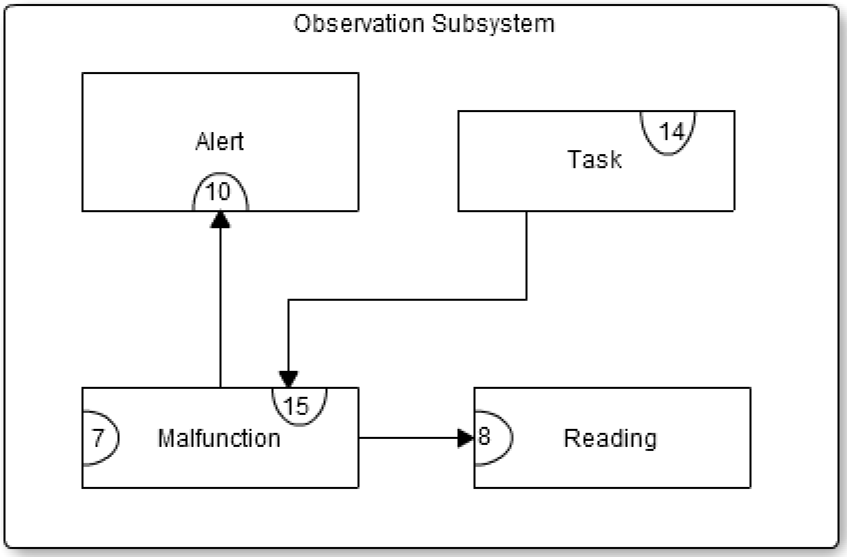
4.2.4. User Interface Subsystem



4.2.5. Math Subsystem



4.2.6. Observation Subsystem



@ END OF DOCUMENT