1 Notification Subsystem

1.1 Overview

The notifications module provides notifications to system users regarding particular system updates that a user would like to be notified about through some medium external to the application.

1.2 External Interface Requirements

1.2.1 User Interfaces

The subsystem will allow for notifications to be pushed to registered users in the form of E-mails. The E-mails will have a descriptive subject associated with it and will be accessible from any device that allows a form of e-mail service.

The subsystem will allow for notifications to be pushed to registered users in the form of SMS's. the SMS's will only have a senders name which will indicate the type of notification and the SMS's content will then go on to expand on the notifications description.

The subsystem will interact with users mostly on a cellular device and a colour screen will be required to get the most out of the application in terms of the E-mail service, SMS's on the other hand are a vital part of most cellular devices and in most cases no extra functionality is required.

1.2.2 Hardware Interfaces

With the use of most cellular devices, a colour screen will be required for the use of e-mails and the device must in all cases be able to display information in a way that is understandable by the user.

The device must have the appropriate hardware in place to allow a form of data connection between itself and the internet.

1.2.3 Software Interfaces

The application will require an e-mail and SMS application in order to receive notifications of this type.

The subsystem will need to have access to a database of users such that they can be added to the user lists for the particular type of notification/s of their preference. These lists will then be referenced when the notifications are being sent to the users.

1.2.4 Communication Interfaces

The subsystem will need to communicate with any other subsystem that wishes to create and send out notifications to the users.

The subsystem will need to communicate with an automated E-Mail service to send out the mass e-mails to the users.

The subsystem will need to communicate with an automated SMS service to send out the mass SMS's to the users.

The users will need to have the appropriate communications interfaces available to allow them to receive notifications. This would include a working cellphone SIM card with a registered cellphone number and/or a working e-mail account that is accessible on a device with a data connection of some form.

1.3 Performance Requirements

The notifications will have to be pushed out in a timely manner such that the time between creation and time of sending of notifications is minimal.

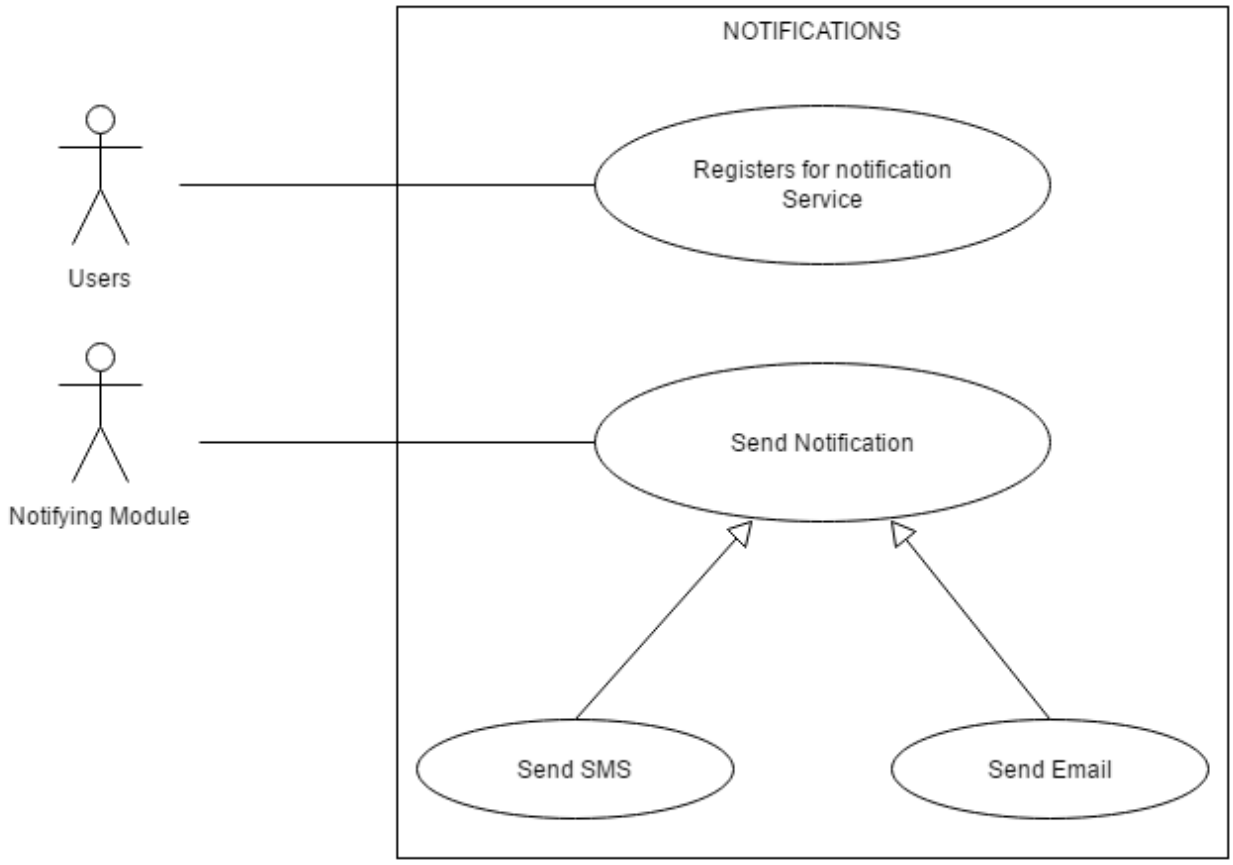
The system will also have to handle sending bulk e-mails and SMS's from multiple notifying modules in a manner that will not cripple the system or put the system at risk.

1.4 Design Constraints

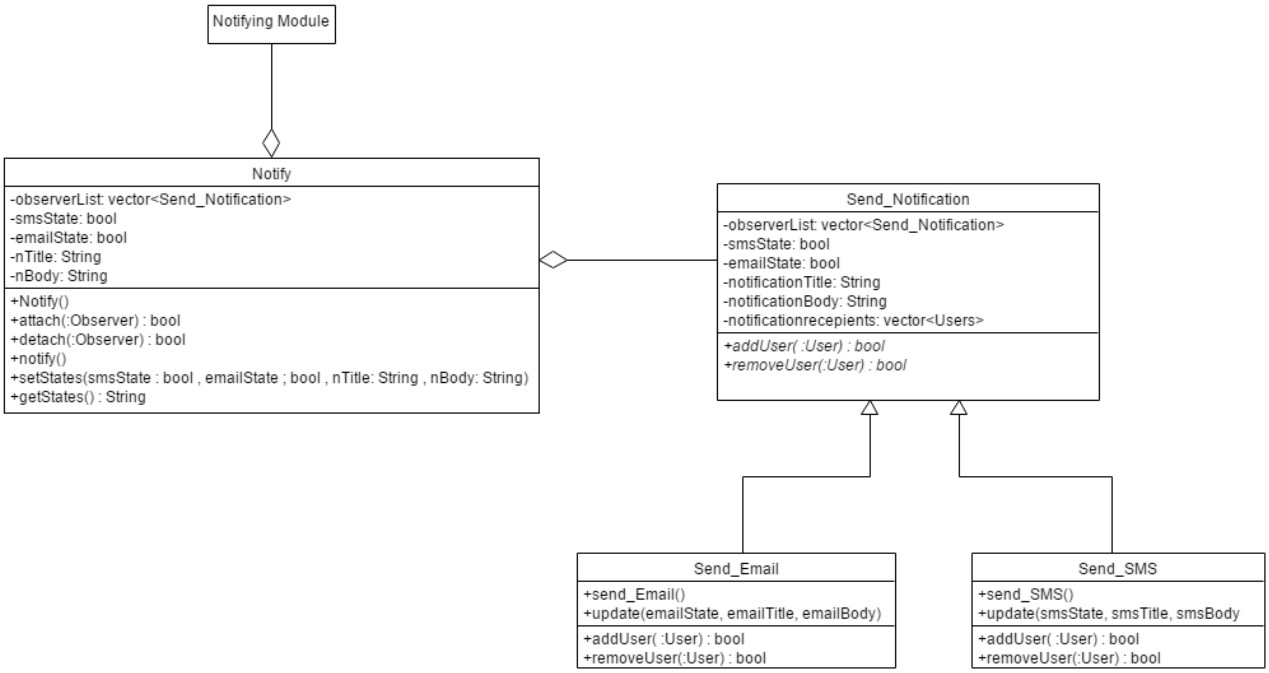
Notifications should not be long such that loading times are affected(e-mail) or that partitioning of messages occurs(sms)
Notifications should be pushed out within a given time frame.
Notifications should conform to the rules of the medium of transport.
No unnecessary data/information should be sent in these notifications.

1.5 Software System Attributes

1.6 UML Diagrams



Notification Use Case Diagram

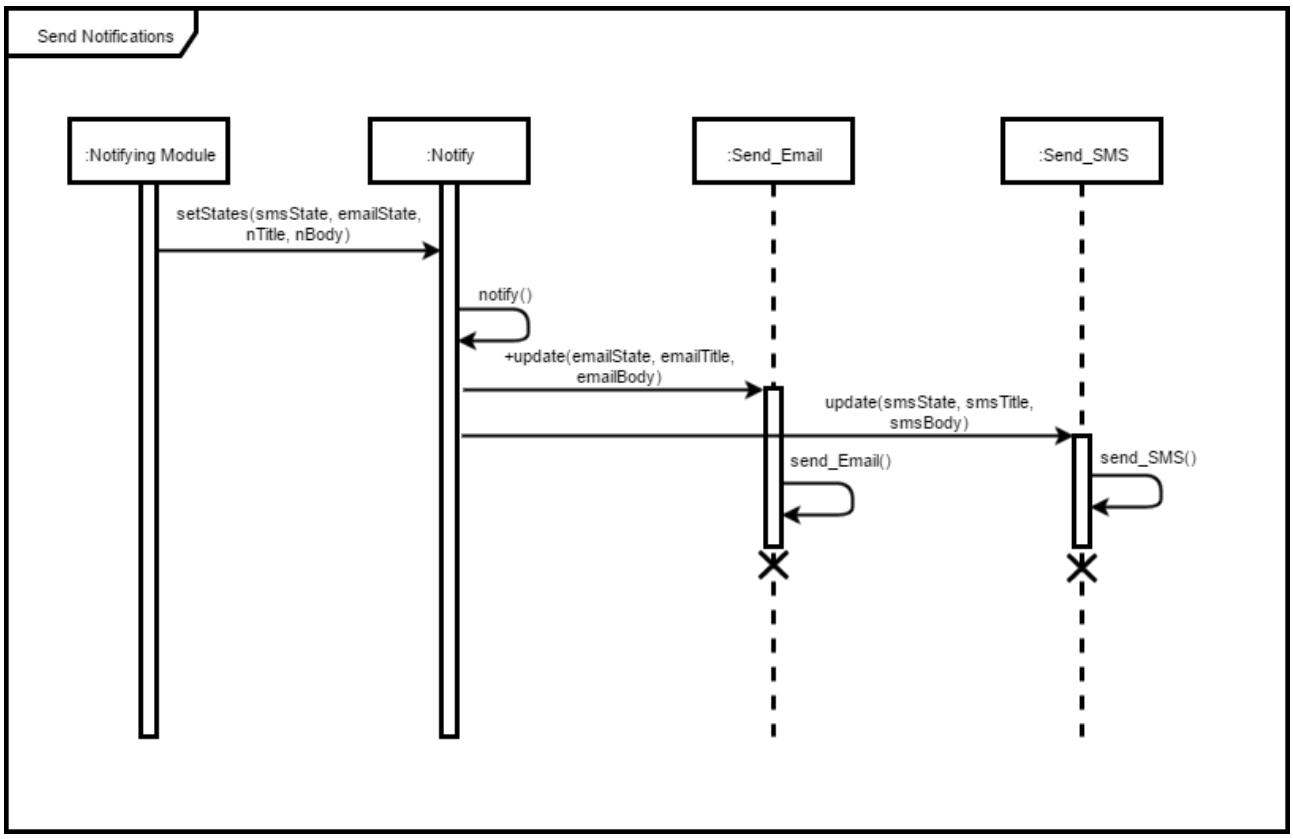


Notification Class Diagram

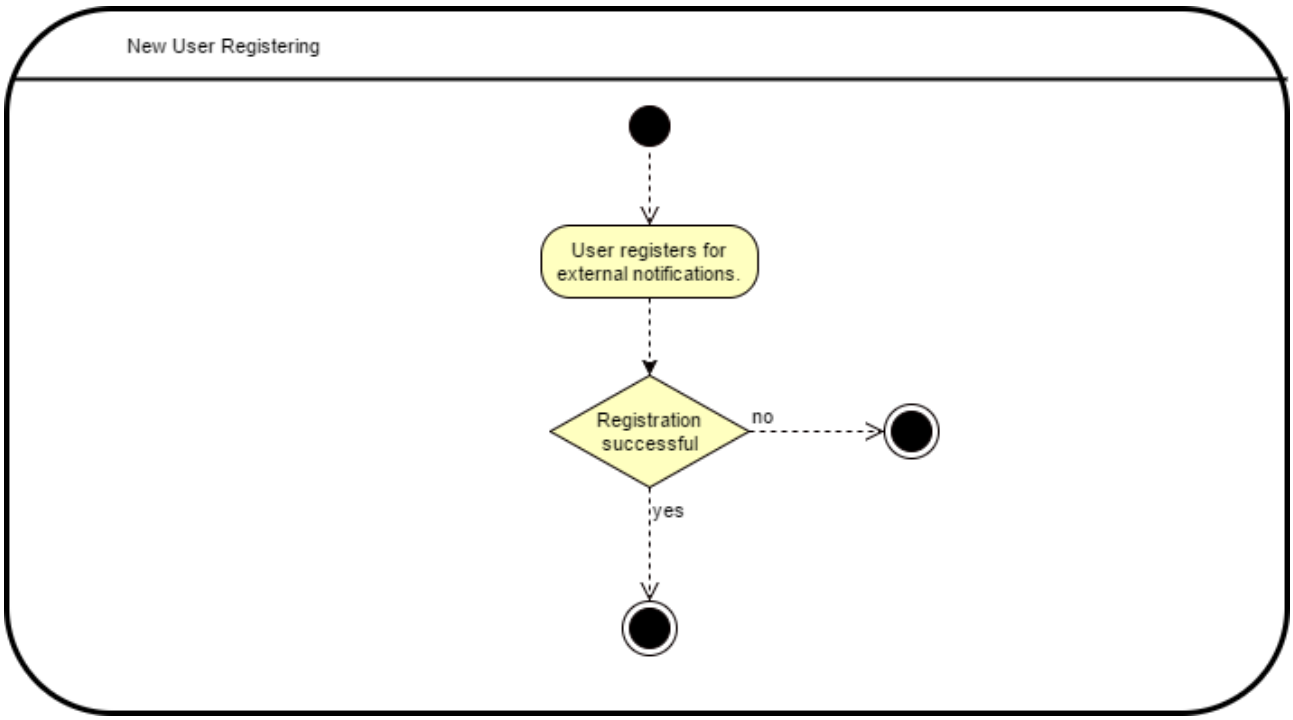
For the notifications subsystem the observer design pattern came to light, as this pattern allows objects to change state and all of its dependents follow suit. This allows a notification state to be used that can update whether or not a notification needs to be sent out. In this case the push method of observer was used. Also evident is the template design pattern that uses the concrete observers to redefine the way in which the notifications are sent. Notifying modules are also given the flexibility to decide which medium/s are used for the notification.

There are methods in the Send Email and Send SMS classes that allow users to be added to the user lists for the notifications to either or both of these mediums. There is also provision for more notification mediums to be added in the future and these can be attached to the Notify class. Should a notification medium be unavailable it can be detached from the Notify class until it is available. The Notify class can be called from any other module that would like to request a notification to be sent out to users which allows for modularity.

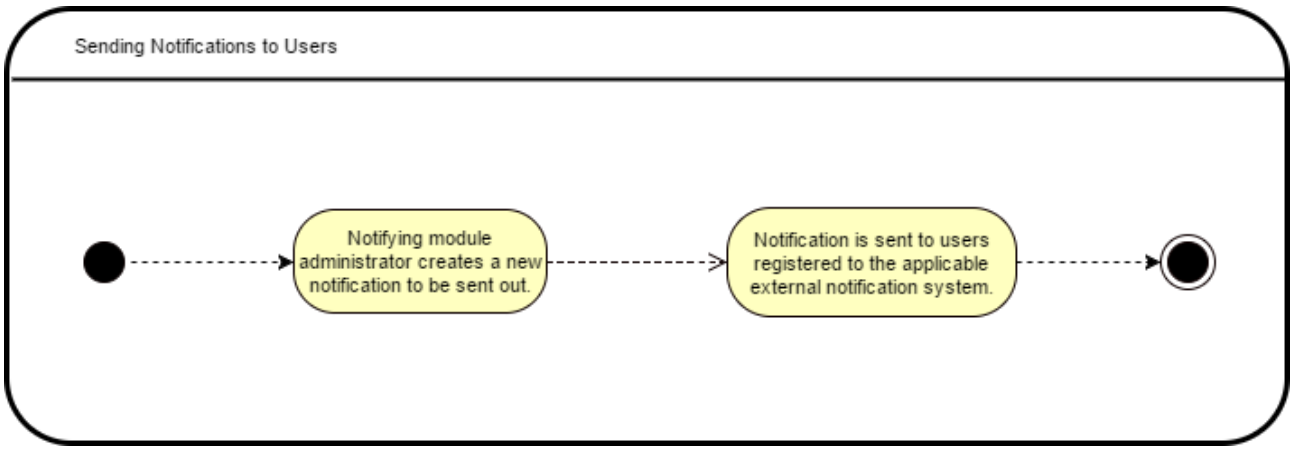
Notify: Subject
Send Notification: Observer(Observer) / Abstract Method(Template)
Send Email: Concrete Observer1(Observer) /Concrete Method(Template)
Send SMS: Concrete Observer2(Observer) / Concrete Method(Template)



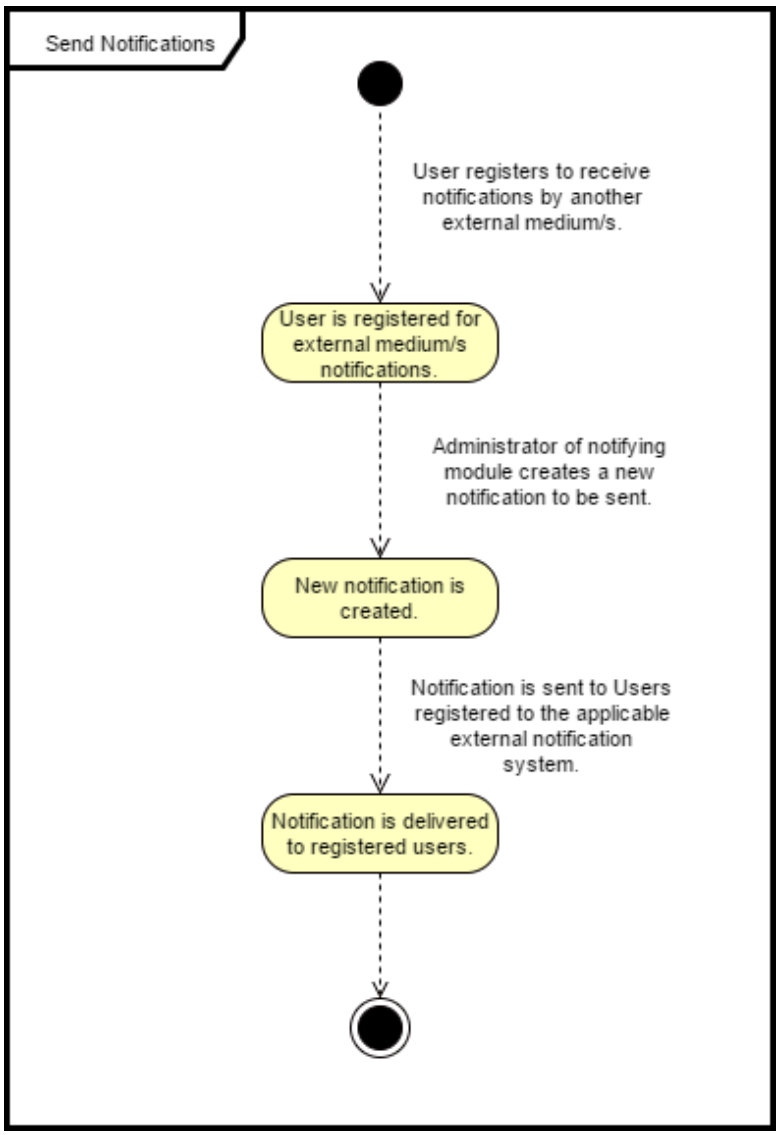
Notification Sequence Diagram



Add User Activity Diagram



Send Notification Activity Diagram



Notification State Diagram