UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

COS301 Mini Project Functional Requirements Specification

Group 2a

Matthew Gouws *u11008602*
Neo Thobejane *u11215918*
Roger Tavares *u10167324*
Rendani Dau *u13381467*
Ivan Henning *u13008219*
Szymon Ziolkowski *u12007367*
David Breetzke *u12056503*
Keagan Thompson *u13023782*

Here's a link to Github.
`https://github.com/MatthewGouws/COS301_2a`

Final Version
February 27, 2015

1 History

Date	Version	Description
17-02-2015	Version 0.1	Document Template Created
17-02-2015	Version 0.1.1	Auto title page added
17-02-2015	Version 0.2	UP Logo Added
18-02-2015	Version 0.2.1	Intro, Purpose, Conventions and Description Added
19-02-2015	Version 0.2.2	Added Skeleton for Required functionality
19-02-2015	Version 0.2.3	Added Bulk of Points - Roger
20-02-2015	version 0.2.4	Added point Headings - David
20-02-2015	version 0.2.5	Added Use Cases - Matthew
20-02-2015	version 0.2.6	Added Use Cases - Roger
22-02-2015	version 0.2.7	Added Details on points - David
23-02-2015	Version 0.3	Updated Use Case prioritization and fixed newPage with Clearpage
23-02-2015	version 0.3.1	Added points - Rendani
23-02-2015	version 0.3.2	Added points - Neo
23-02-2015	version 0.3.3	Added Points - Ivan
24-02-2015	version 0.3.4	Added Diagrams - Ivan
24-02-2015	version 0.3.5	Changed Original Layout removed unnecessary sections
24-02-2015	version 0.3.6	Added Diagrams - David
24-02-2015	version 0.3.7	Added Diagrams - Roger
24-02-2015	version 0.4	Added Project Scope
24-02-2015	version 0.4.1	Added Diagrams - Szymon
25-02-2015	version 0.4.2	Added points - Keagan
25-02-2015	version 0.4.3	Added Diagrams - Neo
25-02-2015	version 0.4.4	Added points - Ivan
25-02-2015	version 0.4.5	Fixed Clearpage Newpage problems
25-02-2015	version 0.4.6	Added Domain Model - David
25-02-2015	version 0.4.7	Added Domain Model - Syzmon
26-02-2015	version 0.5	Added Combined, Master, Use case
26-02-2015	version 0.5.1	Added Diagrams - Neo
26-02-2015	version 0.5.2	Added Diagrams - Ivan
27-02-2015	version 1.0	Version 1 Released

Contents

List of Figures

2 Introduction

The Computer Science Education Didactic and Application Research (CSEDAR) from the university of Pretoria. Have approached us in building a software platform to create a collaborative community by means of an online discussion. To aid students to excel in problem solving as group. Such a tool already exists for students to use however this tool lacks certain functionality, Lecturer - Student interaction, often students are unaware of who is higher ranked in terms of the course (Teaching Assistant, Tutor, Lecturer & student).

2.1 Purpose

This document serves to present the clients requirements on a functional level by use of use-case diagrams, Domain models, pre- & post conditions as well as possible input-output pairs.

2.2 Document Conventions

A ranking system of importance is used for the functional requirements based on a 'star' system with
***** - Critical
**** - Important
*** - Somewhat important
** - Nice to have
* - Not considered

2.3 Project Scope

The scope of this project is to create a Buzz Space system to integrate into the Computer Science Department's website at the University of Pretoria. This software solution will provide an online forum that is both organized and interactive to engage the students in their studies. The project can then later also be expanded to multiple universities and institutions.

2.4 References

Tutorial on Use case diagrams - `http://www.tutorialspoint.com/uml/uml_use_case_diagram.htm`

3 System Description

Buzz will be a complete software unit which is to be integrated seamlessly with existing web servers to be used by courses to encourage the use of online discussion. Users will be able to climb ranks up the online discussion forum and achieve more functionality as they progress. The system will also have certain functionality incorporated into awarding users marks if required so by the teaching staff. Teaching staff will also be able to archive and summarise threads. Users will also only be able to post in threads which pertain to them at that specific time, thus having a thread become 'Ancient' and hence archived no new information may be added to said thread.

4 Functional Requirements

4.1 Use Cases

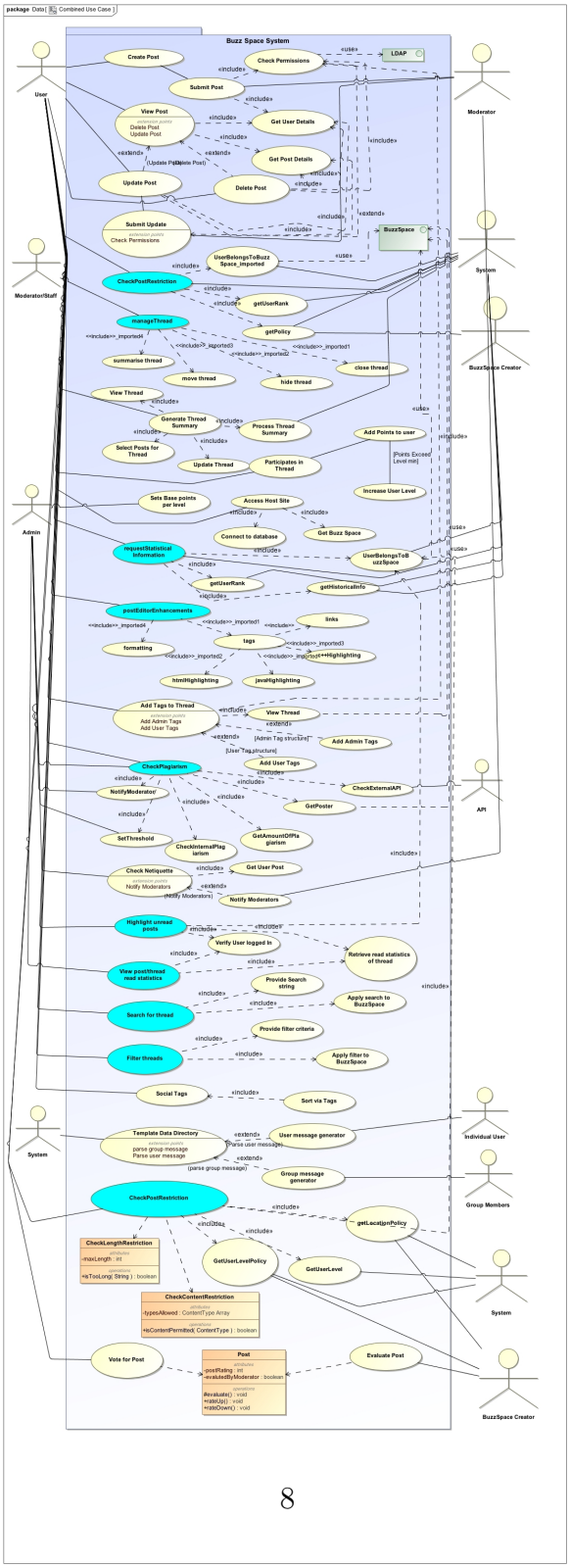


Figure 1: Combined Use Case of Entire System

4.1.1 Use Case Prioritization

Critical:

- Users must be able to create, Read, update and delete posts.
- Users must only be able to post on specific threads/levels based on their level.
- Staff should be able to summarize, close/hide threads and move posts around.
- Buzz must integrate seamlessly with any host site.

Important:

- The system should keep track of what posts have been read by a specific user and highlight unread posts.
- The system should allow for social tagging, allowing users to easily locate threads on what they are looking for.

Important-Nice to have:

- The post length should be restricted. Users of higher levels can post and embed Pictures, Videos etc.
- A users status should automatically update based on their participation.
- Post should be able to be up-voted, shown higher in the thread.
- Statistics should be available for each student displaying their marks, and visual reporting of their level.

Nice to have:

- Semi-automated thread summary creation.
- Create an automated template based on Message and other users.
- Provide searching and filtering.
- Enhancement of posts, such as Rich-text-format editor.
- Apply organization of content based on tags, base structure or ownership structure.
- Detect if a post is plagiarised.
- Detect if Netiquette rule are broken.

4.2 Required Functionality

The Following system processes detail the functional requirements of the individual points.

1. Users must create, Update and delete posts.
    (a) Create Post
        i. Elaboration - In order for the user to communicate with others in the Buzz space, (s)he need to be able to create posts. User must have sufficient permissions to post in certain areas of the Buzz space. This function is not available to guests.
        ii. Importance - *****, Critical
        iii. Dependency level - With out the ability to create posts users will not be able to communicate with one another, which makes the Buzz space useless.
        iv. Pre-conditions
            A. User must have necessary permissions.
            B. Buzz space must exist.
            C. User must be registered.
            D. User must be logged in and not a guest.
        v. Post-conditions
            A. Post will be successfully created.
        vi. Requester - Client

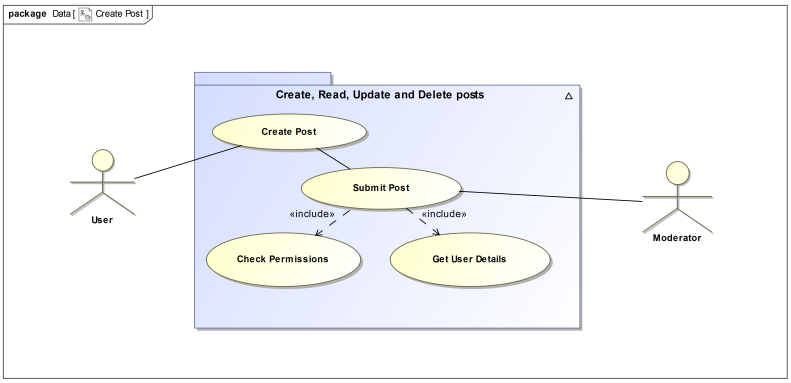


Figure 2: Create post use case

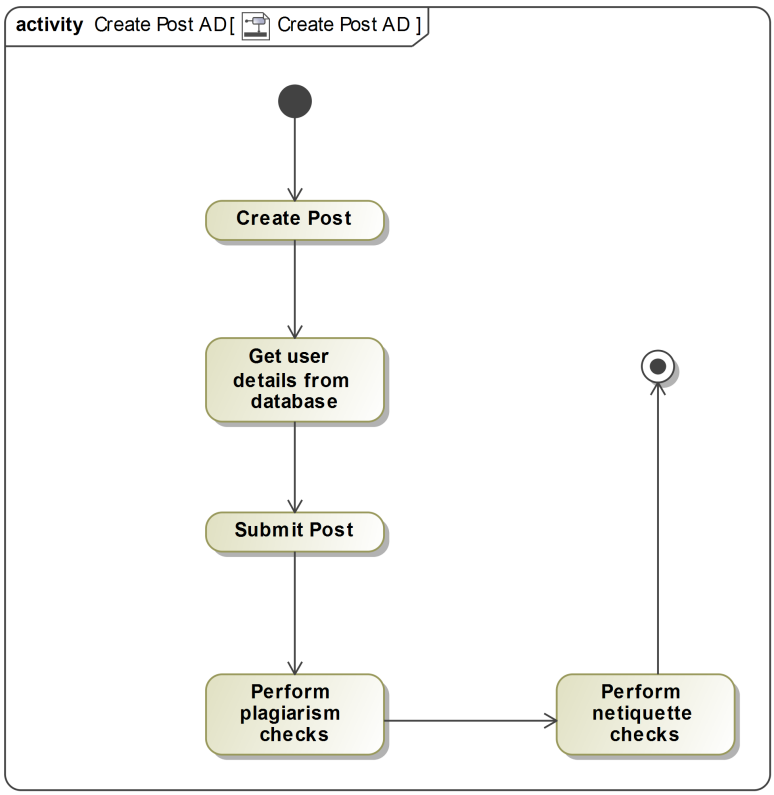


Figure 3: Create post activity diagram

(b) Read Post

i. Elaboration - This function allows users read posts made by other users of the Buzz space.

ii. Importance - *****, Critical

iii. Dependency level - This function is very important because without it, the Buzz space is useless as users will not be able to read each others posts, which defeats the purpose of the Buzz space.

iv. Pre-conditions

A. Buzz space and post must exist.

B. Access to Buzz space.

v. Post-conditions

A. If the user is not a guest post will be marked as read.

vi. Requester - Client

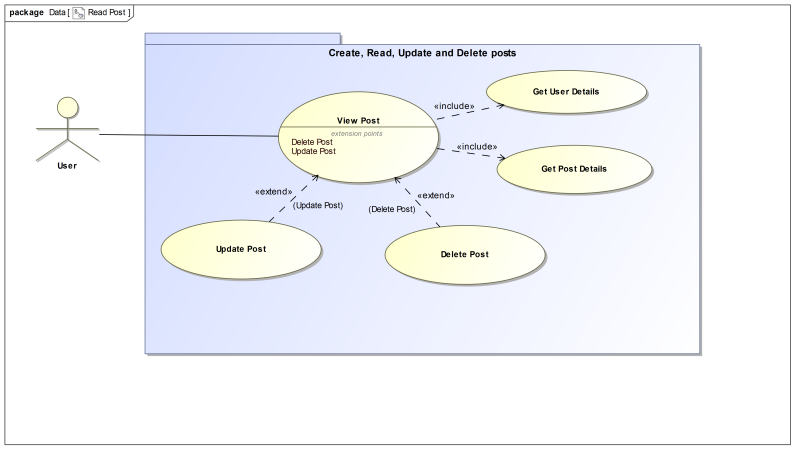


Figure 4: Read post use case

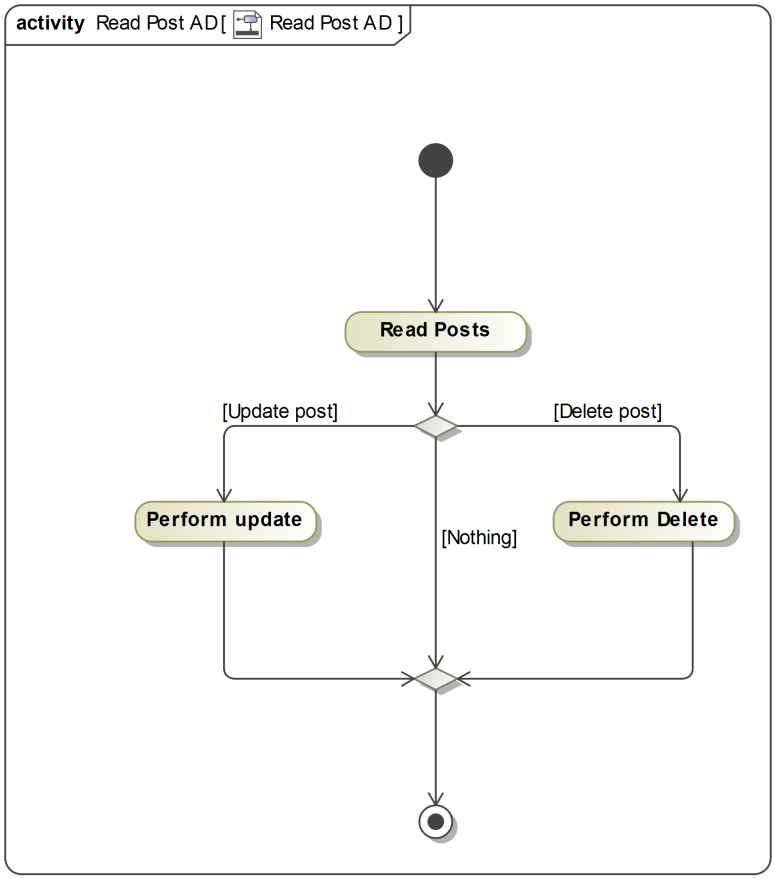


Figure 5: Read post activity diagram

(c) Update Post

i. Elaboration - This function allows the user to update his/her post. It is only available to certain users such as the owner of the post or the moderators.

ii. Importance - *****, Critical

iii. Dependency level - Updating posts is another important process needed in the Buzz space. People often make mistakes in their posts and have to correct it. The ability to update the post will help achieve that.

iv. Pre-conditions

   A. User must be either the owner of the post or the moderator of that Buzz space.

v. Post-conditions

   A. Post will be updated.

vi. Requester - Client

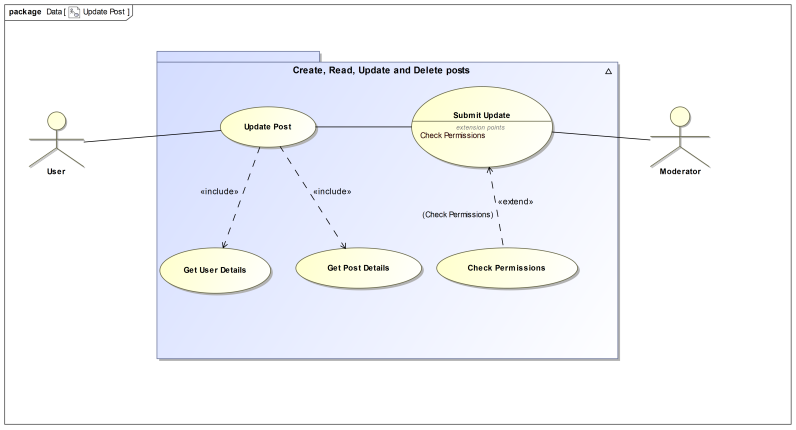


Figure 6: Update post use case

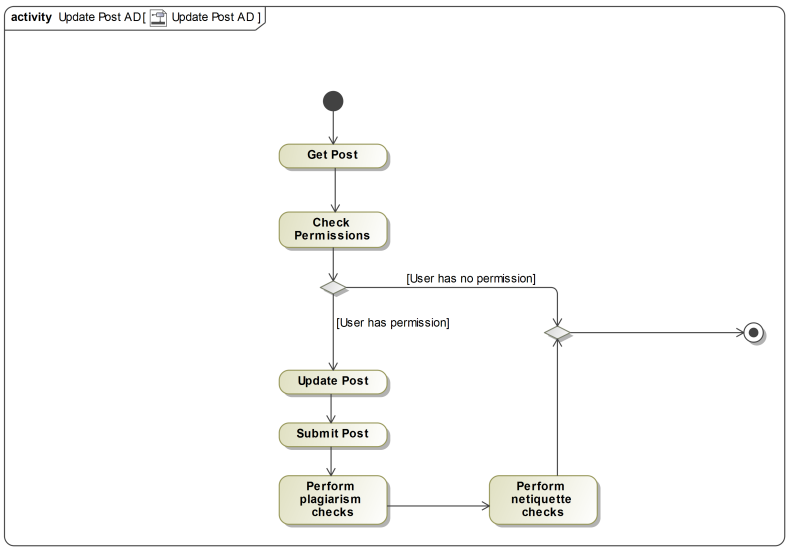


Figure 7: Update post activity diagram

(d) Delete Post

i. Elaboration - Deleting posts allows for users to remove their post from a Buzz space and also allows the moderators to delete posts that they think are not suitable for the Buzz space.

ii. Importance - *****, Critical

iii. Dependency level - The deleting of a post is also very important to a Buzz space. Owners of a post would like to remove their post if they think it is not relevant to the topic and moderators should be able to delete user posts if they think the post is not relevant.

iv. Pre-conditions

   A. User must be either the owner of the post or the moderator of that Buzz space.

   B. Post must exist.

v. Post-conditions

   A. Post will be marked as deleted and hidden from Buzz space. Post is not physically deleted from database.

vi. Requester - Client

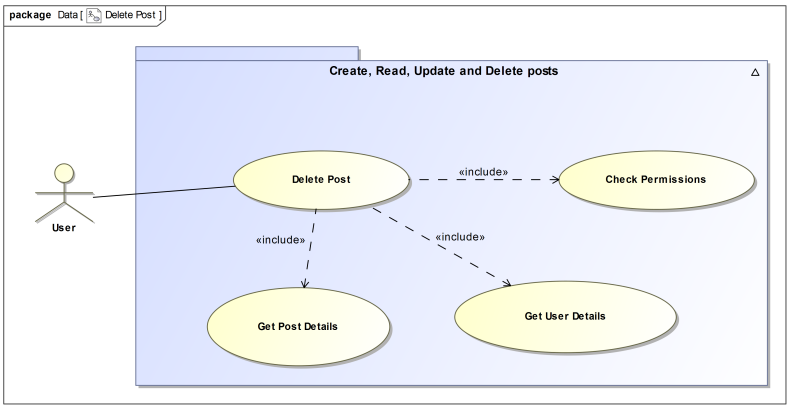


Figure 8: Delete post use case

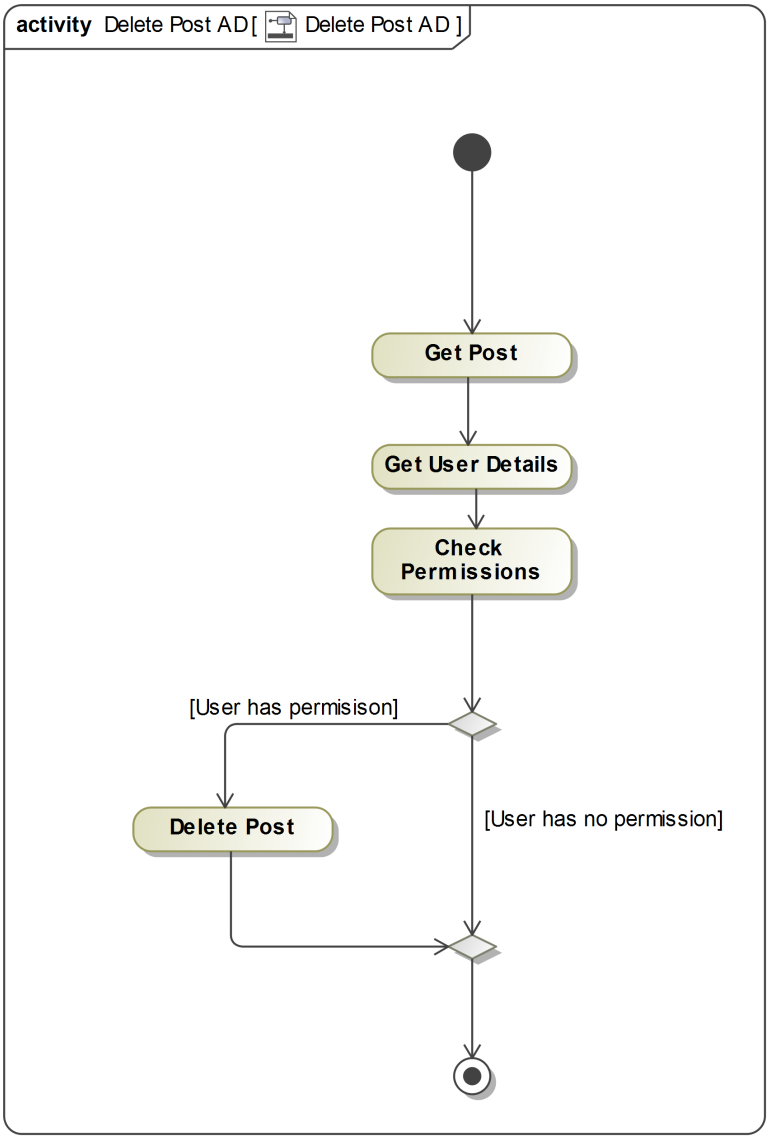


Figure 9: Delete post activity diagram

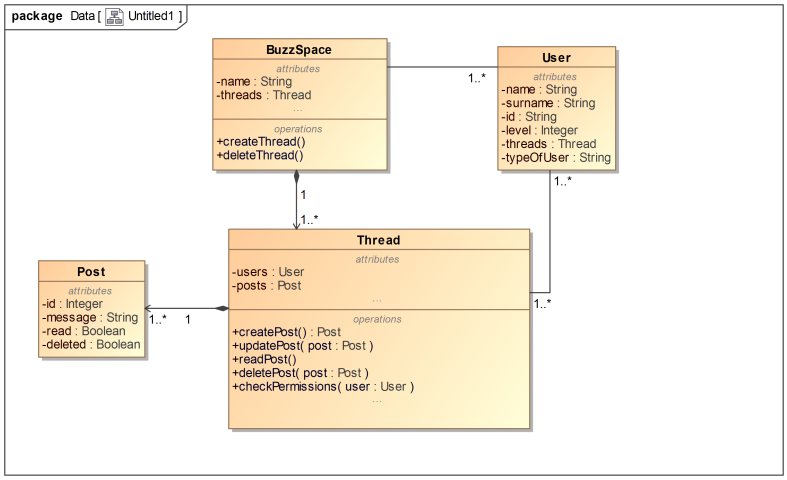


Figure 10: Create, read, update and delete posts class diagram

2. Keep track of who has read what and highlight unread messages for each user.

   (a) Elaboration - This process will allow the student/user to see which thread he/she still has to read by applying some form of emphasis on the thread. It will also allow a lecturer/board moderator to view read statistics of a thread such as how many students have read a thread and who those students are.

   (b) Importance - ****

   (c) Dependency level - Highlighting unread posts is important because it will allow users to efficiently follow discussions without having to find specific points where they last read and this in turn will help users post replies faster keeping discussions relevant. It is also important that the lecturer/moderator can view statistics of who has read which posts as this will give them an idea of how many students follow discussions and who those students are.

   (d) Pre-conditions

      i. A user must be logged in and registered for that particular buzz space.

      ii. A user must be logged in and have elevated privileges, such as lecturer/moderator status, to view read statistics of a post

   (e) Post-conditions

      i. User is presented with formatted view highlighting unread threads

      ii. User(lecturer) is presented with read statistics of a particular thread

   (f) Requester - Client

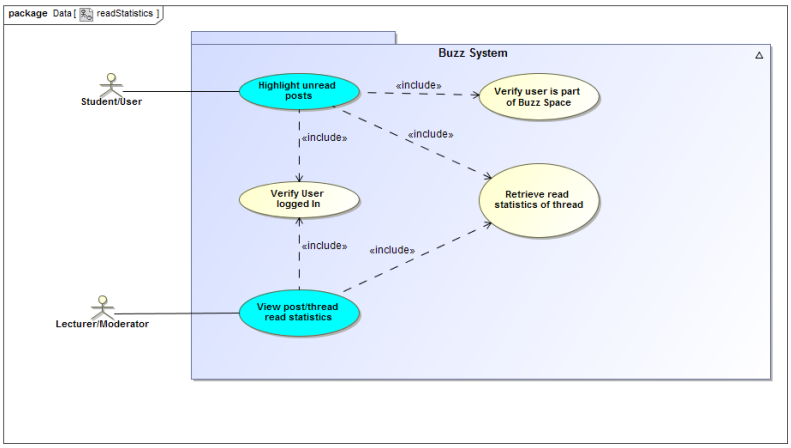


Figure 11: Highlighting/Read Statistics Use Case

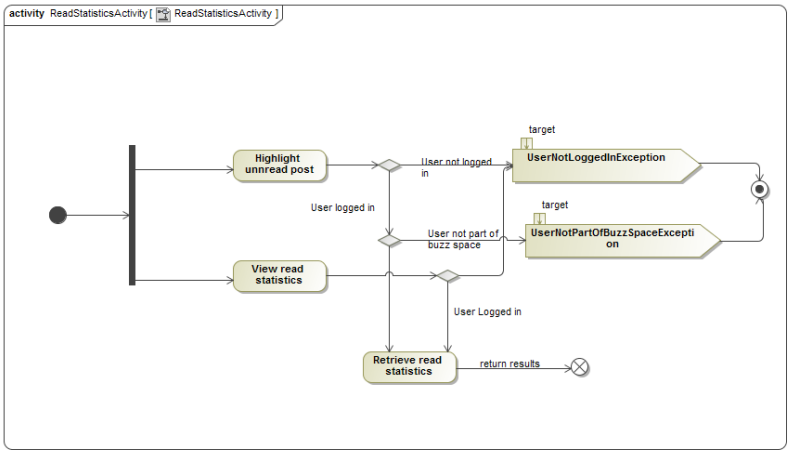


Figure 12: Highlighting/Read Statistics Activity

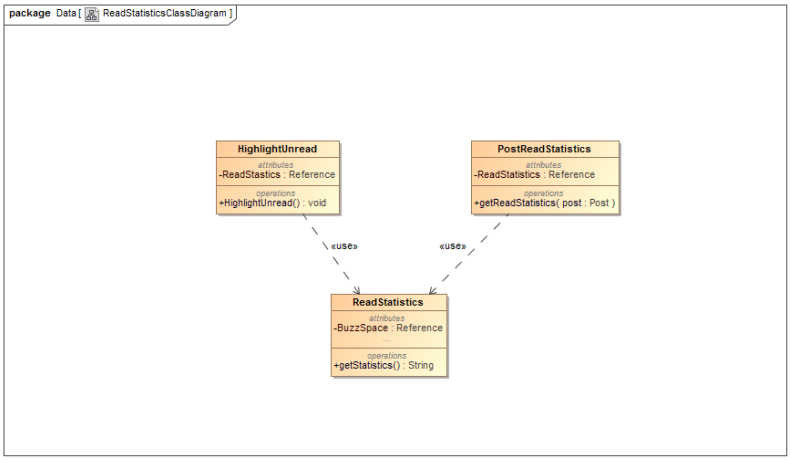


Figure 13: Highlighting/Read Statistics Class Diagram

3. Restrict the length of messages and the type of content allowed in messages based on the level where it is posted as well as on the status of the user posting the message.

   (a) Elaboration - Restrictions on the length of message should be configurable by policy, according to user level and post location. This will allow Buzz Space creators specific control over content type and post length for users of different level/rank.

   (b) Importance - ***

   (c) Dependency level - Relies on the ranking system to be implemented and provide a user's level/rank. Two policies must be set by the Buzz Space creator, one for message length according to level/rank, and one for permitted content types according to level/rank.

   (d) Pre-conditions

      i. User must belong to a Buzz Space.

      ii. User must attempt to post.

      iii. Policies must be acquired.

   (e) Post-conditions

      i. User successfully posted a post that contains only permitted content types.

      ii. User successfully posted a post that contains less than or equal to the permitted character length.

   (f) Requester - System

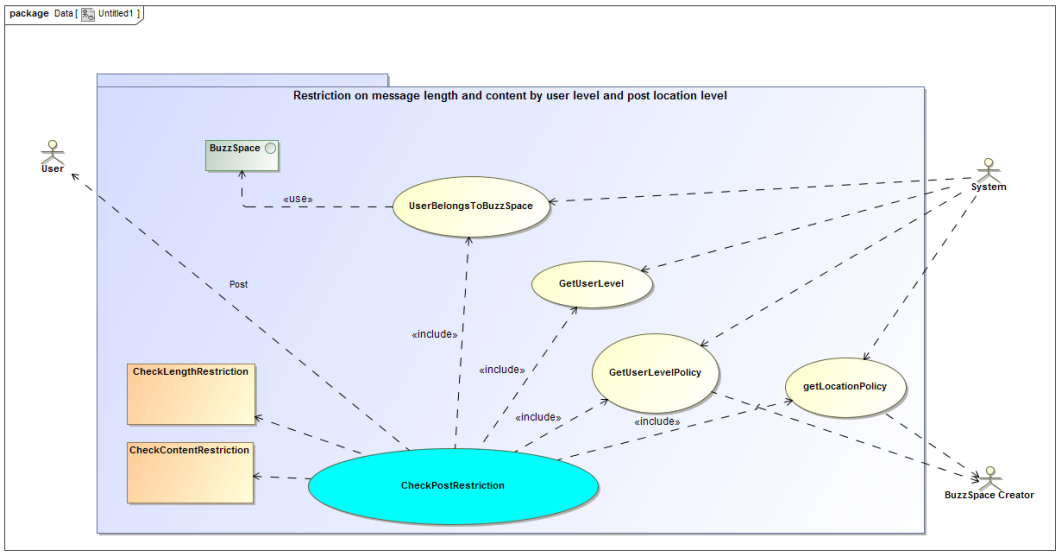


Figure 14: Message Length and Content Restriction Use Case

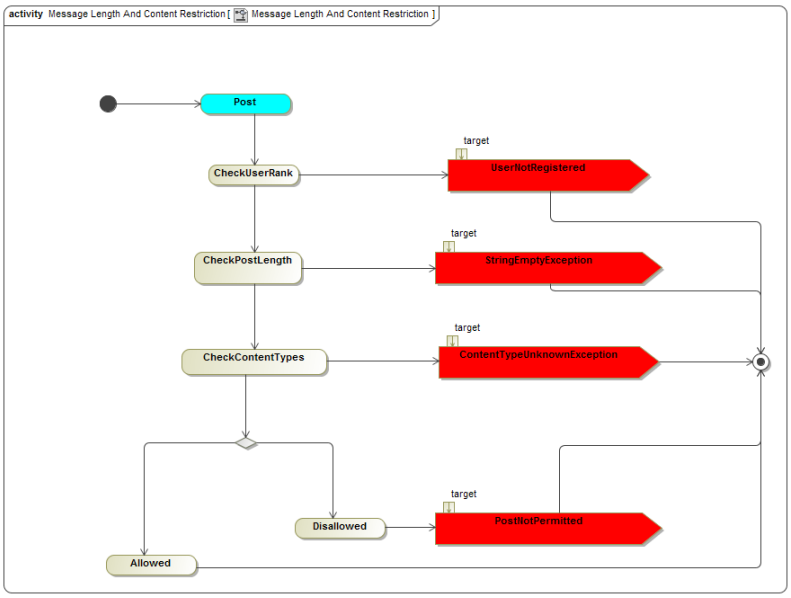


Figure 15: Message Length and Content Restriction activity diagram

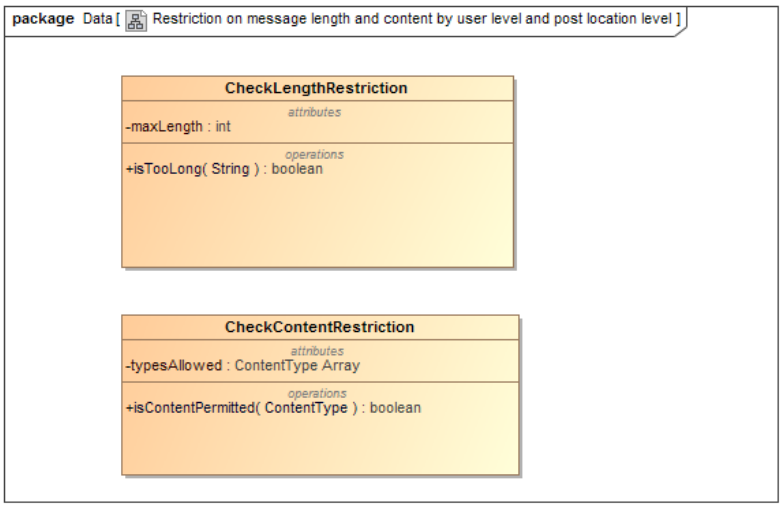


Figure 16: Message Length and Content Restriction activity diagram

4. Restrict users to post on specified levels based on their status.

   (a) Elaboration - User interaction with the current Buzz space must be configurable by policy to allow users with higher levels to post to the Buzz space on higher levels such as directly below the main post while restricting low level users to only post in lower levels like sub level posts or even sub sub level posts. This allows high level users to post higher up in the Buzz spaces hierarchy while restricting low level users to the bottom.

   (b) Importance - ****

   (c) Dependency level - Relies on the ranking system to be implemented so that it can request user levels. A policy to govern the levels has to be supplied by the creator of the Buzz space.

   (d) Pre-conditions

      i. User must be part of the specific Buzz space.

      ii. Policy acquired.

      iii. User must try to post.

   (e) Post-conditions

      i. User successfully posted to the correct level as specified in the policy.

   (f) Requester - System (This is an automated system requirement)

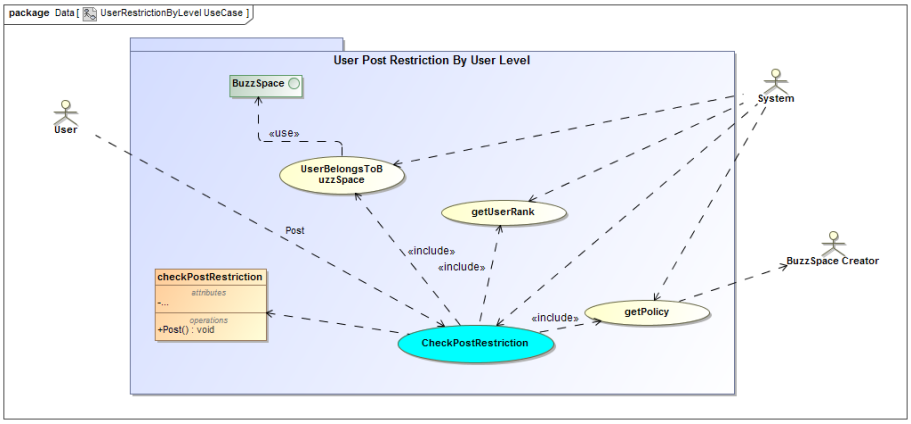


Figure 17: User Post Restriction By User Level Use Case

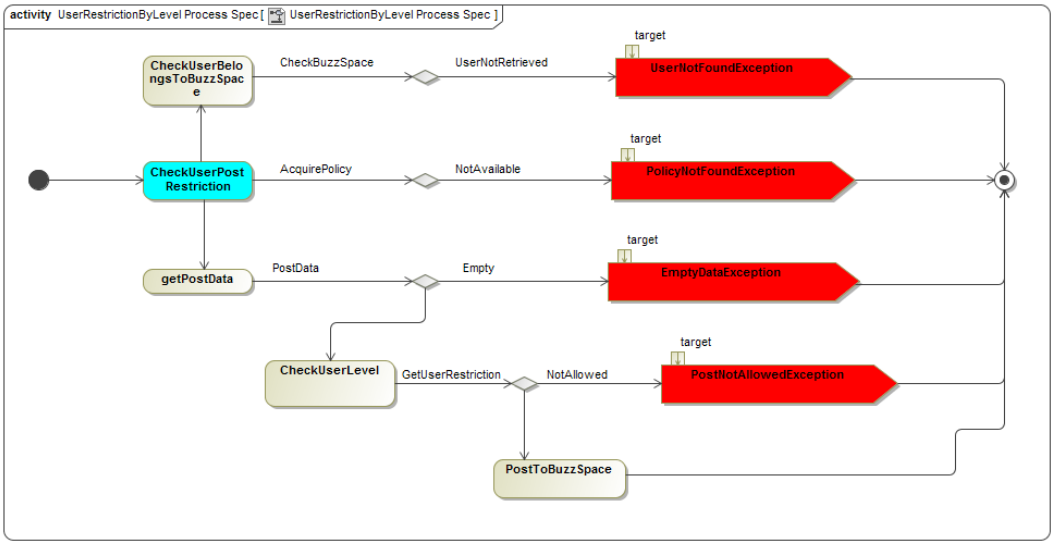


Figure 18: User Post Restriction By User Level Process Specification

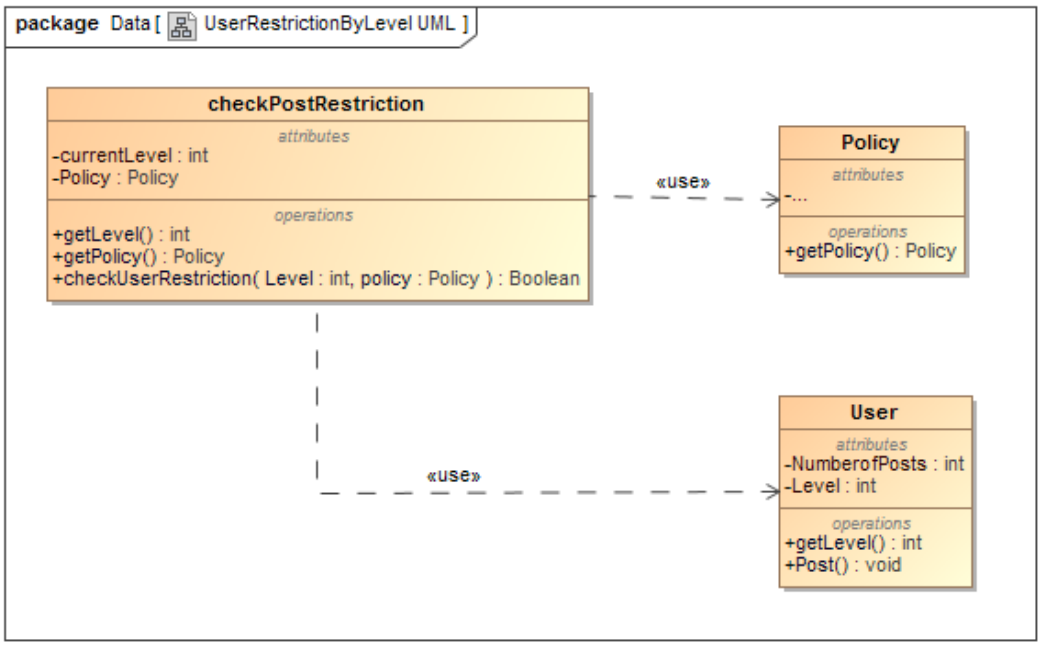


Figure 19: User Post Restriction By User Level UML

5. Allow staff to manage content i.e. summarise, close or hide threads and move things around.

   (a) Elaboration - After a staff member has viewed through the thread he can choose to add a summary of what the thread is about. He can then close the thread to stop further posts but still visible to other members or hide the thread to make it only visible to selected members.

   (b) Importance - ****.

   (c) Dependency level - Relies on the user having the rights or high enough rank/owner of the thread.

   (d) Pre-conditions

      i. The user has a high enough rank.

      ii. The user is the owner of the thread.

   (e) Post-conditions

      i. N/A.

      ii. N/A.

   (f) Requester

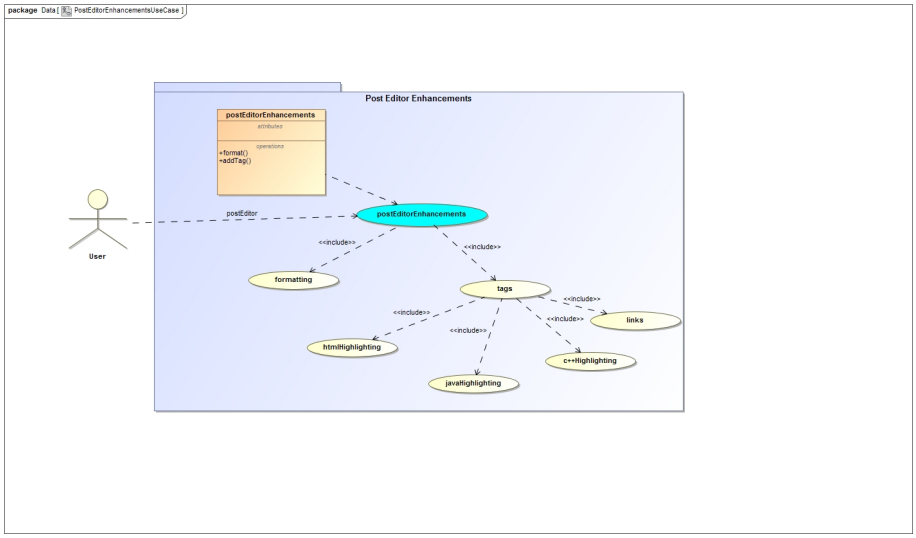


Figure 20: Staff Management of Content - Use Case

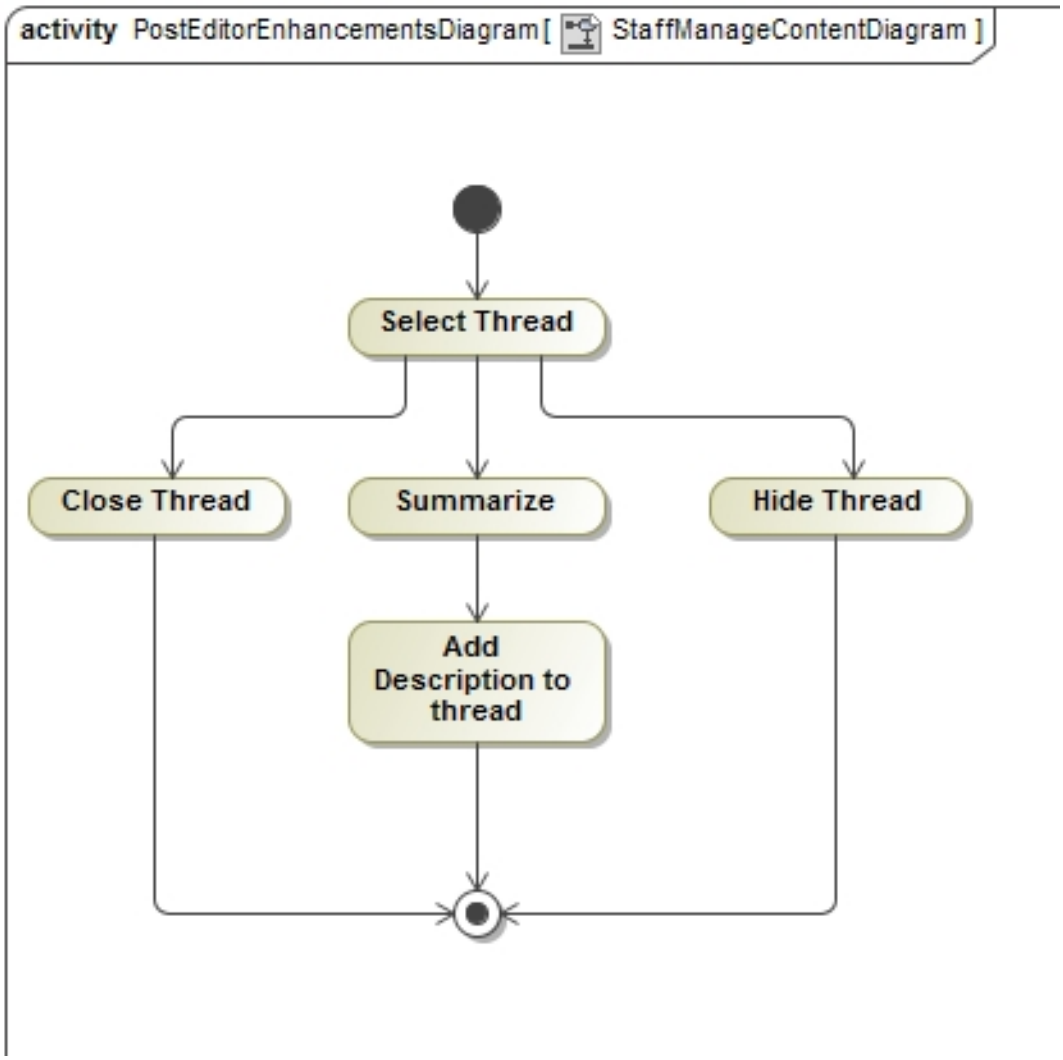


Figure 21: Staff Management of Content - Process Specification

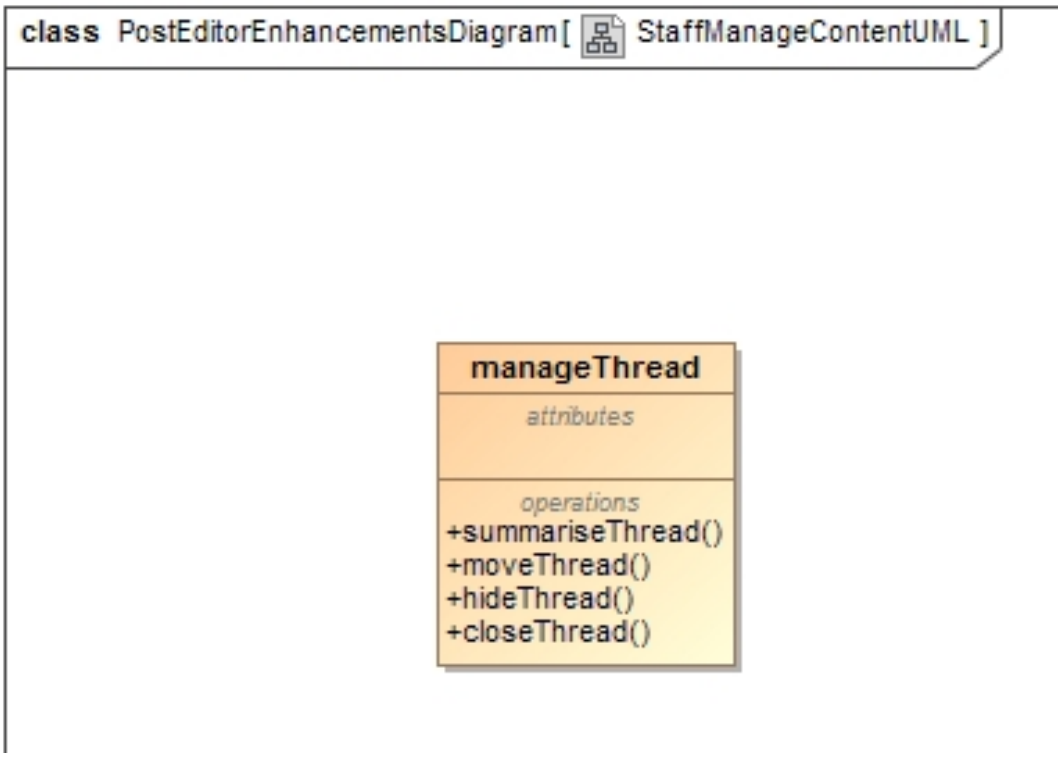


Figure 22: Staff Management of Content - UML

6. Provide functionality to support semi-automatic creation of thread summaries

   (a) Elaboration After an allocated user has gone through a thread, selecting the posts that are important enough to be added into the thread summary, the System must process the thread and create the summary using the relevant posts.

   (b) Importance - ***

   (c) Dependency level Relies on the ability to give staff/privileged users the ability to edit threads. It also needs the ranking system as well as a policy governing user levels to be available to it.

   (d) Pre-conditions

      i. User must be given the privileges required to view the thread at a higher level and select/unselect posts for inclusion/exclusion in the summary

      ii. Allocated user must submit thread for summarizing to be done by the System

   (e) Post-conditions

      i. User successfully selected the posts that need to be included in the summary and submitted the thread for summarizing

      ii. System processed the thread posts and created one summary of the thread

   (f) Requester Allocated user

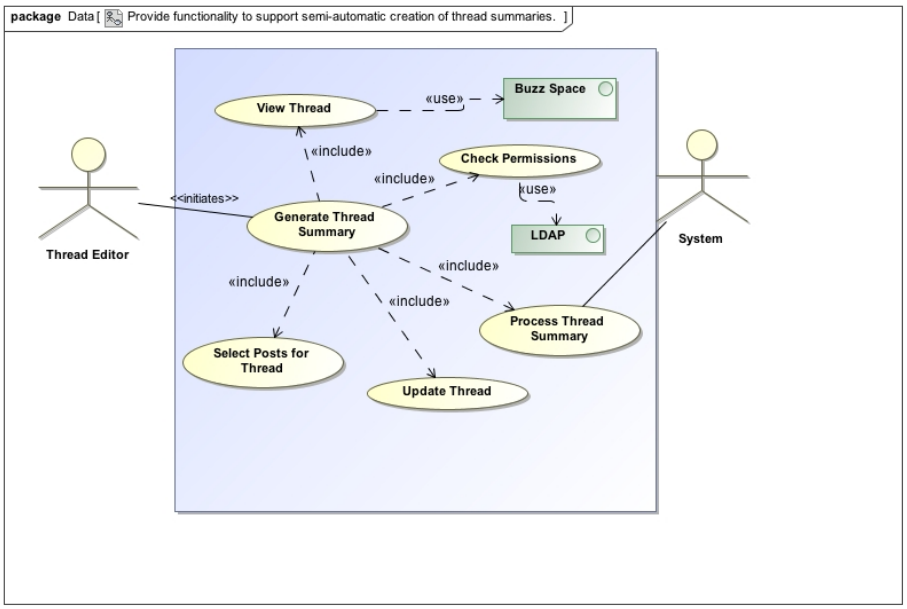


Figure 23: Semi-automatic creation of thread summaries Use Case

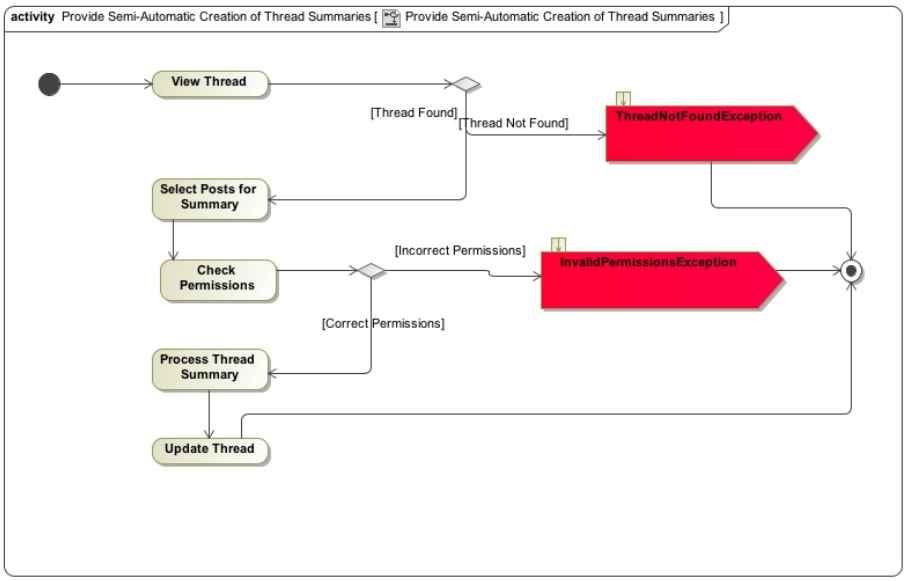


Figure 24: Semi-automatic creation of thread summaries Activity Diagram

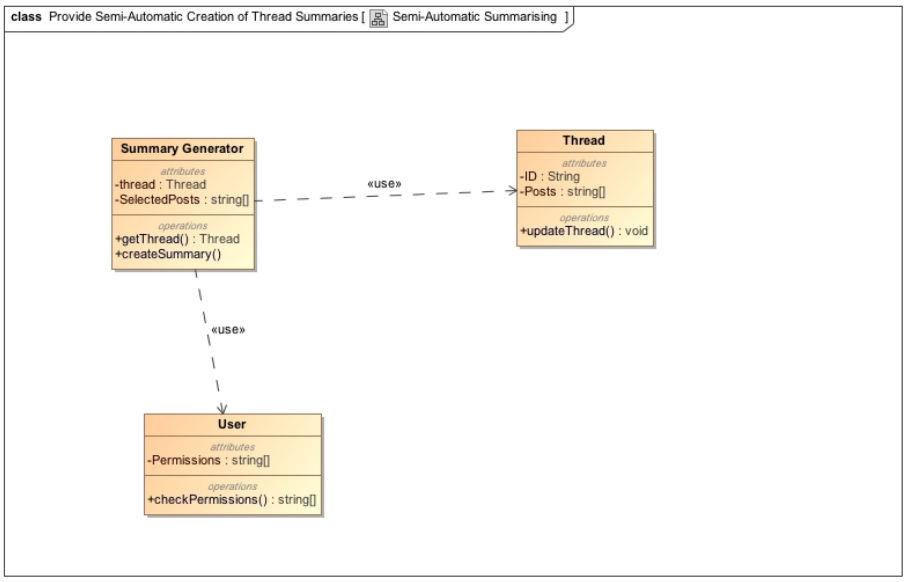


Figure 25: Semi-automatic creation of thread summaries Class Diagram

7. Create automated template based messages to individual users or specified groups

   (a) Elaboration - This generates a template message about a topic to an individual user or a group.

   (b) Importance - ***

   (c) Dependency level - This depends on the user input; the system will parse through the user input and build a summary accordingly

   (d) Pre-conditions

      i. Condition 1 - Users have to be connected to buzz either logged in or guest account to get automated messages.

      ii. Condition 2 - Group must exist and have at least one member.

   (e) Post-conditions

      i. Condition 1 - Individual user gets a system generated message.

      ii. Condition 2 - Group members get a system generated message.

   (f) Requester - The system.

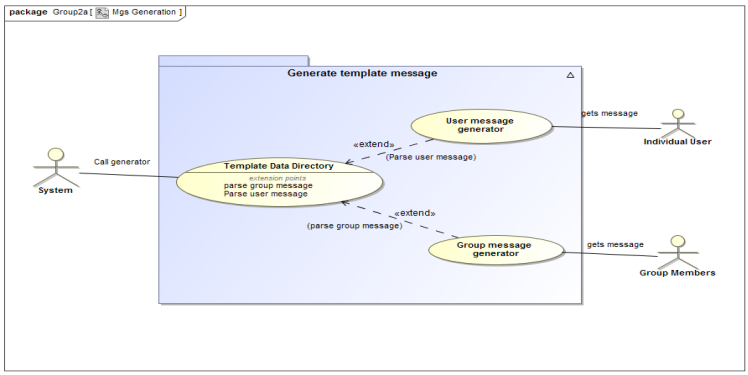


Figure 26: Generate template message for user or group.

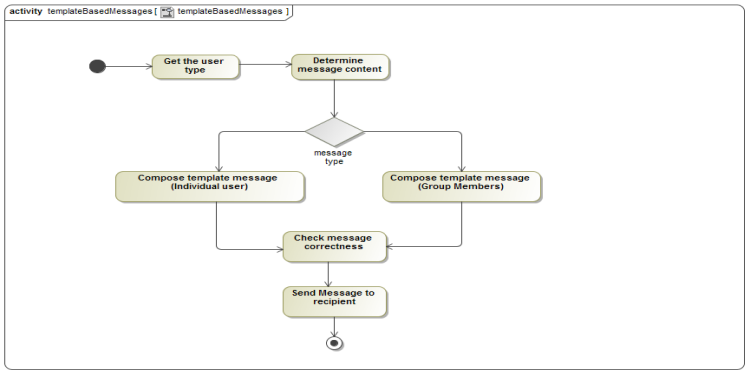


Figure 27: Process specification for generate template message for user or group.

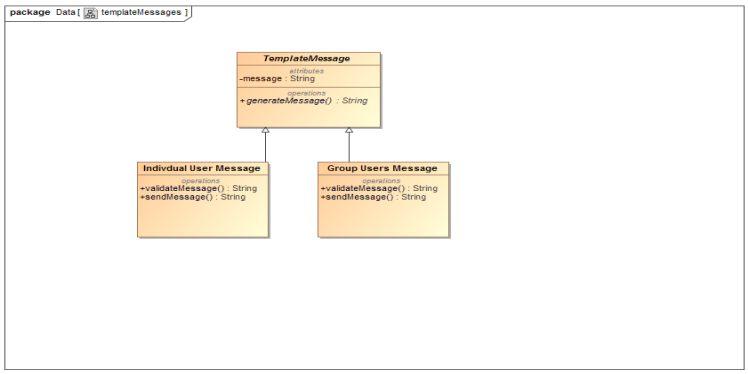


Figure 28: Class diagram for generate template message for user or group.

8. Automatically change the status of a user based on the users participation

   (a) Elaboration - The System should be configured in such a way that when a user participates often the user will progress through the levels of the system, Specific number of points required per level.

   (b) Importance - ****

   (c) Dependency level - Requires the users level to be implemented before a user can participate and increase in level, Users should be able to post on Buzz

   (d) Pre-conditions

      i. User is in level x

      ii. User only needs y amount of points to progress

   (e) Post-conditions

      i. User achieved y amount of point

      ii. User is now in level x+1

   (f) Requester - System (Automatically checks each time a user posts)

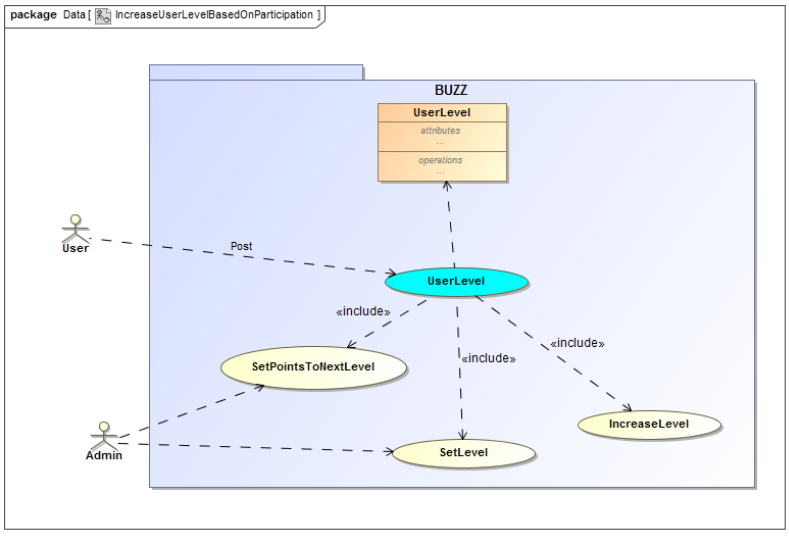


Figure 29: Automatically Change users level based on participation Use Case

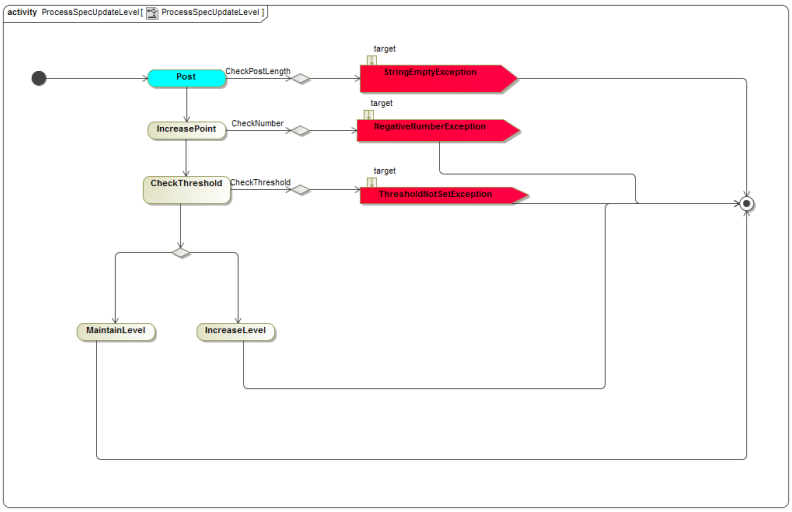


Figure 30: Process Specification for Automatically Update User Level

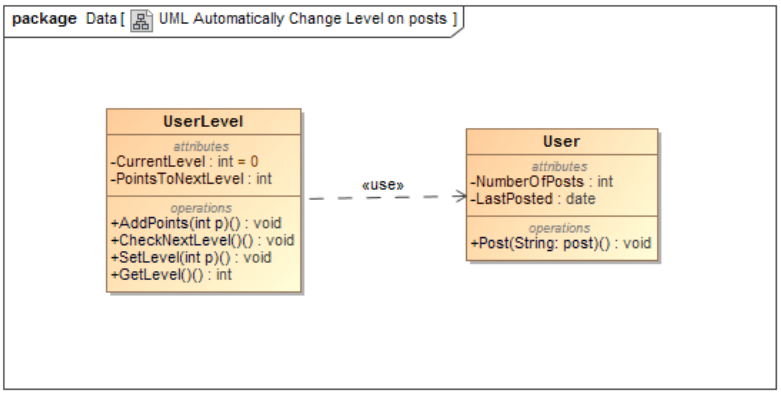


Figure 31: Automatically Change users level based on participation UML

9. Integrate seamlessly with any specified host site.

   (a) Elaboration - This process allows owners of the host site to integrate the Buzz space into their site with minimal changes required to their code.

   (b) Importance - *****, Critical

   (c) Dependency level - The integration of the Buzz Space onto a host site is also important. Host site should not be heavily modified to accommodate the Buzz space, the process must be simple and easy.

   (d) Pre-conditions

      i. Must have host site.

      ii. Must have access to host site.

   (e) Post-conditions

      i. Users will be able to interact with one another on Buzz space through the host site.

   (f) Requester - Client

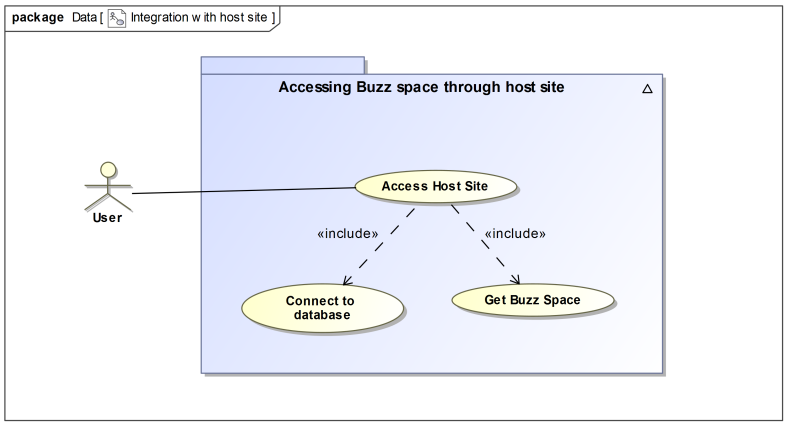


Figure 32: Integration of Buzz Space in host site use case diagram

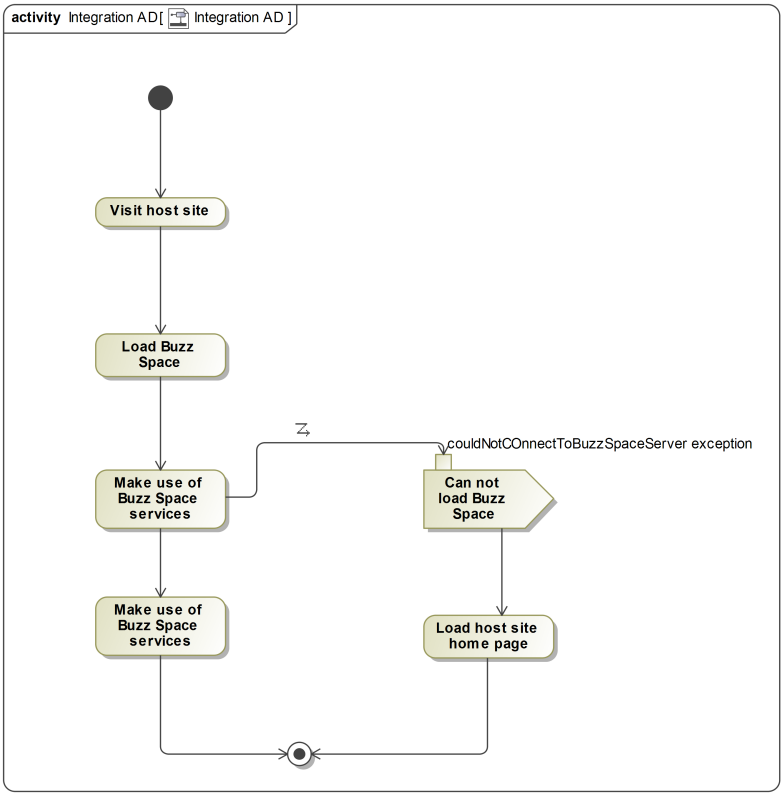


Figure 33: Integration of Buzz Space in host site activity diagram

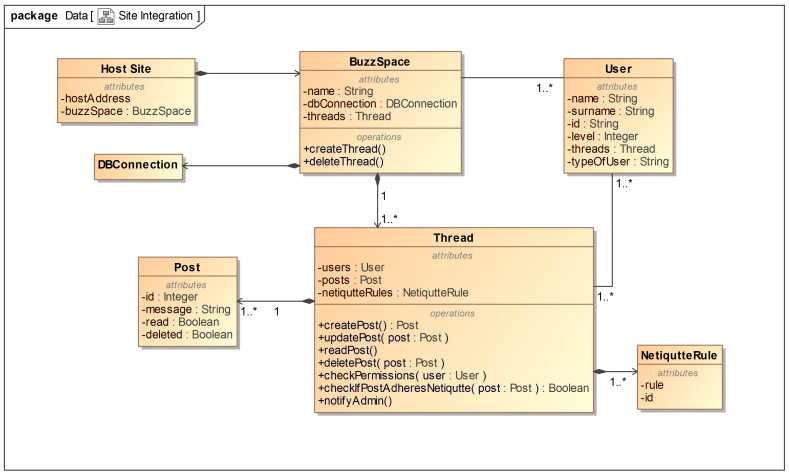


Figure 34: Integration of Buzz Space in host site class diagram

10. Provide functions such as searching and filtering.

    (a) Elaboration - This function will allow users to search for threads by topic, original poster, date etc. and filter posts by the topic such as assignments, practicals, tutorials, general etc.

    (b) Importance - **

    (c) Dependency level - Overall this function does not impact the use of the system and is not critical to the functioning of the system, although it makes navigating through the buzz space somewhat easier.

    (d) Pre-conditions

        i. User enters a search query

        ii. User chooses filter condition

    (e) Post-conditions

        i. User is presented with results of search query

        ii. User is presented with filtered threads

    (f) Requester - Client

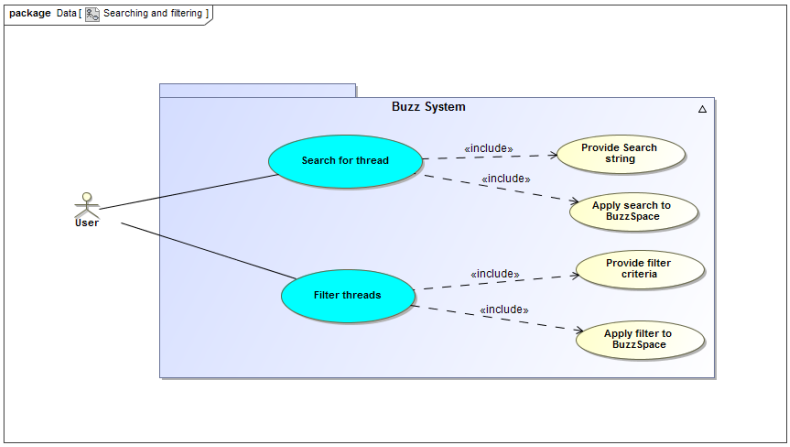


Figure 35: Searching and Filtering Use Case

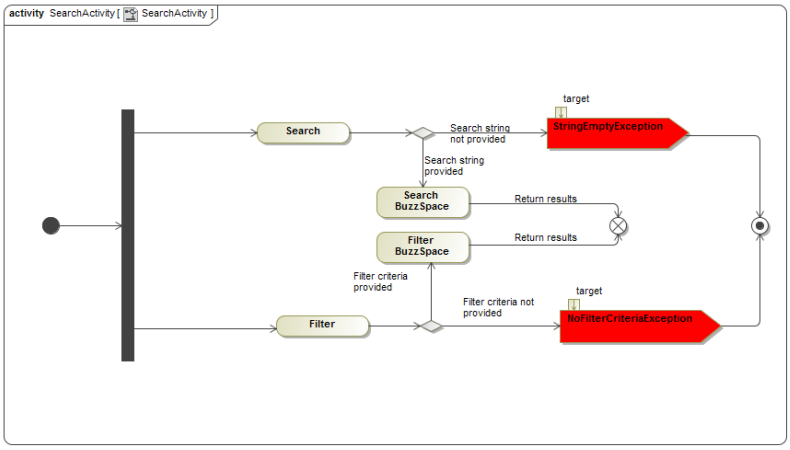


Figure 36: Searching and Filtering Activity Diagram

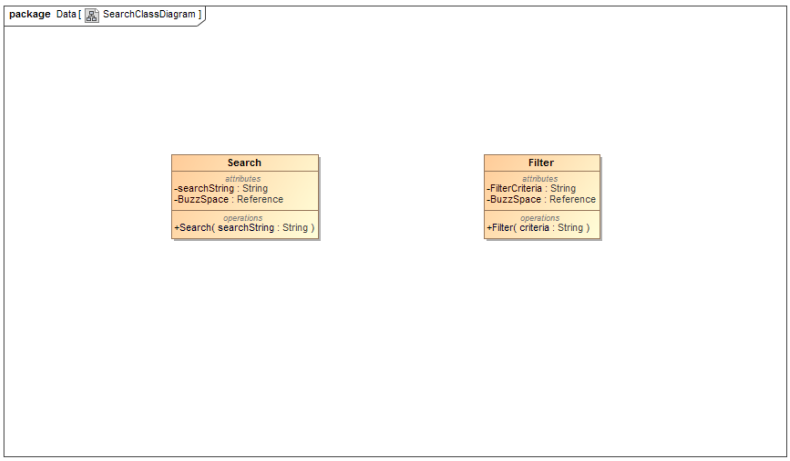


Figure 37: Searching and Filtering Class Diagram

11. Provide functionality to evaluate posts and vote for posts.

    (a) Elaboration - Users must be able to vote a post up or down. Buzz Space creators must be able to evaluate posts. This will result in more relevant posts getting higher ratings and thus standing out.

    (b) Importance - ***

    (c) Dependency level - Depends on the post having been created.

    (d) Pre-conditions

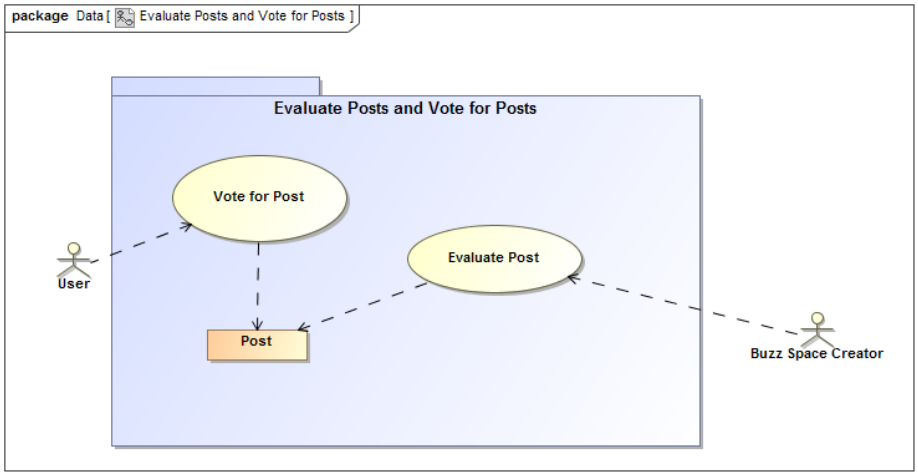
        i. Post must be created

    (e) Post-conditions

        i. Post is voted up or down.

        ii. Post is evaluated.

    (f) Requester - User



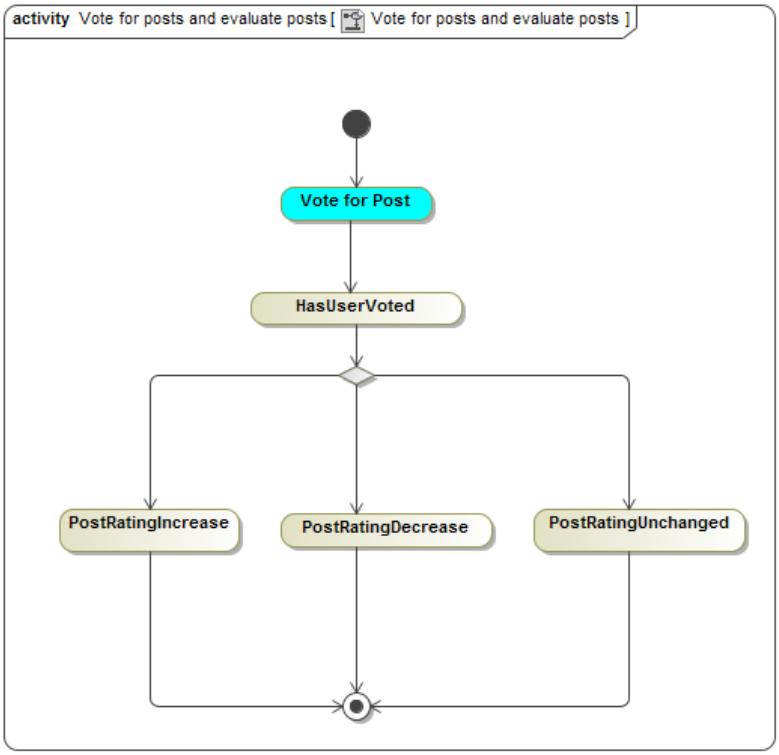


Figure 38: Vote For Posts And Evaluate Posts

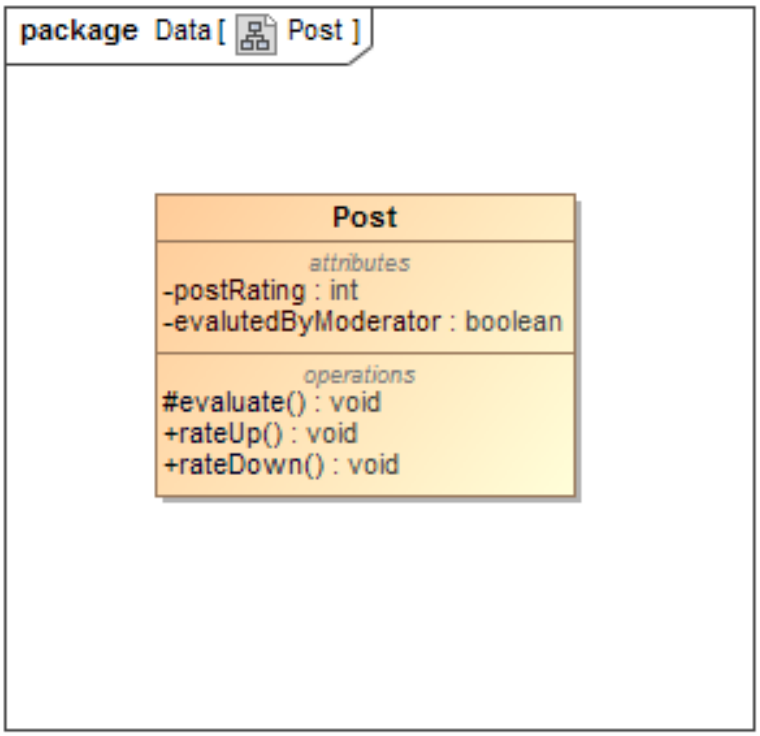


Figure 39: Posts UML

12. Gather statistical information on each user for graphical representation.

    (a) Elaboration - System must continuously gather each users contribution and participation in the Buzz space to allow the user to view a graphical representation of where they stand or rank up amongst their peers. Provide a game like scoreboard to motivate the users.

    (b) Importance - **

    (c) Dependency level - Requires the ranking system to be active in order to pull the stats.

    (d) Pre-conditions

        i. User has participated at least x times to gather historical data

        ii. More than y users have contributed to specific Buzz Space

    (e) Post-conditions

        i. User receives the graphical representation of their participation.

    (f) Requester - User

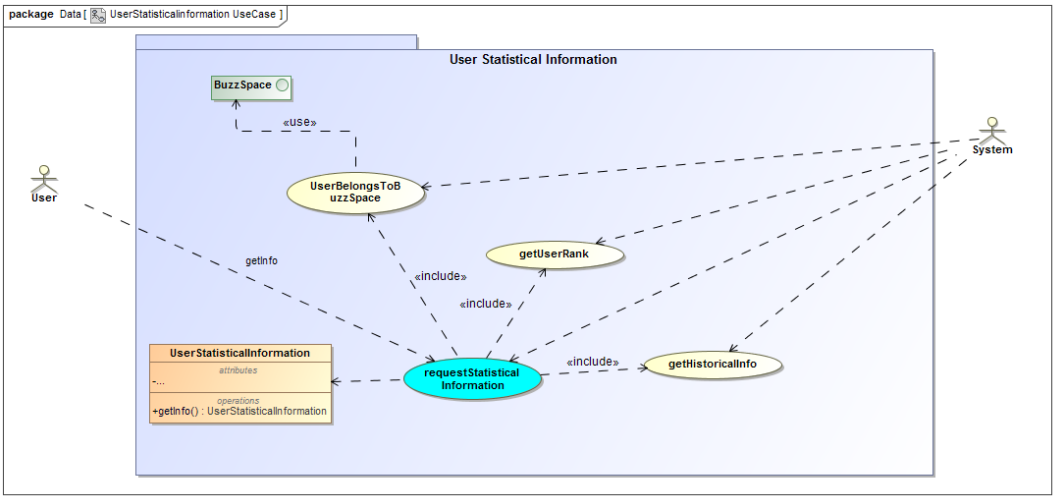


Figure 40: User Statistical Information Use Case

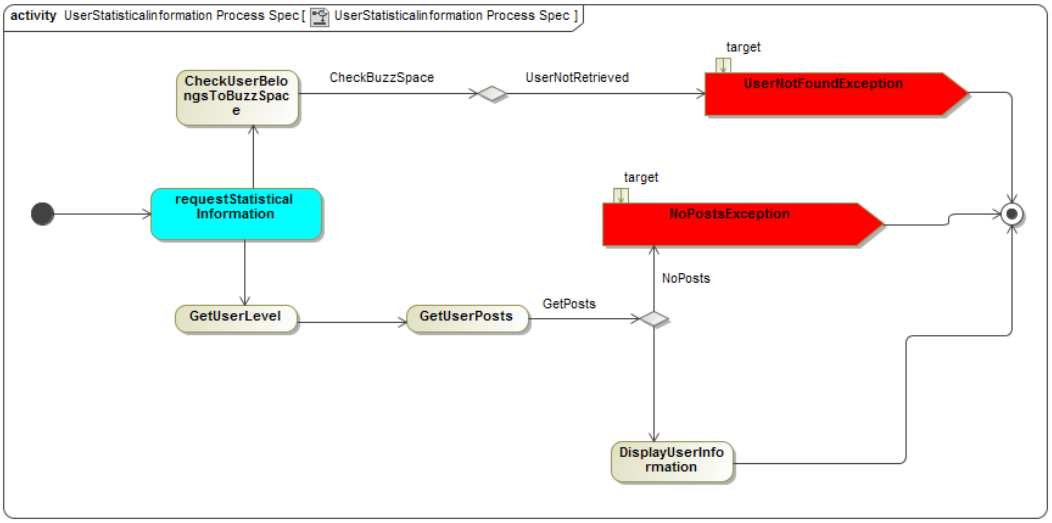


Figure 41: User Statistical Information Process Specification

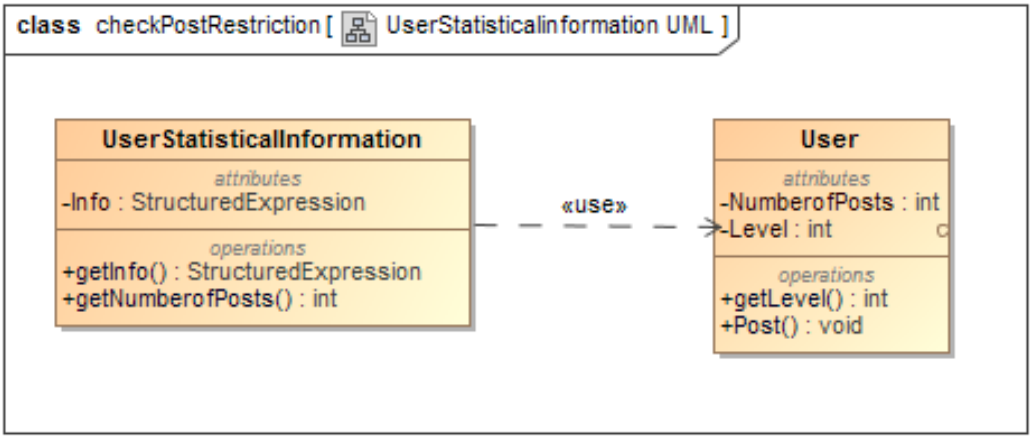


Figure 42: User Statistical Information UML

13. Enhancement of the post editor

    (a) Elaboration - When a user is posting a reply or creating a new thread there will be options to change formatting or styling of the text as well as embed code within tags to add highlighting and increase readability.

    (b) Importance - ***

    (c) Dependency level - Relies on a user being allowed to make a post on a thread or create a new thread.

    (d) Pre-conditions

        i. User allowed to post reply to thread

        ii. User allowed to create new thread

    (e) Post-conditions

        i. Links posted are allowed

        ii. N/A

    (f) Requester

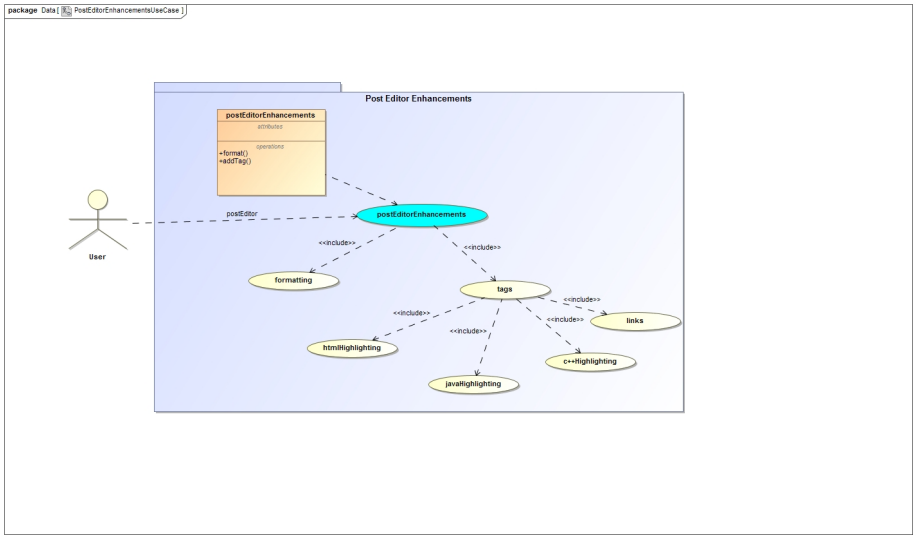


Figure 43: Post Editor for enhanced posts - Use Case

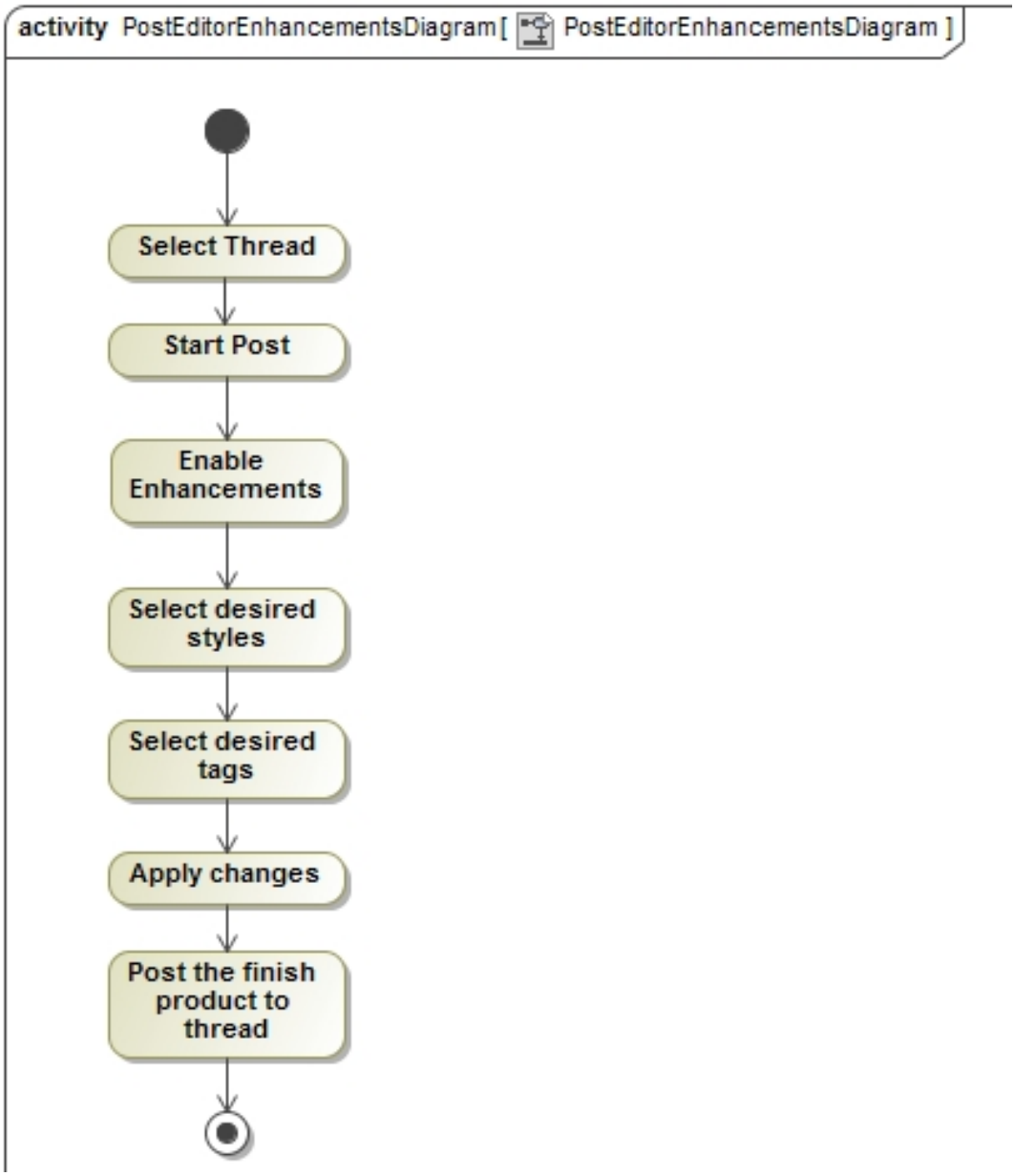


Figure 44: Post Editor for enhanced posts - Process Specification

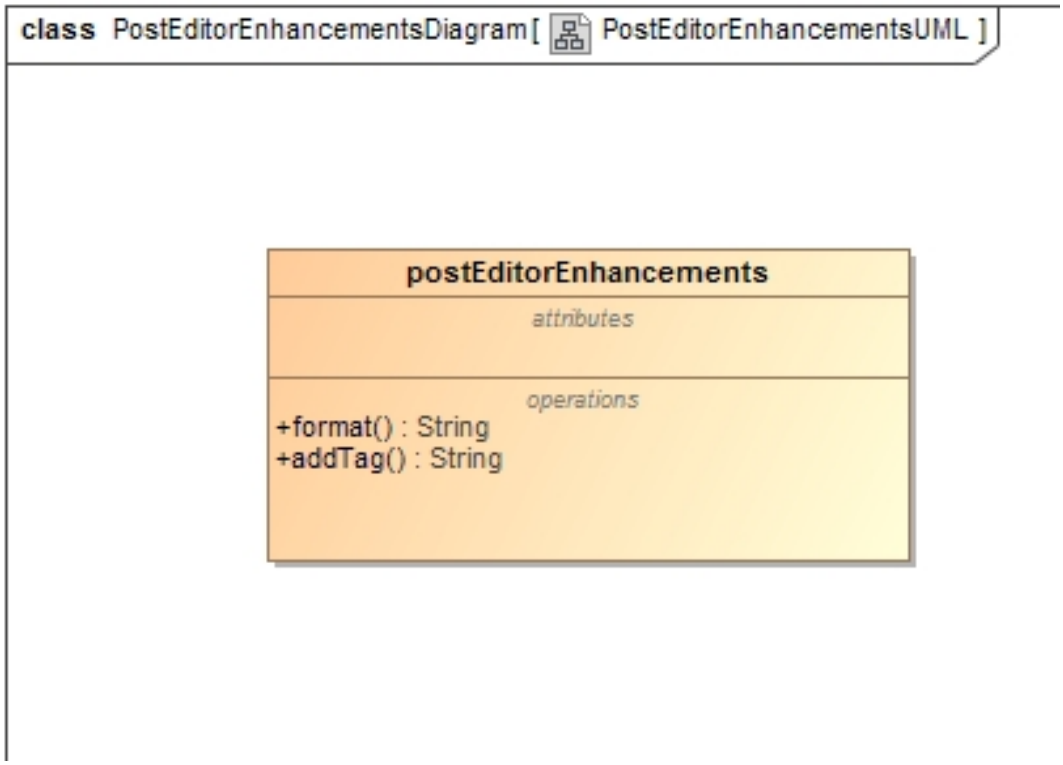


Figure 45: Post Editor for enhanced posts - UML

14. Provide functions to apply social tagging.

    (a) Elaboration A user should be able to add a set of tags to his/her own posts as well as to threads/other posts if their user level allows which will be used for more efficient searching of relevant posts.

    (b) Importance - **

    (c) Dependency level Relies on the ranking system being implemented as well as the policy governing user levels so that users with correct privileges can add tags to posts and/or threads.

    (d) Pre-conditions

        i. User has correct privileges to add tags to the specific thread/post

        ii. User wants to add tags to thread/post

    (e) Post-conditions

        i. User has entered tag set and tag set has been set for the thread/post.

    (f) Requester User

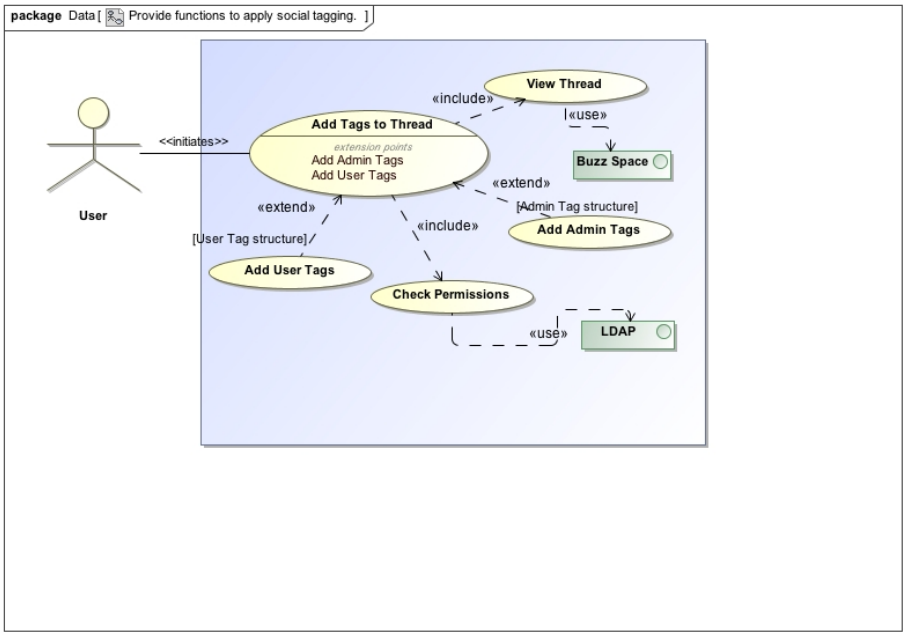


Figure 46: Apply Social Tagging Use Case

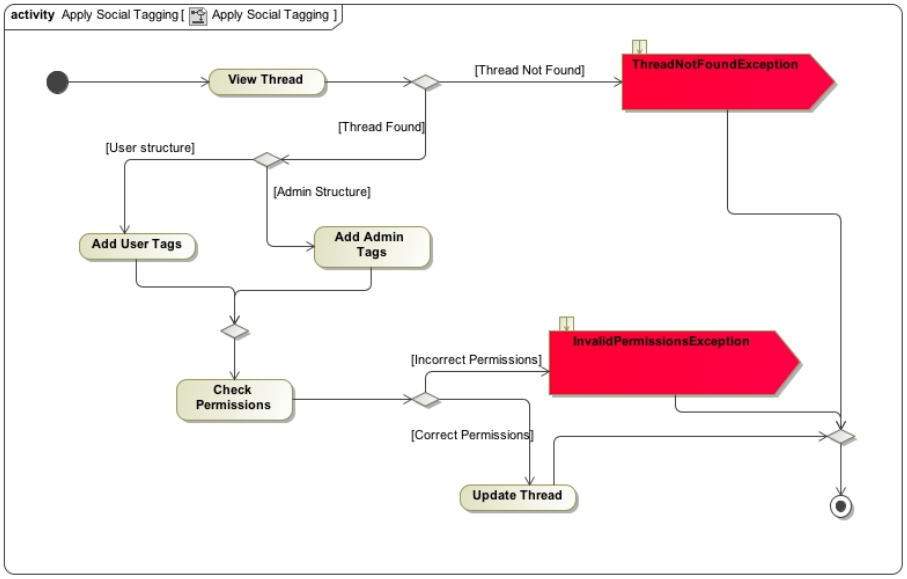


Figure 47: Apply Social Tagging Activity Diagram

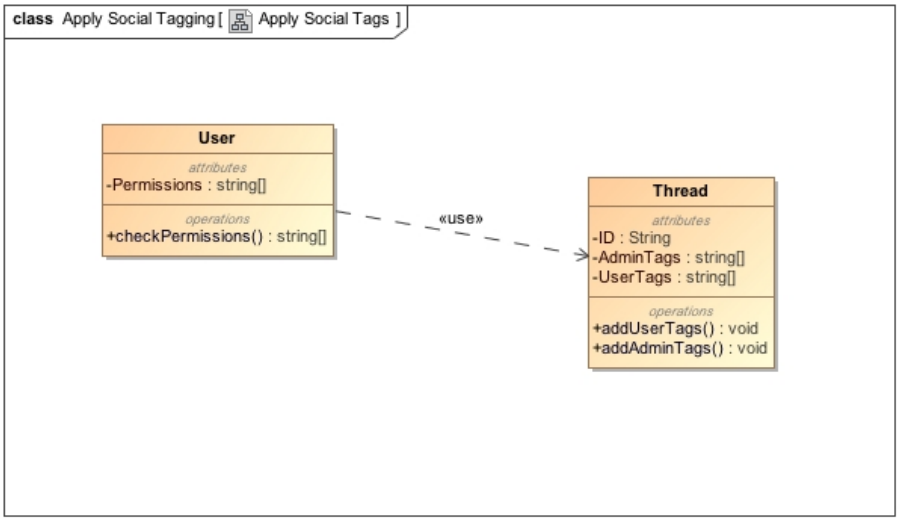


Figure 48: Apply Social Tagging Class Diagram

15. Apply self-organasation based on social tagging and allow the user to view according to the base structure, own structure or public structure

    (a) Elaboration - The user is able to use social tags that tell us more about a post thus is also able to search for a thread according to topics that have those social tags and arrange topic threads accordingly via tags. The user is also able to view posts according to the base structure given by the buzz system or specify their own structure by making use of the social tags there is also a general structure for the public who are not registered users.

    (b) Importance - ***

    (c) Dependency level - This feature depends on the user selecting tags in which to order the base structure of the posts that they see.

    (d) Pre-conditions

        i. Condition - Base structure of posts that is unsorted according to social tags.

    (e) Post-conditions

        i. Condition - Structure that is sorted according to the user's selected organisation of social tags.

    (f) Requester - The user.

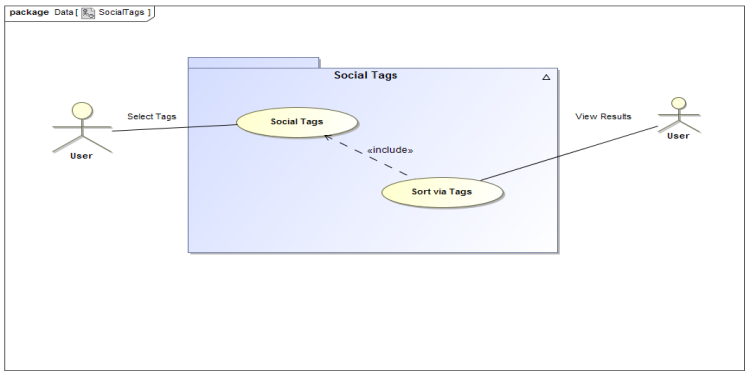


Figure 49: Self-organasation of data via social tags.

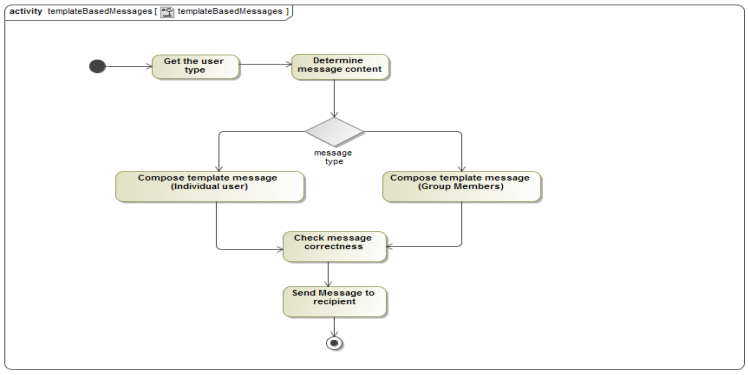


Figure 50: Process specification for self-organasation of data via social tags.

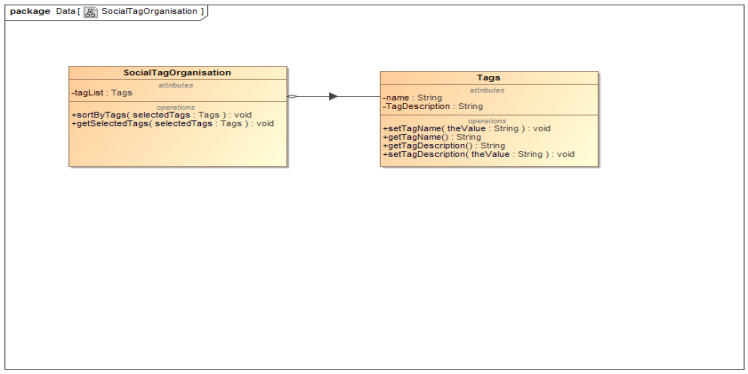


Figure 51: Class diagram for self-organasation of data via social tags.

16. Detect if a post is plagiarised

    (a) Elaboration - The entire post will be checked to see if it has been copied directly from another source, a full post quote will be open searched in a search engine, if any hits are found the post will be marked as possibly plagiarised and send to administrator

    (b) Importance - **

    (c) Dependency level - User must be able to post to Buzz

    (d) Pre-conditions

        i. User posts a post

    (e) Post-conditions

        i. Post is marked as Plagiarised - Added to Buzz(Invisible, Message sent to user and Administrator

        ii. Post is marked as not Plagiarised - Posted to Buzz

    (f) Requester - System, Automatically checks to see if the post is plagiarised.

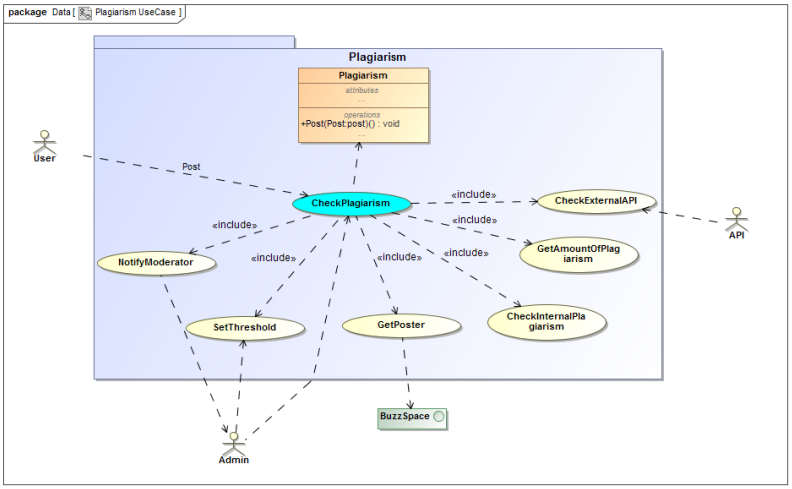


Figure 52: Plagiarism Check Use Case

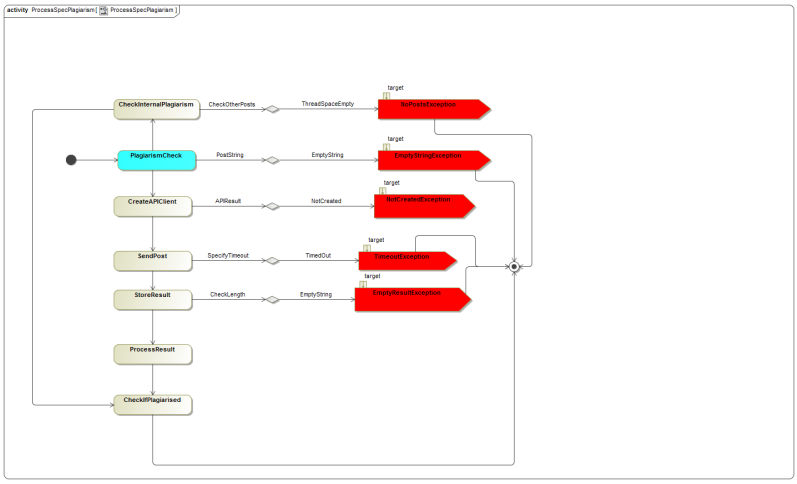


Figure 53: Process Specification for Checking Plagiarism API and Internal Checks

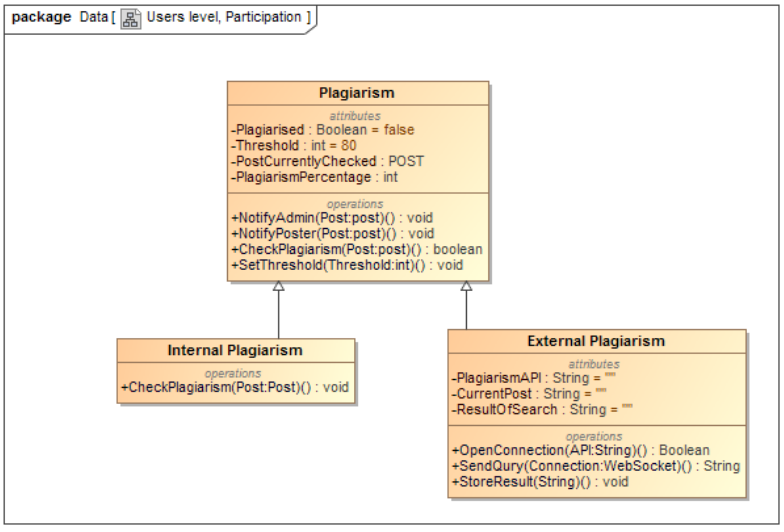


Figure 54: Plagiarism Check UML

17. Detect violation of netiquette rules.

    (a) Elaboration - Many of the users are new to online interaction and often do not know/follow the netiquette rules. This process will check their post against a set of netiquette rules, if the post does not follow these rules the moderator will be notified via e-mail.

    (b) Importance - ***, Nice to have

    (c) Dependency level - This function is not very important to the system, it is a nice feature to have. The Buzz space can operate without this process.

    (d) Pre-conditions

        i. User must create a post.

    (e) Post-conditions

        i. Users post will be created successfully if it follows the netiquette rules.

        ii. If the post does not follow one of the netiquette rules, the moderator will be notified about this and post will be flagged.

    (f) Requester - System

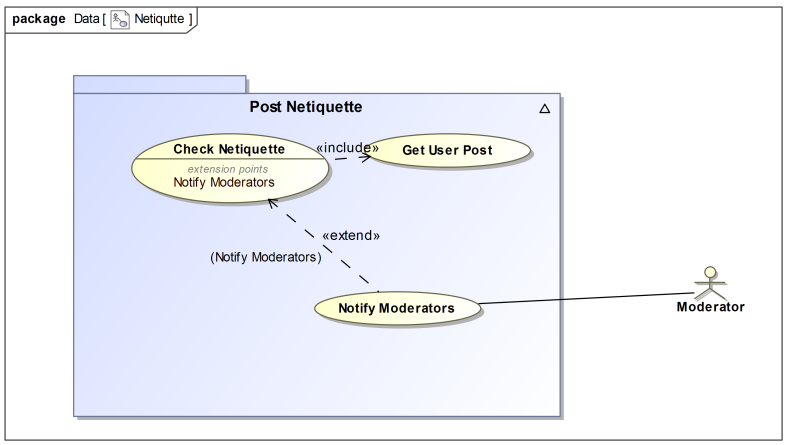


Figure 55: Check if post violates netiquette rules use case

...

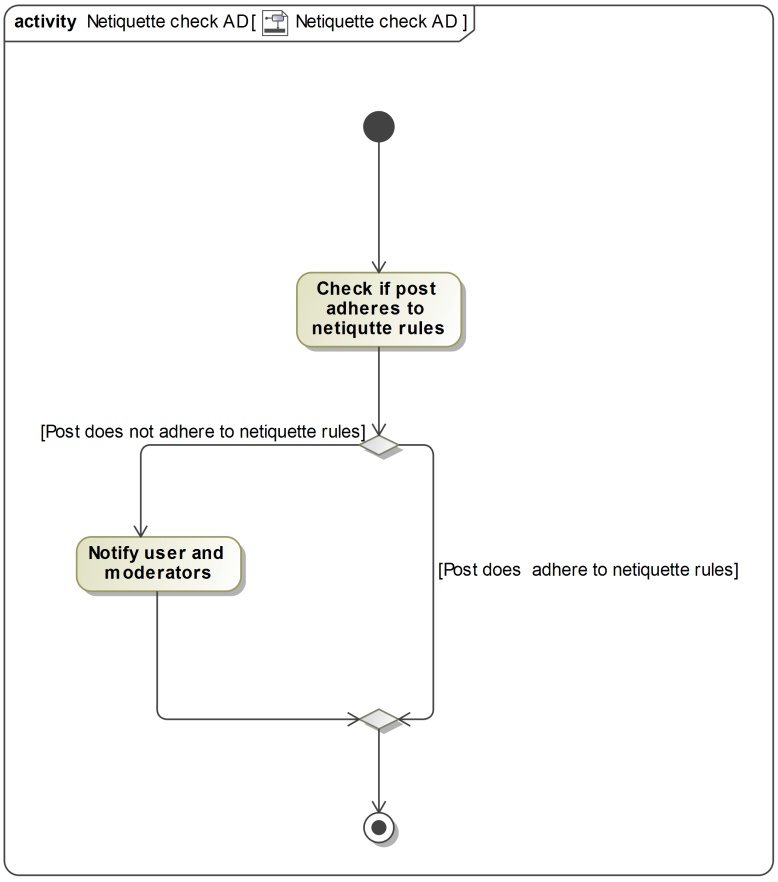


Figure 56: Check if post violates netiquette rules activity diagram

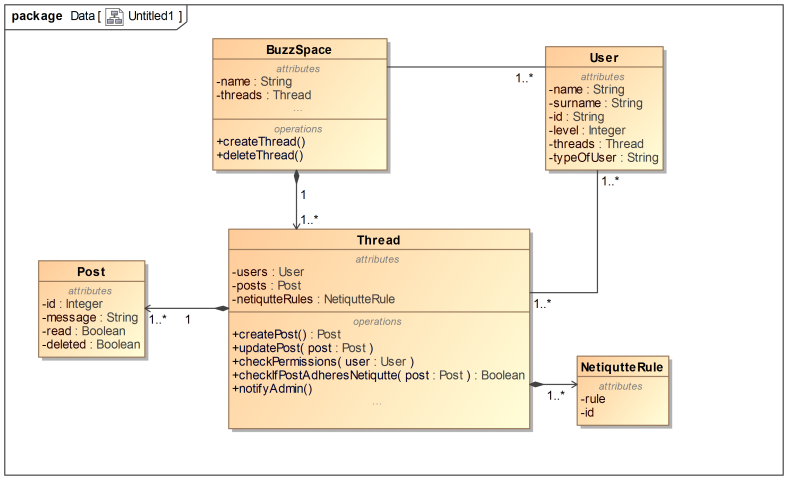


Figure 57: Check if post violates netiquette rules class diagram