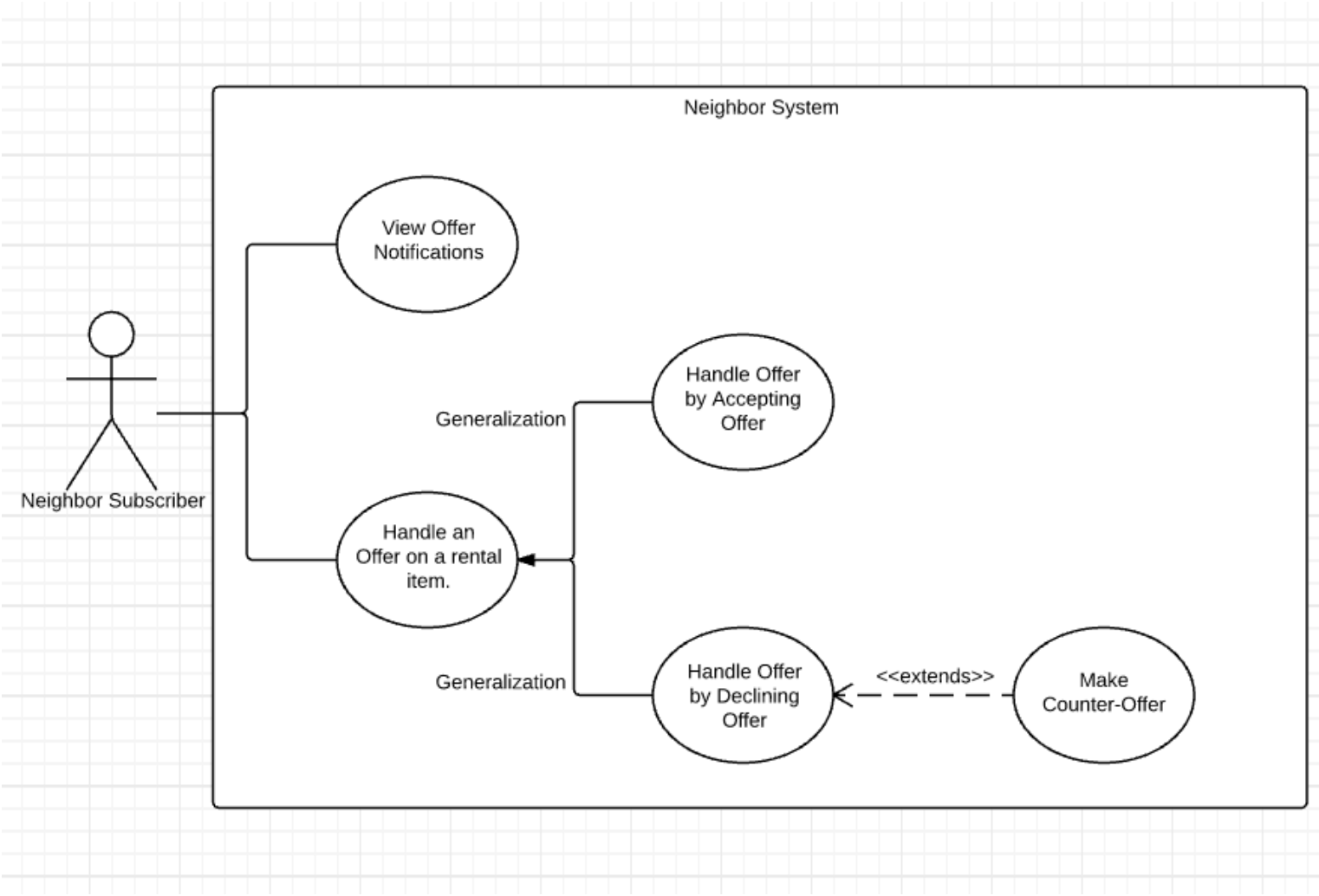
Project Part 2: Extra Credit Opportunity

Brennan McConnell
CSCI 4448: Object Oriented Analysis and Design

1.) Use Case Diagram and Use Case Documents



Note* Regarding the diagram above. Although the 'generalization' relationship is not usually written. We discussed in the grading interview that is would be good to include it for clarity. Additionally, I felt that this scenario did indeed require the generalization relationship since the two ways of handling an offer are specializations of 'Handling an Offer'.

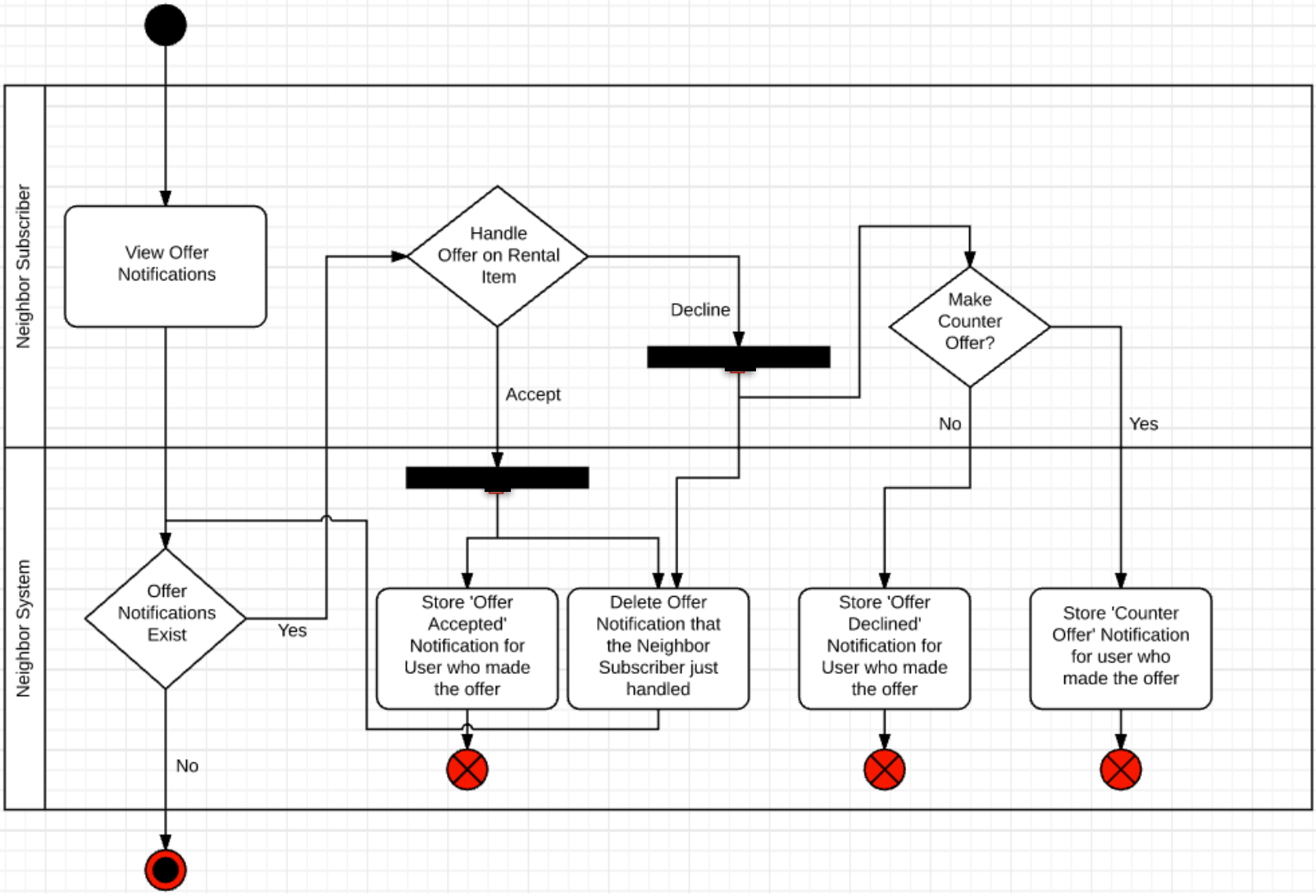
Use Case ID:	**UC-01**
Use Case Name:	**Log onto Neighbor**
Description:	**Users will either Sign-In to access their homepage, or be asked to Sign-Up in order to create a new account which they will then Sign-In to.**
Actors:	**Neighbor User**
Pre-conditions:	**Must navigate to the website at www.Neighbor.com (this domain may change).**
Post-conditions:	**The user will be redirected to their own home screen after logging onto the system.**
Frequency of Use:	**Daily by many different Neighbor Users**
Flow of Events:	**Actor Action : System Response** **1.) Navigate to Neighbor Website : Display index.html to User** **2.) Click either Sign-up or Sign-in : Display proper form** **3.) Provide information and enter : Verify Information. If no account found for a sign-in, redirect to the sign-up form. If information is valid, log the user into their personal home page.**
Variations:	**None**

The log-in use case document is provided even though the diagram is not shown. This is because the directions specified to include related use case documents for use cases needed to complete the use case diagram, and seeing as being logged onto the Neighbor systems is a pre-condition for use cases depicted in the diagram above, I felt it important to include.

Use Case ID:	**UC-03**
Use Case Name:	**View offers on your rental items made by other Neighbor Users.**
Description:	**A User should be able to view Offer Notifications (offers made by fellow Neighbors on a rental item the user has posted).**
Actors:	**Neighbor User**
Pre-conditions:	**User must be logged onto the system and currently on their home page.**
Post-conditions:	**Upon selecting the 'View Offer Notification' tab, relevant information will be updated and displayed on the screen for the user to view.**
Frequency of Use:	**Multiple times daily, by multiple users.**
Flow of Events:	**Actor Action : System Response** **1.) Click on the 'View Offer Notifications' tab : Display the corresponding information retrieved from the database on the web page.**
Variations:	**None**

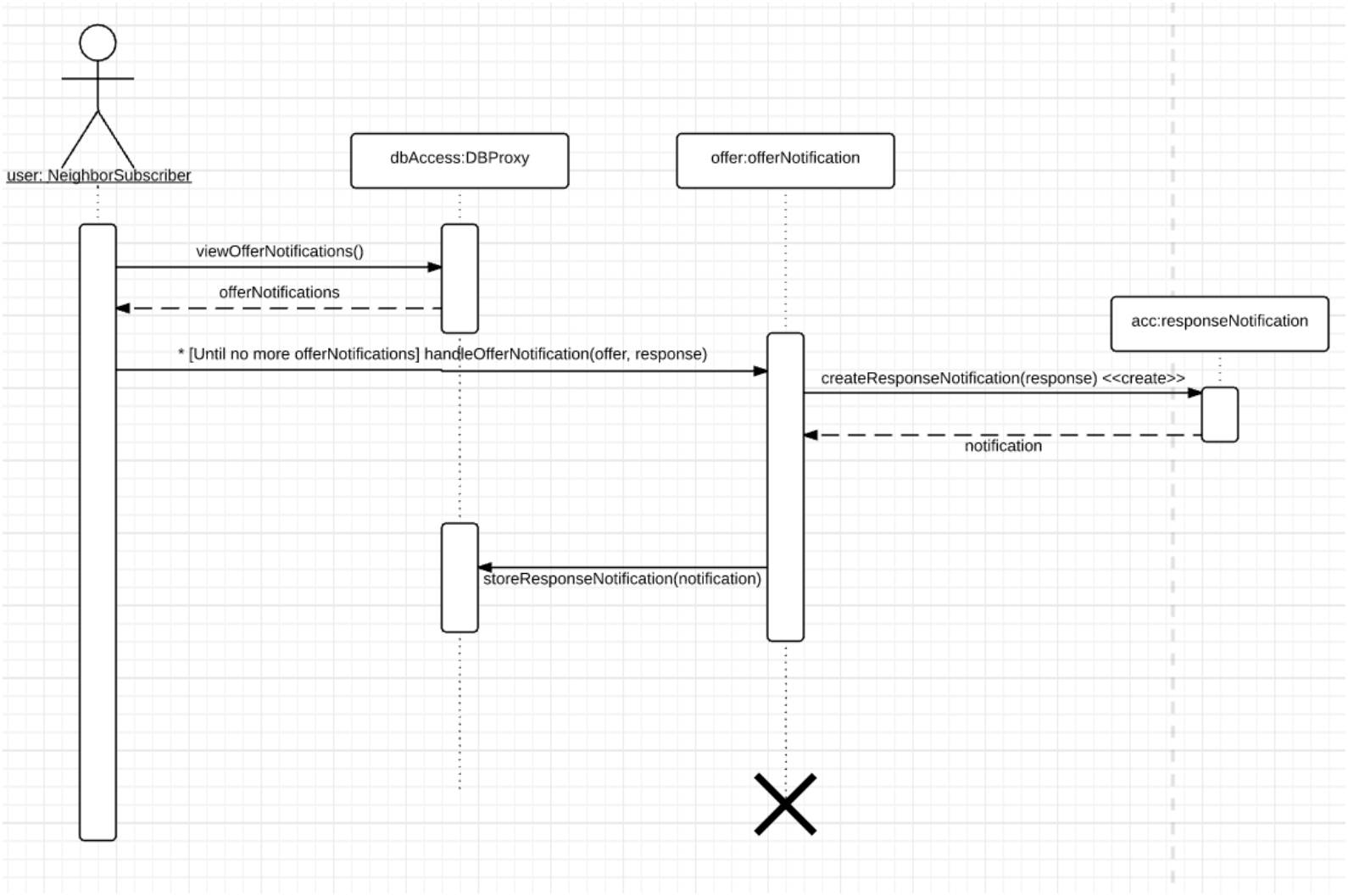
Use Case ID:	**UC-07**
Use Case Name:	**Handle an offer on one of your rental items.**
Description:	**Users should be able to accept or decline an offer on one of their available rental items made by another Neighbor User. If they decline they offer, the owner should have the option to respond with a counter-offer during declining.**
Actors:	**Neighbor User**
Pre-conditions:	**User must be logged onto the system and be on their homepage currently viewing their offer notifications.**
Post-conditions:	**Depending on the owner's action (accept or decline), a notification will be sent back to the user who sent the offer. If they accept the offer, their contact info will be sent back to the rentee.**
Frequency of Use:	**A few times daily, by multiple users.**
Flow of Events:	**Actor Action : System Response** **1.) User will see the notification in a list form, and be able to click 'Accept' or 'Decline' : The Renter's response will be sent back as an alert to the Rentee.**
Variations:	**None**

2.) Activity Diagram



Note* The red circle with the X inside represents a 'Flow Final Node' which was introduced in UML 2.0. Reference here: http://www.uml-diagrams.org/activity-diagrams-controls.html. This terminates a flow but does not terminate the activity and has no effect on other flows in the activity. This is used above, as the Neighbor Subscribers response towards the offer notification is stored in the system and that particular flow is terminated, while the other flow continues to see if there are still notifications that the user has not yet handled.

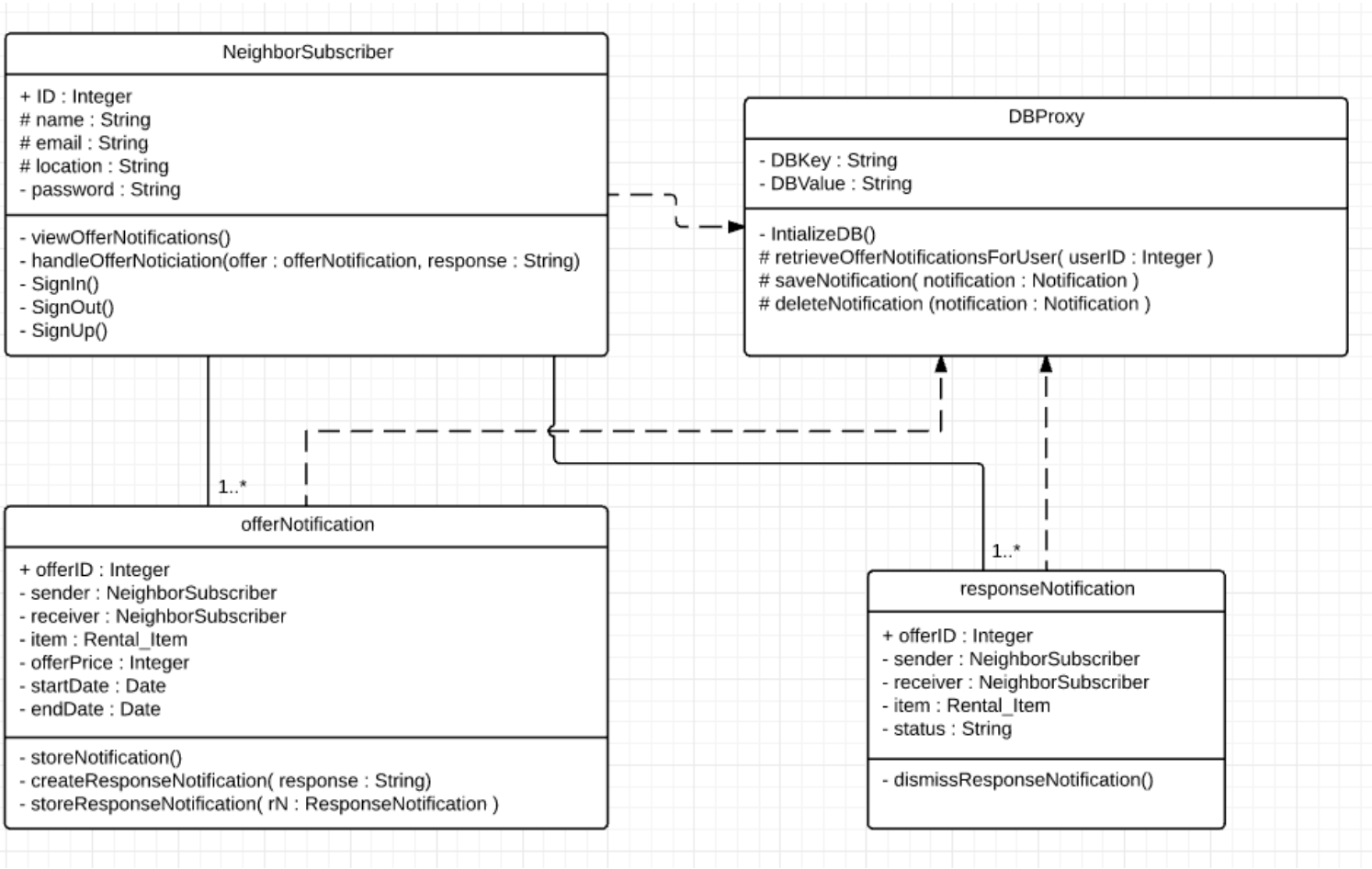
3.) Sequence Diagram



As you can see this diagram does indeed match the activity diagram and the use case diagram. The user retrieves his notifications from the database (Neighbor system), and handles them one by one. His response can vary and the response is handled within the offerNotification object. This object then generates an appropriate Notification and stores it back in the system, following this, the offerNotification that has been handled is destroyed.

Note* An important distinction arises here between this and the activity diagram. In the activity diagram, we have activity's for storing acceptedOffer, declinedOffer, and counterOffer. And there is a distinction among storing these 3 different objects. However, they are all responseNotifications simply with different status' as can be seen in the class diagram and sequence diagram.

4.) Class Diagram



The classes that exist above are the 4 classes that we have used or will need throughout the use case diagram, activity diagram, and sequence diagram. **A very important observation** is to be made that the Rental_Item class is not depicted in our class diagram. This is because this use case never interacts with a rental item, only the notifications regarding said rental items, thus in proper accordance with the directions, the use case modeled throughout this document does not require the Rental_Item class.

A note on relationships. NeighborSubscriber, OfferNotification, and ResponseNotification all have a dependency on the DBProxy (the way to communicate with Neighbors Database), and there is a semantic relationship because all these classes will use DBProxy. Additionally, there is associations between both types of notifications and the NeighborSubscriber.