***Software Requirement Specification Analysis Artifact***

***Software Engineering I***

***CIS – 375***

***Professor: Thomas Steiner***

***Due Date: July 30, 2018***

***Version 1.0***

***The Debuggers:***

***Marco Seman***

***Martin Zanaj***

***Jason Lu***

***Maricruz Zamora***

***Mohammed Altulz***

***Audrey White***

Table of Contents

[**1.** **Introduction/Overview** 3](#_Toc521274484)

[**2.** **System Design (also called architectural design) - overall design of software as discussed in class** 4](#_Toc521274485)

[**3** **Decomposition - link between system design and detailed design.** 10](#_Toc521274486)

[**4** **Detailed Design (also called component-­‐level or program design)** 11](#_Toc521274487)

[**1.** **Use Case #1: New Customer** 11](#_Toc521274488)

[**2.** **Use Case #2: Customer Information** 12](#_Toc521274489)

[**3.** **Use Case 3: Service Request** 12](#_Toc521274490)

[**4.** **Use Case 4: List/Priority** 13](#_Toc521274491)

[**5.** **Use Case 6: Time Block & Job** 13](#_Toc521274492)

[**6.** **Use Case 7: Generate Customer Bill** 14](#_Toc521274493)

[**7.** **Use Case #8: Assistant Obtains Reports** 14](#_Toc521274494)

[**8.** **Use Case #9: Notification** 15](#_Toc521274495)

[**9.** **Use Cases #10: Survey** 15](#_Toc521274496)

[**5** **Data Design** 16](#_Toc521274497)

[**6** **User Interface Design –description of the user interface to the software** 18](#_Toc521274498)

[ **Login:** 18](#_Toc521274499)

[ **Menu:** 19](#_Toc521274500)

[ **New Job:** 20](#_Toc521274501)

[ **Waiting List:** 21](#_Toc521274502)

[ **Technician Time Block** 21](#_Toc521274503)

[ **Job List** 22](#_Toc521274504)

[ **Manager Approval:** 22](#_Toc521274505)

[ **Notification:** 23](#_Toc521274506)

[ **Payment:** 24](#_Toc521274507)

[ **Report:** 25](#_Toc521274508)

[ **Survey:** 26](#_Toc521274509)

[ **Technician** 27](#_Toc521274510)

[ Exit /Discard 27](#_Toc521274511)

[**7** **Other sections will depend on the type of design.** 28](#_Toc521274512)

[ **Data Flow Diagram** 29](#_Toc521274513)

[ **State Transition Diagrams** 30](#_Toc521274514)

[ **Complete ERD** 31](#_Toc521274515)

[**8** **Appendices, References, Index, Glossary, and so on, Access Code** 32](#_Toc521274516)

# **Introduction/Overview**

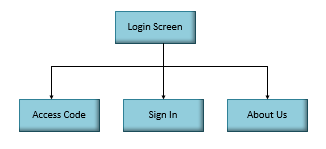
Osric owns Osric's Office Appliances and Décor where he employs subcontractors and technicians to decorate the offices of top business executives. Due to large demand of Osric's business, he has decided to develop an idea to assist customers of higher priority in a much appropriate time. The goal for our project is to take Osric's idea where he prioritizes customers and develop an efficient software that may save the company time, money, and satisfy its customers. By doing so, we must simulate the effectiveness of the scheme using data and statistics on a job mix. The statistics will identify the average waiting time before a job has started, the average queue length, the percentage of time the queue is empty on day and night, the number of blocks when a technician is idle, and the number of jobs that cannot be continued at night because no technician is available. Afterwards, we must compare Osric’s scheme to the pure first-come-first-serve bases using the same job list. Once the simulation is complete, we must look at the statistics from both schemes and determine which is more time and cost effective for Osric.

# **System Design (also called architectural design) - overall design of software as discussed in class**

The first thing that will be shown is welcome screen and it supposed to have:

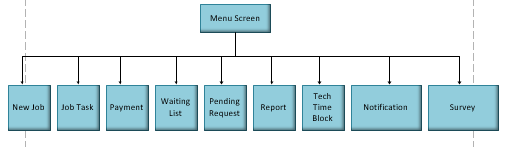
**2.1. Login Screen:**

Our first page should prompt the administrator (who in our case will be assistant) to login.



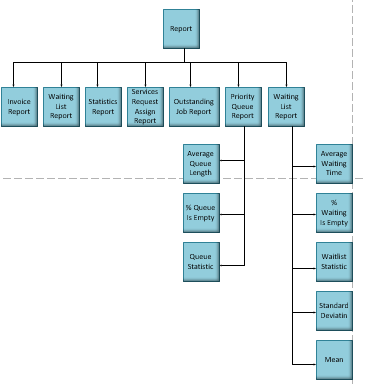
**2.2. Menu Screen:**

In the menu screen, the Administrator will select an option of their choosing.



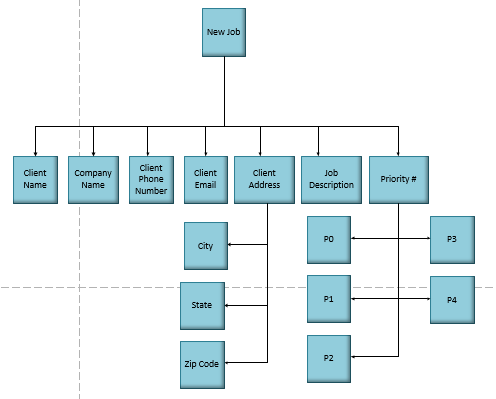
* 1. **Report Screen:**

When this button is initiated, the administrator will be able to view the following statistics report



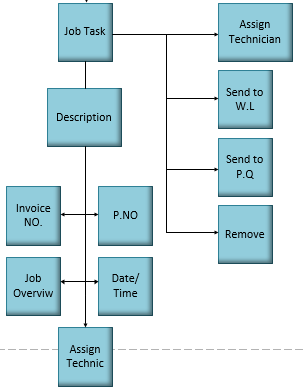
* 1. **New Job**

In this event, the administrator will submit a new job and will be required to fill out the following information:



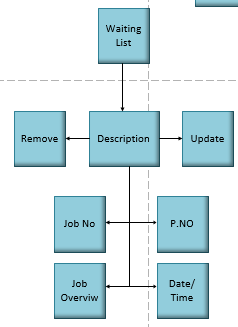
* 1. **Job Task**

In this tab, the administrator will be able to take a job and will edit it based on their requirements



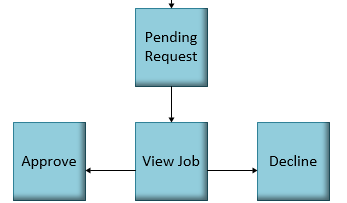
**2.6 Waiting List**

When this button is initiated, the assistant will be able to view the waiting list, along with being able to remove a job and update the waiting list which will sort the waiting list based on their priority, date, and time.



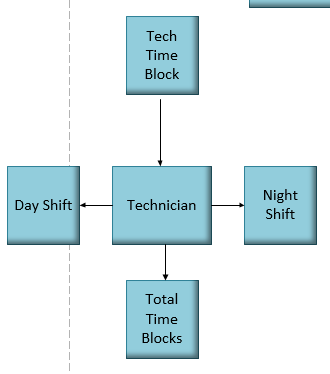
* 1. **Pending Request:**

In this scenario, the manager will be able to view a job, then determine whether he/she will decline or approve the job request.



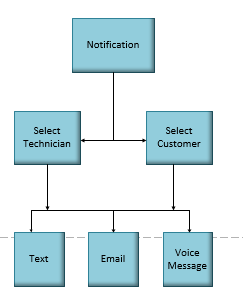
* 1. **Technician Time Block:**

In this scenario, there will be a need to input the amount of time blocks for the technician

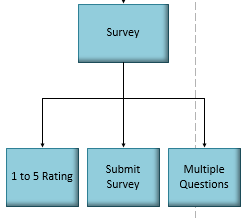


**2.9 Notification:**

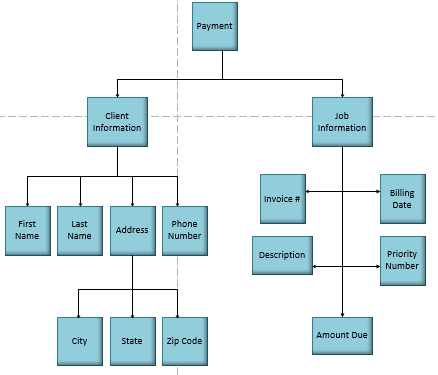
The customer and or technician will be notified whenever a job is either complete, uncomplete, or postponed



* 1. **Notification**



**2.11 Payment:**

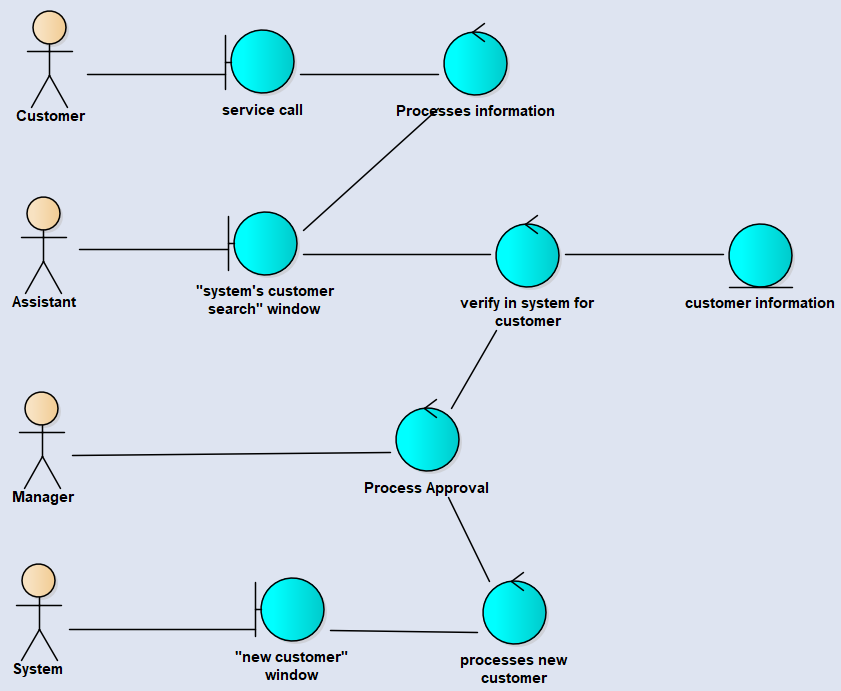


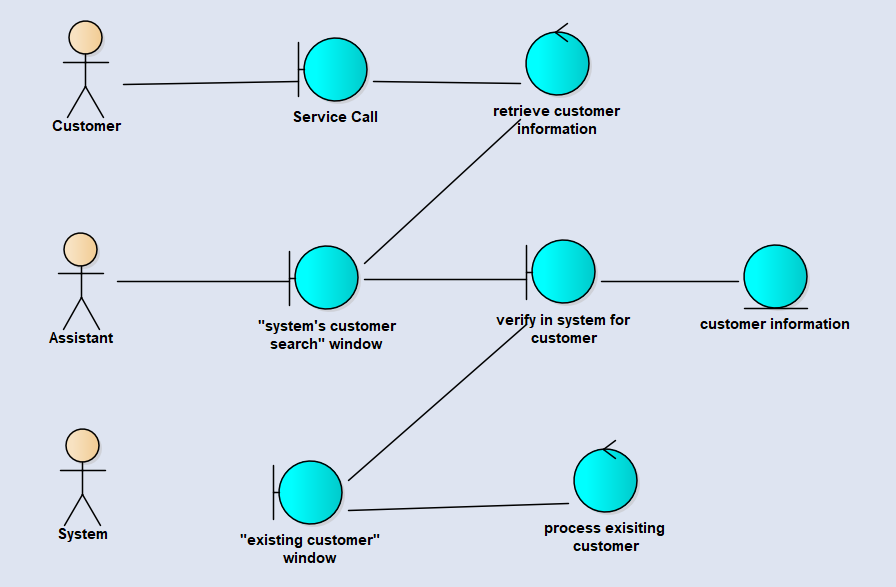
# **Decomposition - link between system design and detailed design.**

# **Detailed Design (also called component-­‐level or program design)**

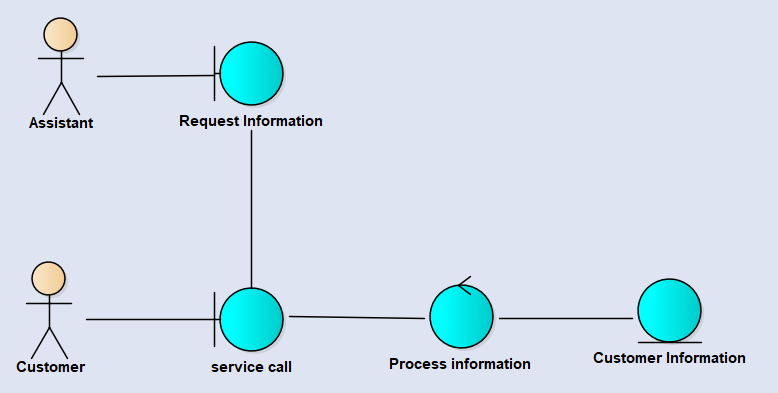
Detailed description of each program module or class/object. For each module, you should specify the software interface and other characteristics.

## **Use Case #1: New Customer**

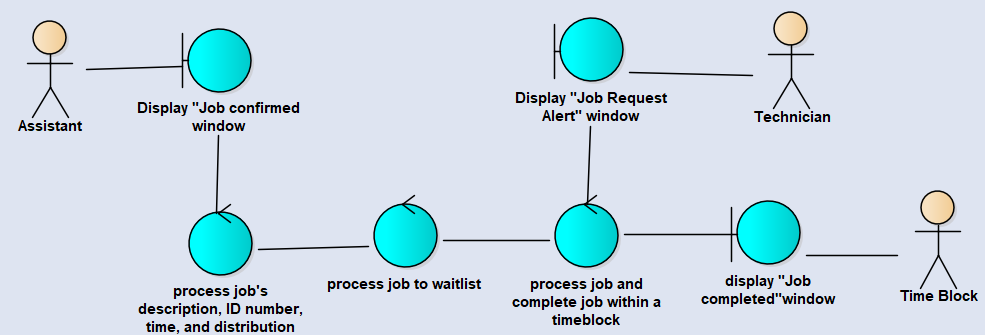




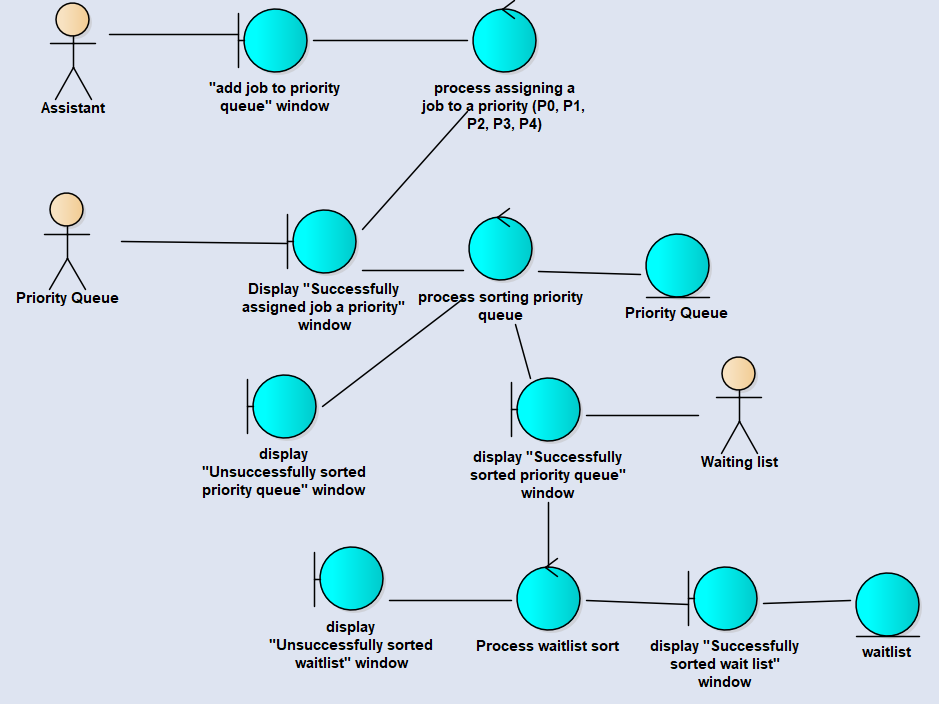
## **Use Case #2: Customer Information**



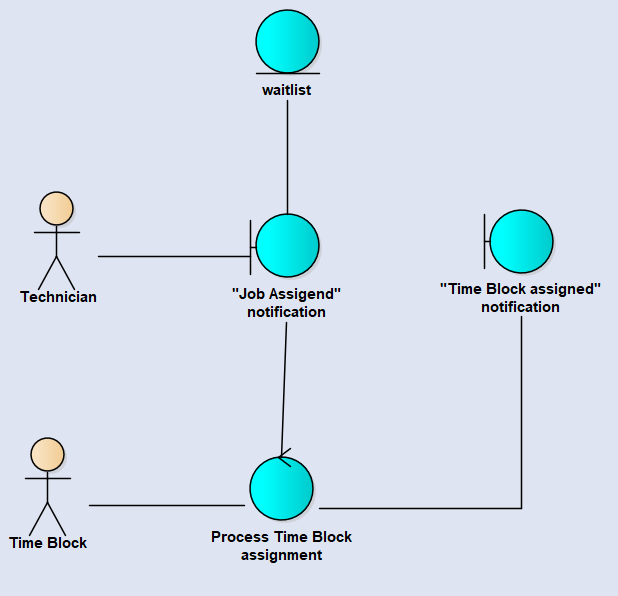
## **Use Case 3: Service Request**



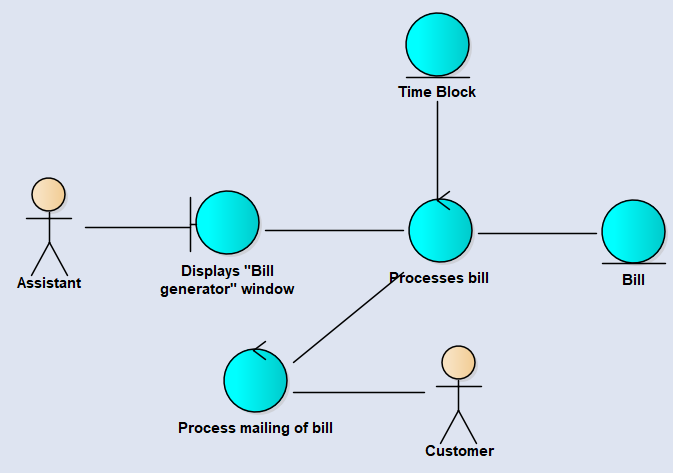
## **Use Case 4: List/Priority**



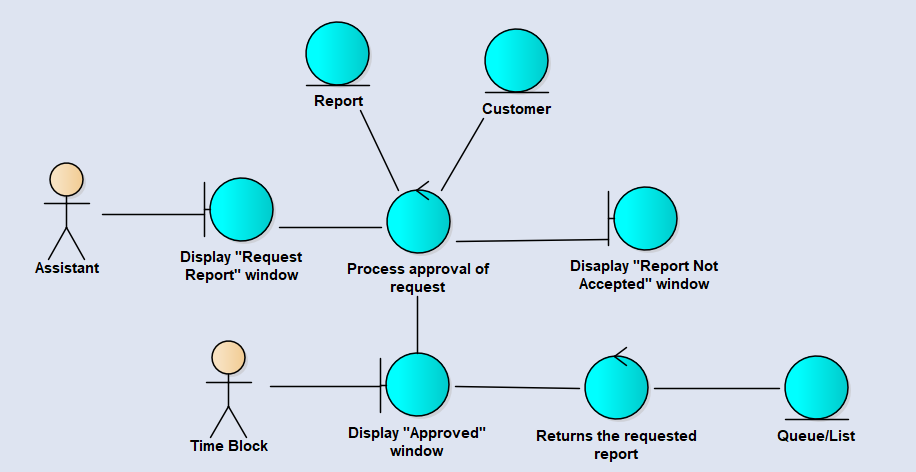
## **Use Case 6: Time Block & Job**



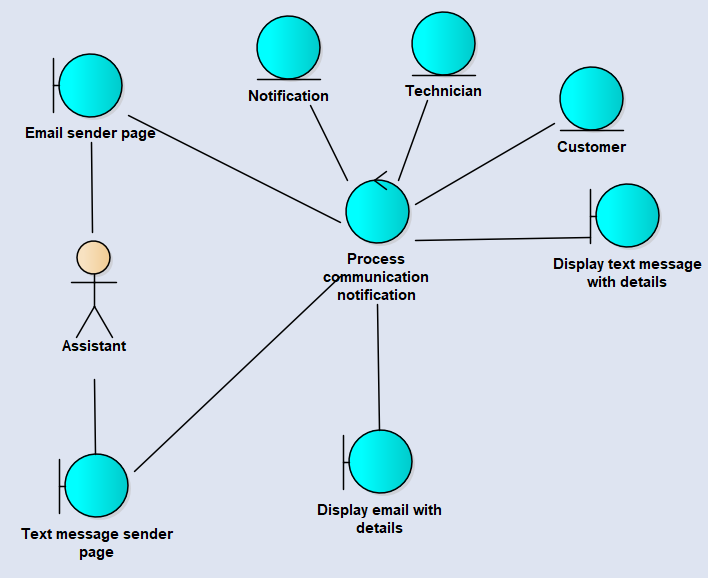
## **Use Case 7: Generate Customer Bill**



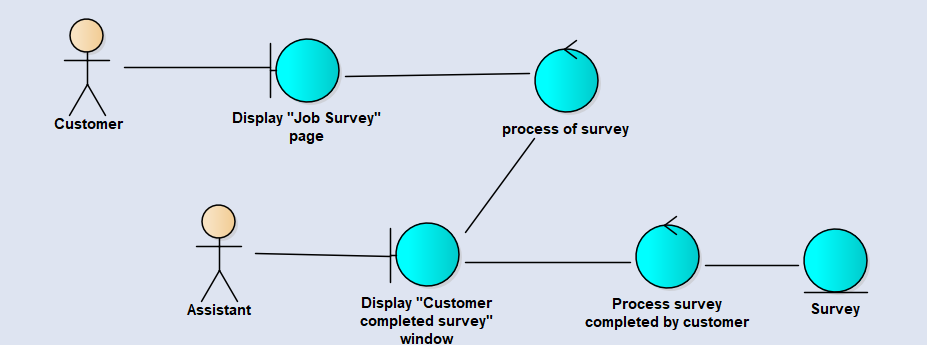
## **Use Case #8: Assistant Obtains Reports**



## **Use Case #9: Notification**



## **Use Cases #10: Survey**



# **Data Design**

* Priority Queue

*What is a PQ?*

Abstract data type that resembles a queue (FIFO, first-in-first-out in which data is structured in a way

that the first element in will also be the first one to leave the queue), but with the different that each

element that will come into the PQ, will have a determinate priority associated with it.

*Operations*

-In most PQ’s the most frequent operations that appear are:

Is\_Empty() checks that if the PQ is empty (no elements)

Insert() inserts a new element in the PQ

Get\_Max() which simply takes out the element sitting on top of the queue

*Our System*

-In our system such data structure will be used in order to implement our own PQ. The PQ will work like

a traditional PQ with the difference being what determined the priority. The priority of each job will be

determined according to a number from 0 to 4, with 0 signifying an urgent job request, and 4 signifying

a less urgent request. The system will ensure that each job is properly classified according to priority,

and it will be sent to list in which it will await to be picked up. In addition to being labeled with a

determinate priority, as the days pass each job will increase in urgency an eventually be assigned to an

available technician.

* Bucket Sort

*What is a bucket Sort?*

-It is a sorting algorithm that works by distributing the elements of an array into a number of buckets. Each bucket is then sorted individually, either using a different sorting algorithm, or by recursively applying the bucket sorting algorithm. It can be considered a comparison algorithm given it compares the different values and places them into their appropriate place.

*Operations*

-In most bucket sorts the implementation is fairly simple given it is a simple for loop with a recursive call

*Our System*

-In our system such algorithm would be implemented in order to determine the priority of a

determinate job. Hence, each time a new job will be processed, a priority would be associated with that

number, once each job has been processed it will be sorted with the bucket sort in its appropriate

“bucket.”

* Object Oriented Approach

*What is it?*

*-*Programming paradigm based on the concept of “objects”, in which may contain data, in the form of

fields, often known as attribute; and code in the form of procedures, often known as methods. In OOP

programs are designed by making the different object interact with each-other. The most popular

languages are class-based, meaning objects are instances of classes, which typically also determines

their type.

*Components*

The main components of such technique are: objects & classes, and the different methods & attributes

that interact with each other in order to complete a determinate process.

*Our System*

-This is the core philosophy of our entire program. The main idea behind our design is the

implementation of classes and objects, and the different process are simply the intercommunications

between the different objects and the methods present within them.

* Languages

C++

C#

* Platform

Visual Studio 2015/2017

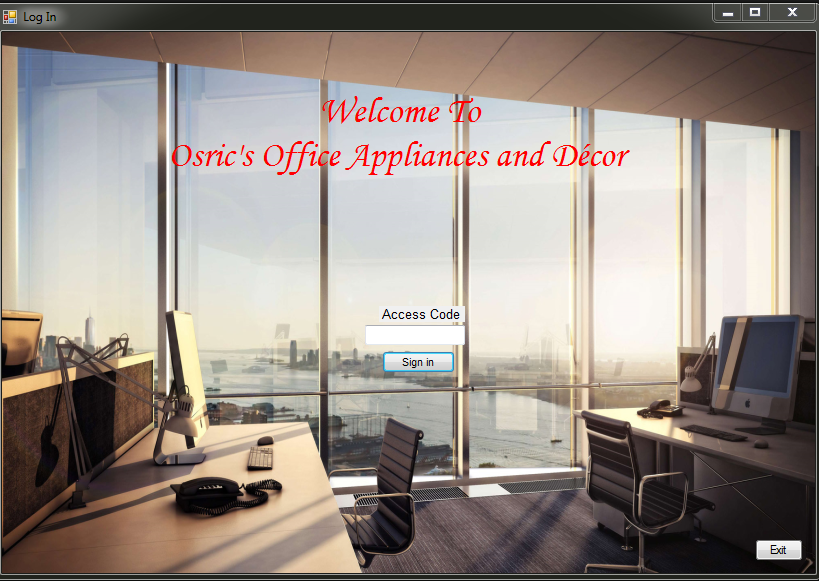
Windows Forms

# **User Interface Design –description of the user interface to the software**

(Note: user can be another software)

## **Login:**

* + Module Name: Log In
  + Description: Assistant will use their access code for logging
  + Action: Input arguments, clicking, processing to next module



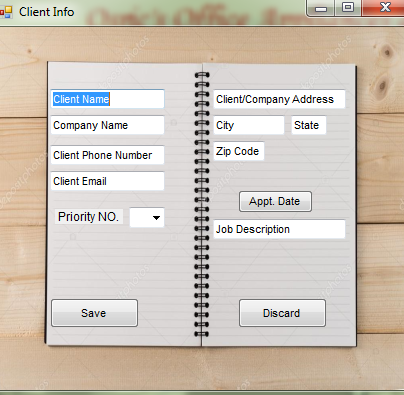
## **Menu:**

* Module Name: Menu
* Description: This is the view of the assistant which can check the different operations: New Job, Waiting List, Tech Time Block, Job Task, Pending Request, Notification, Payment, Report, Survey.
* Action: Assistant can open new interfaces, check data, check status, overview of the system.



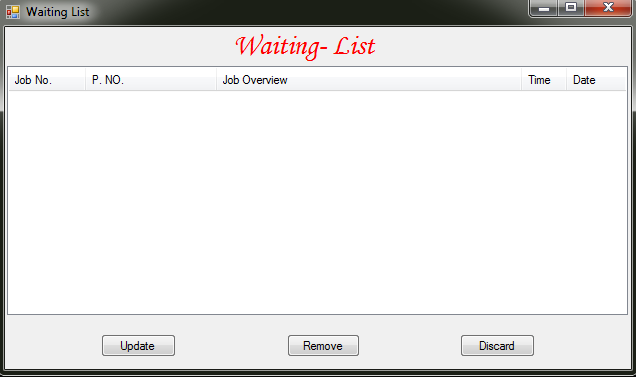
## **New Job:**

* Module Name: Client Info
* Description: This is the interface regarding the client information. Such interface will pop-up if there is a new, or existing customer. In this interface the assistant or technician will be able to retrieve: Client Name, Company Name, Client Phone Number, Client Email, Priority NO., Client/Company Address, City, State, Zip Code, App Date, Job Description.
* Action: The assistant or the technician will have to click on the module “Client Info.” Once there has been a click, the assistant/technician can either access the info, or update it.



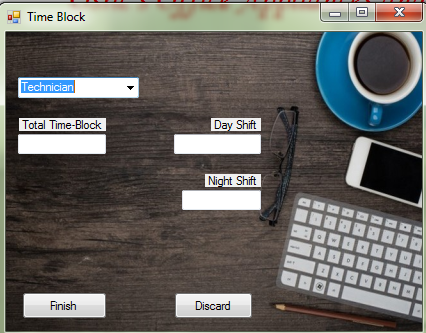
## **Waiting List:**

* Module Name: Waiting List
* Description: The waiting list will be used as a “sitting” place for the different jobs awaiting to be completed. Each job sitting in the first position will be the next one to be picked up by the next available technician. The waiting list will be composed of elements such as: Job No, Priority No, Job Overview, Time, Date, Update, and Remove.
* Action: The assistant will have to accept a job request, send it through the priority queue, and then eventually send it to the waiting list to be picked up by the technician.

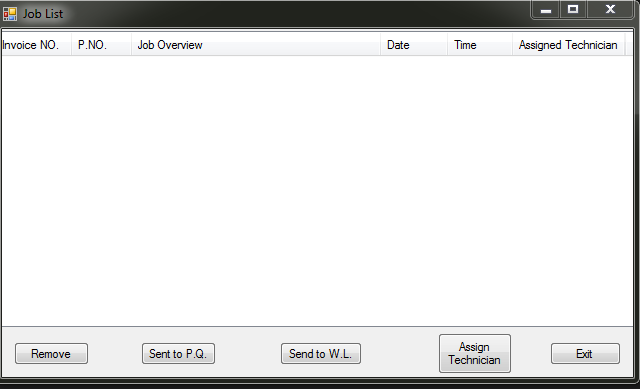


## **Technician Time Block**

* Module Name: Time Block
* Description: The interface will be pertinent to the time block, more specifically the amount of time that it will take to complete a determinate job. Such module will be composed of: Technician, Day Shift, Night Shift, Total Time-Block, Finish, and Discard.
* Action: The technician will take a determinate number of time blocks to complete a job. The assistant will be able to access such information, so as to be able to bill the customer for the service provided.

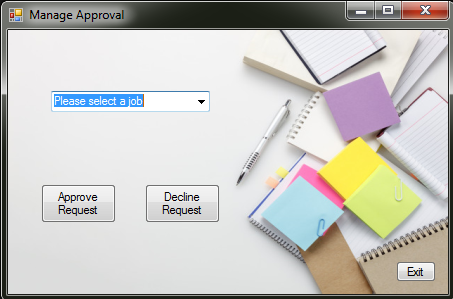


## **Job List**

* Module Name: Job Task
* Description: The job list will be an overview of the different jobs that have been assigned that are present in the system. The assistant will be able to easily see the different jobs, and the different information's that come with each specific job. This module will have components such as: Invoice No, Priority No, Job Overview, Date, Time, Assigned Technician, Remove, sent to PQ, send to WL, Assign Technician.
* Action: The assistant will be able to access such interface and modify it according to the required needs.
* 

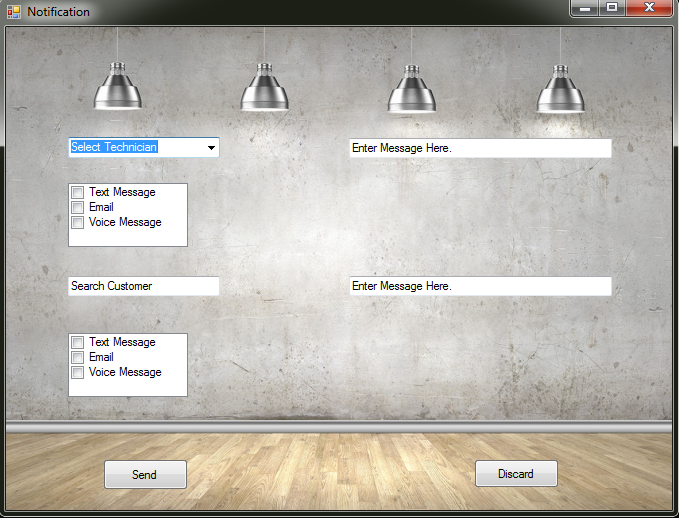
## **Manager Approval:**

* Module Name: Job Approval
* Description: This module will pop up whenever the customer requesting a service is not in the existing database, hence a new customer. The manager will either approve the job or decline the job according to data from within the system. The module will offer to the manager the options to: select a job, approve request, and decline request.
* Action: The assistant will seek approval from the manger in case a new customer requests a service with Osric.



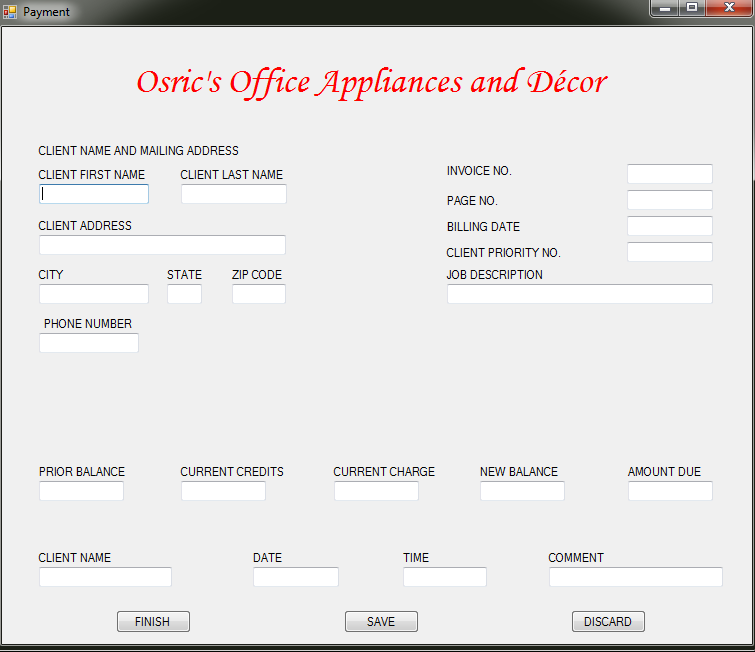
## **Notification:**

* Module Name: Notification
* Description: This module will ensure that there is constant communication between the customer and the assistant/technician. In the module there are present three distinct ways of communication: Text Message, Email, Voice Message (calling). Within the module there are tabs were the customer can access a specific assistant, or technician and can leave a determinate message. Once the customer has formulated a message it can decide to either send it through or discard it in case it changes mind.
* Action: The customer will have to open this determinate module, if it intends to communicate with the assistant or the technician.



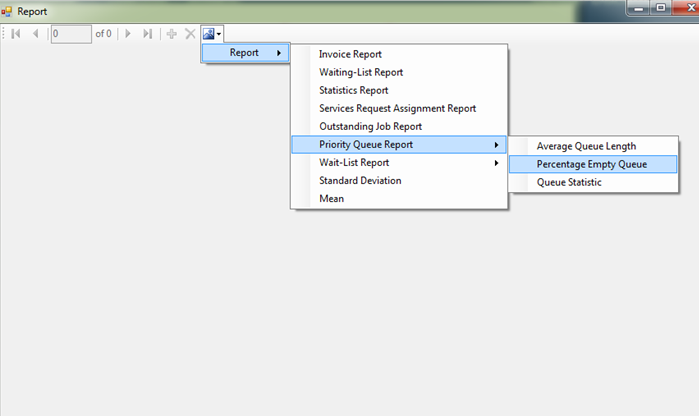
## **Payment:**

* Module Name: Payment
* Description: This module is pertinent and essential for the processing of the different payments that a customer will have to make once a service will be complete. Such module will have the different information regarding the job information, and the billing information. The different components in such modules are: Client First Name, Client Last Name, Client Address, City, State, Zip Code, Phone Number, Invoice NO, Page NO, Billing Date, Client Priority NO, Job Description, Priori Balance, Current Credits, Current Charge, New Balance, Amount Due, Date, Time, Comment.
* Action: The assistant will complete the section regarding the different metrics about the job charging aspect, and the customer will be able to submit a determinate payment.



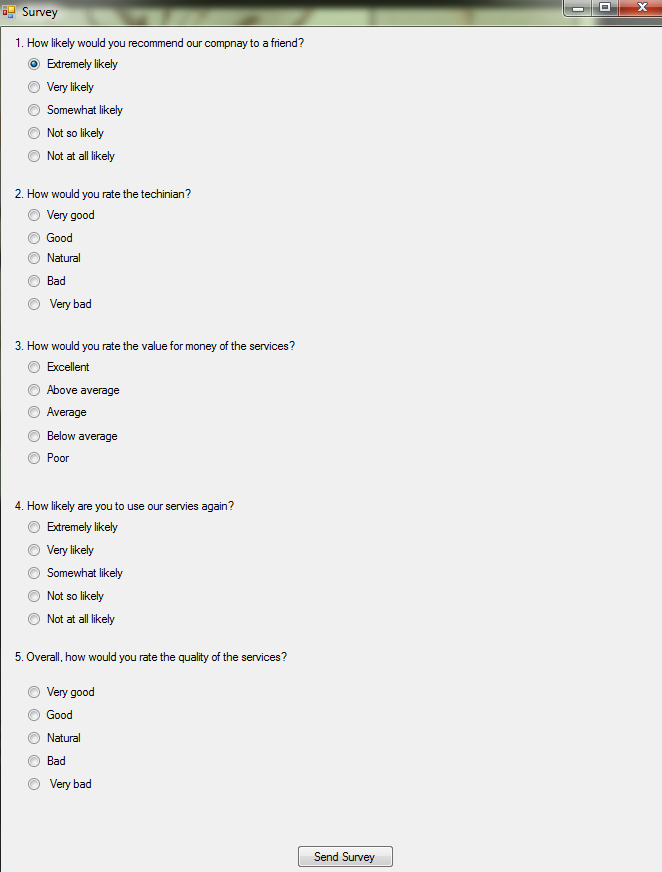
## **Report:**

* Module Name: Report
* Description: The assistant will be able to check the different reports within the system, and such reports being: Invoice Report, Waiting-List Report, Statistics report, Services Request Assignment Report, Outstanding Job Report, Priority Queue Report, Wait- List Report.
* Action: The assistant will have to click on whatever report he needs to access. Once a report is clicked, a determinate metric will show up according to the data traffic present in the system.



## **Survey:**

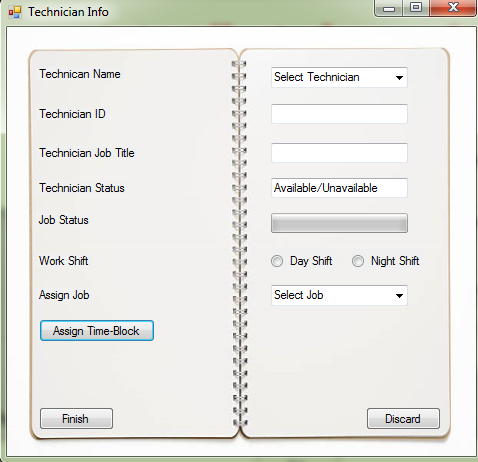
* Module Name: Survey
* Description: The survey module is an interface that will permit the customer to rate a determinate technician once a job has been completed. The customer has the option to decline or accept doing the survey in question. The survey will be made up of 5 different questions, each with a possible answer of: Extremely likely, Very likely, Somewhat likely, Not so likely, and Not at all likely. Once the survey will be completed, the customer will be able to send it to the assistant.
* Action: The technician will have to complete a determinate job, so that the customer can rate the work performed.



## 

## **Technician**

* Module Name: Technician Info
* Description: This interface contains some of the basic information about the technician and it will be used when the assistant assigning job to the technician, it can also check for availability of the technician and job progress
* Action: Assistant will input information about the technician, select the job that need to be assign, click for the day/night shift, click to assign time block, when the assistant is finish, the assistant will either click finish to discard the action

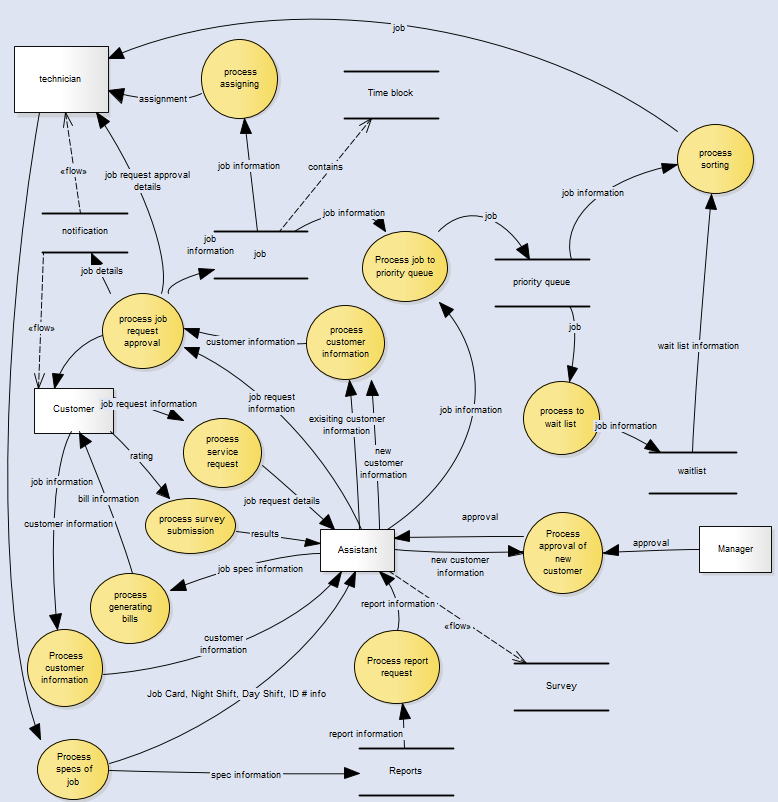


## Exit /Discard

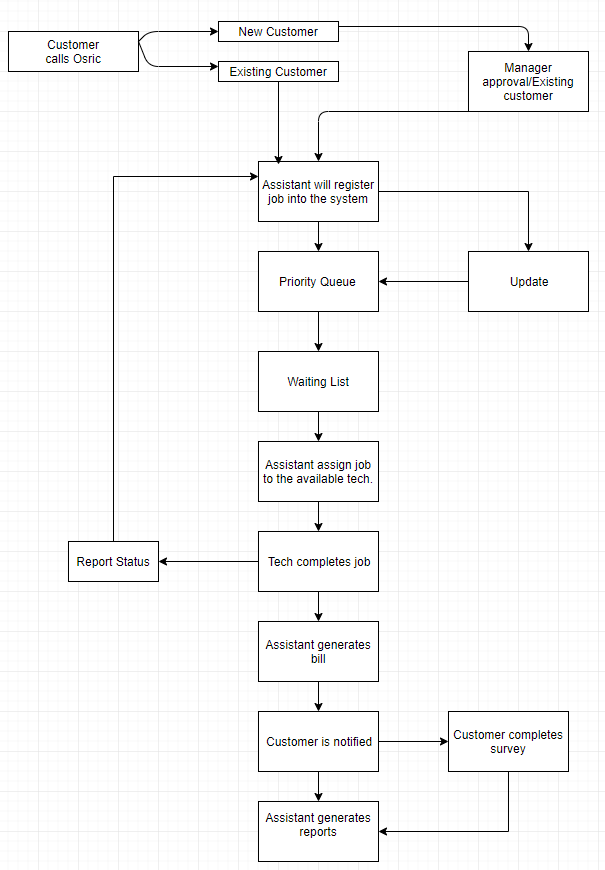
* Description: This action will close the current window
* Action: By clicking on the exit/discard button

# **Other sections will depend on the type of design.**

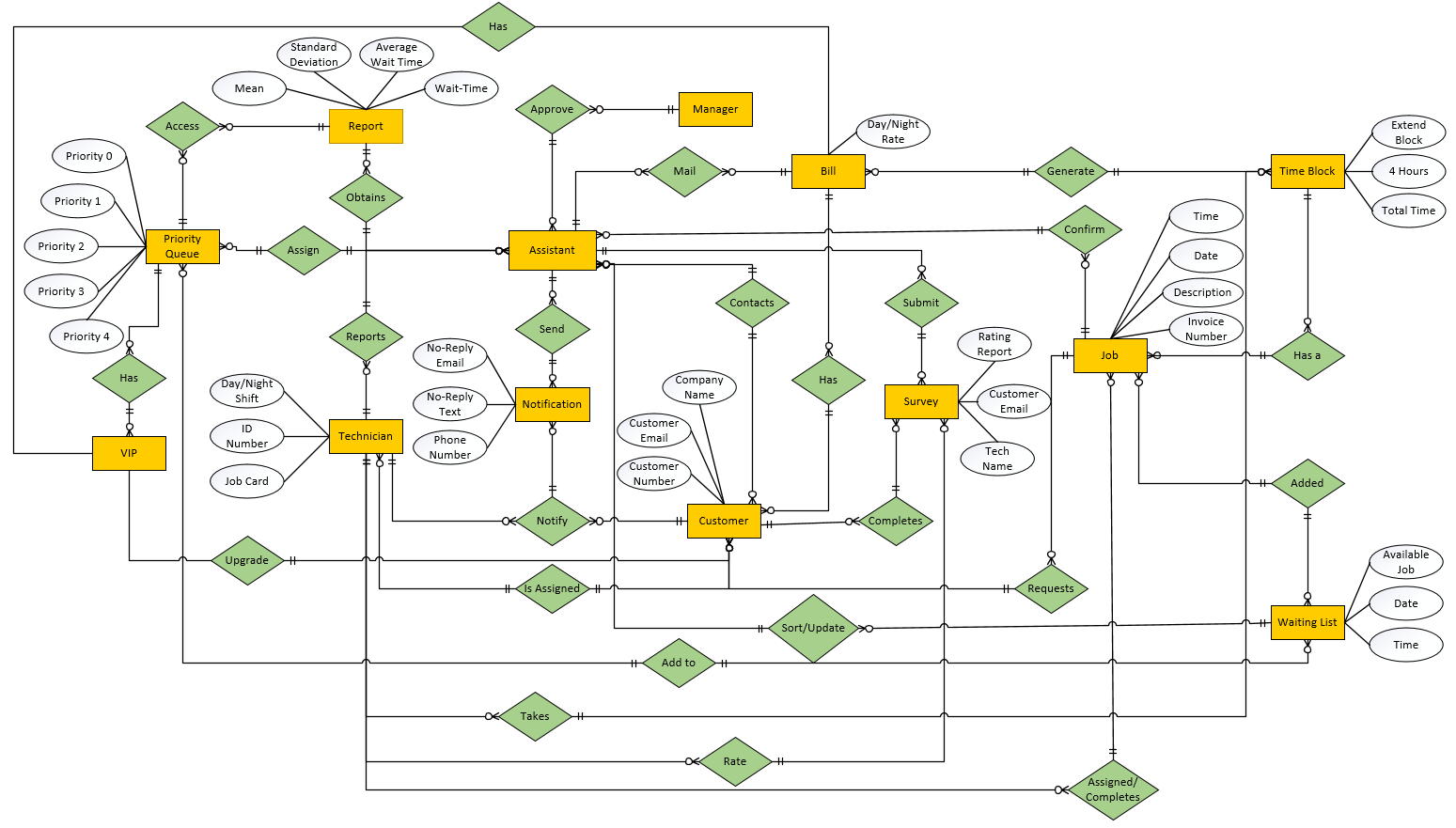
## **Data Flow Diagram**



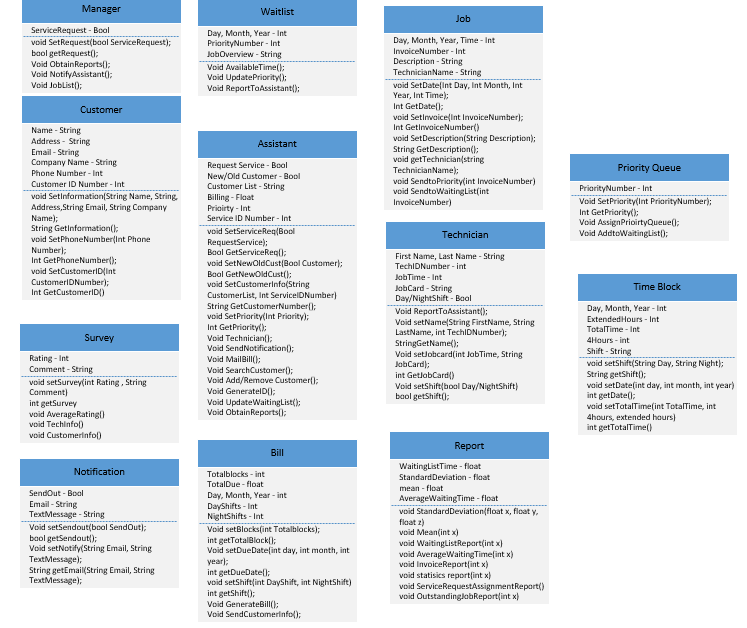
## **State Transition Diagrams**



## **Complete ERD**



* **UML DIAGRAM:**



# **Appendices, References, Index, Glossary, and so on, Access Code**

We’ll Fucking do this boring shit after class

# **Possible References:**

## **Example of Design Document:**

* + <http://robotics.ee.uwa.edu.au/courses/design/examples/example_design.pdf>

## **Chegg Example 1:**

* + <https://www.chegg.com/homework-help/object-oriented-software-engineering-1st-edition-chapter-10-problem-17-solution-9780073523330?trackid=27a8fb89&strackid=7c85125c&ii=10>

## **Chegg Example 2:**

* + <https://www.chegg.com/homework-help/object-oriented-software-engineering-1st-edition-chapter-13-problem-29-solution-9780073523330?trackid=27a8fb89&strackid=7c85125c&ii=15>

## **UML Based:**

* + <https://www.chegg.com/homework-help/object-oriented-software-engineering-1st-edition-chapter-12-problem-10-solution-9780073523330?trackid=27a8fb89&strackid=7c85125c&ii=16>

Data Structure

<https://en.wikipedia.org/wiki/Bucket_sort>

<https://en.wikipedia.org/wiki/Priority_queue>