UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

Department of Computer Science

COS 301 - Software Engineering

COS 301 - Mini Project

Author:	*Student number:*
Hanrich Potgieter	u12287343
Chris Cloete	u13029721
Jason Richard Evans	u13032608
Kale-ab Tessera	u13048423
Lelethu Zazaza	u13028023
Goodness Adegbenro	u13046412
Herman Willem Keuris	u13037618
William Seloma	u10155865

February 27, 2015

DECLARATION OF ORIGINALITY

UNIVERSITY OF PRETORIA

The University of Pretoria places great emphasis upon integrity and ethical conduct in the preparation of all written work submitted for academic evaluation.

While academic staff teach you about referencing techniques and how to avoid plagiarism, you too have a responsibility in this regard. If you are at any stage uncertain as to what is required, you should speak to your lecturer before any written work is submitted.

You are guilty of plagiarism if you copy something from another author's work (e.g. a book, an article or a website) without acknowledging the source and pass it off as your own. In effect you are stealing something that belongs to someone else. This is not only the case when you copy work word-for-word (verbatim), but also when you submit someone else's work in a slightly altered form (paraphrase) or use a line of argument without acknowledging it. You are not allowed to use work previously produced by another student. You are also not allowed to let anybody copy your work with the intention of passing if off as his/her work.

Students who commit plagiarism will not be given any credit for plagiarised work. The matter may also be referred to the Disciplinary Committee (Students) for a ruling. Plagiarism is regarded as a serious contravention of the University's rules and can lead to expulsion from the University.

The declaration which follows must accompany all written work submitted while you are a student of the University of Pretoria. No written work will be accepted unless the declaration has been completed and attached.

Full names of students: ______________________________

Student numbers: ______________________________

Topic of work: ______________________________

Declaration

1. I understand what plagiarism is and am aware of the University's policy in this regard.
2. I declare that this assignment report is my own original work. Where other people's work has been used (either from a printed source, Internet or any other source), this has been properly acknowledged and referenced in accordance with departmental requirements.
3. I have not used work previously produced by another student or any other person to hand in as my own.
4. I have not allowed, and will not allow, anyone to copy my work with the intention of passing it off as his or her own work.

SIGNATURES: ______________________________ DATE: ____________________

Software Requirements Specification and Technology Neutral Process Design

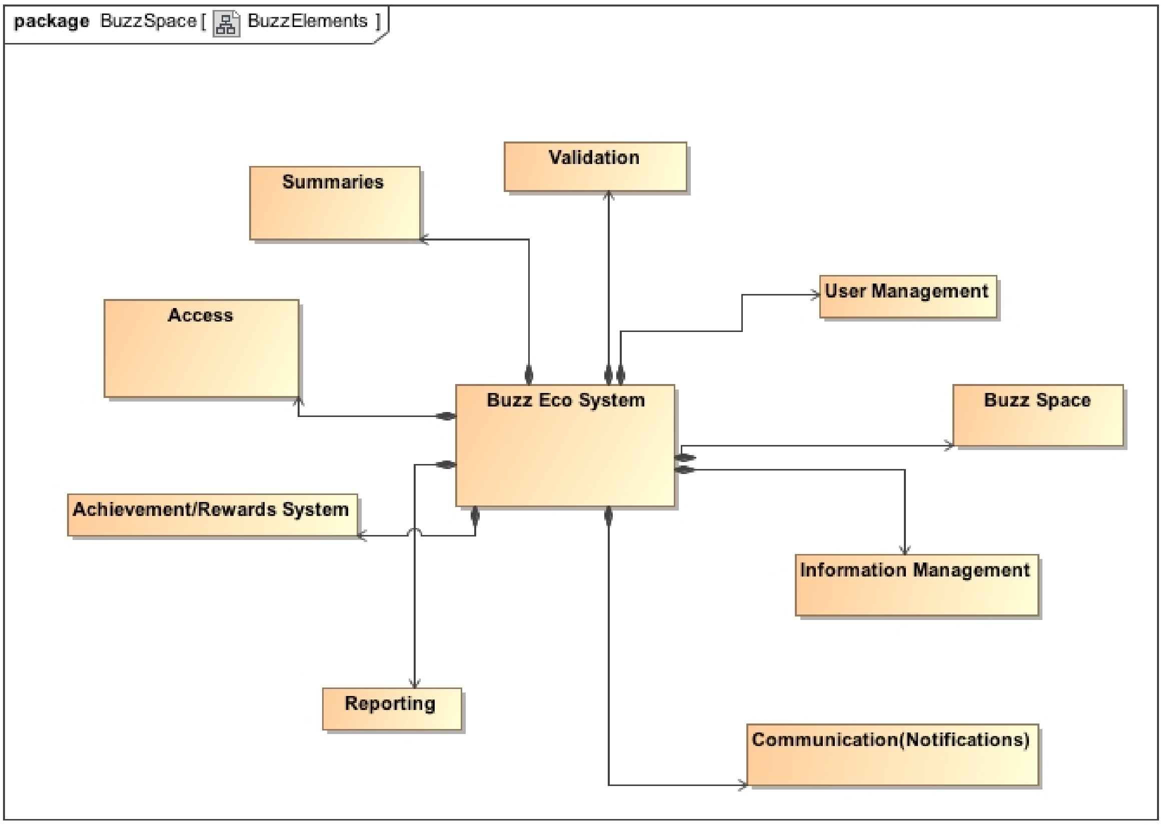
Buzz Space Discussions/Mini Project

Version: Version 0.2 Alpha For further references see gitHub. February 27, 2015

Contents

For further references see gitHub or got to the link https://github.com/DieBaber/COS301-GROUP6-A.git

1 Functional requirements



1.1 Introduction

We use this document to give a high level overview of the buzz discussion board. We have identified the various components of our system. The purpose of this document is to create a dynamic and scalable solution. We also want to include an achievement system that rewards users for using the discussion board. This document will inform you on how we will achieve a system that is both scalable and pluggable. We have identified the use cases of the various components of the discussion board and helped expand on them.

1.2 Use case prioritiation

Critical

- BuzzSpace
- CRUD posts(Creating,Reading; Updating; Deleting).
- System Access
- Information Management

Important

- User Management
- Communication(Notifications)
- Reporting

Nice-To-Have

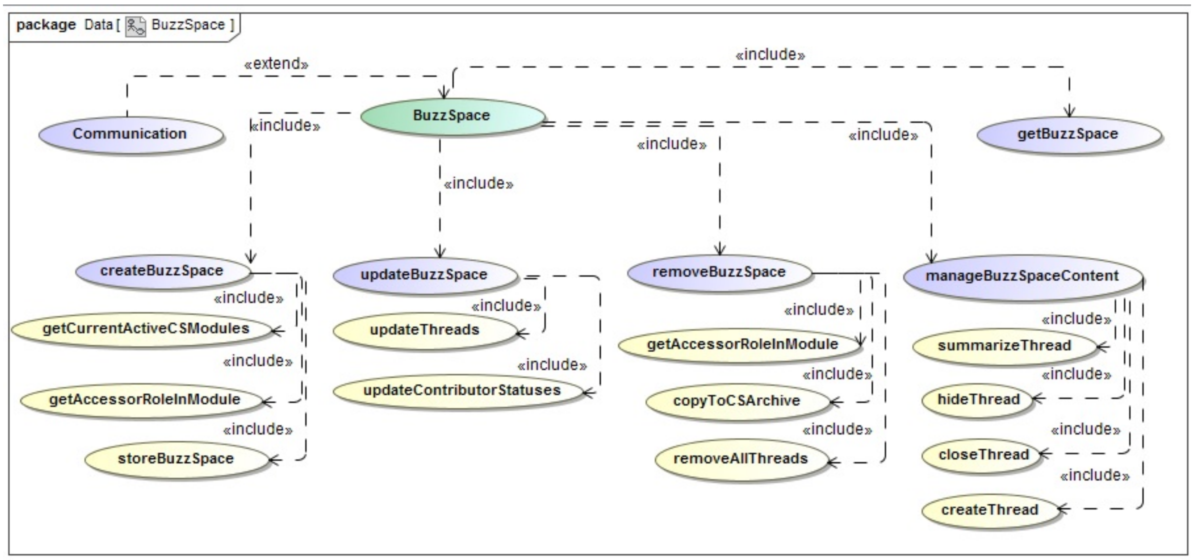
- Achievement/Rewards System
- Reporting
- Summaries

1.3 Use case/Service contracts

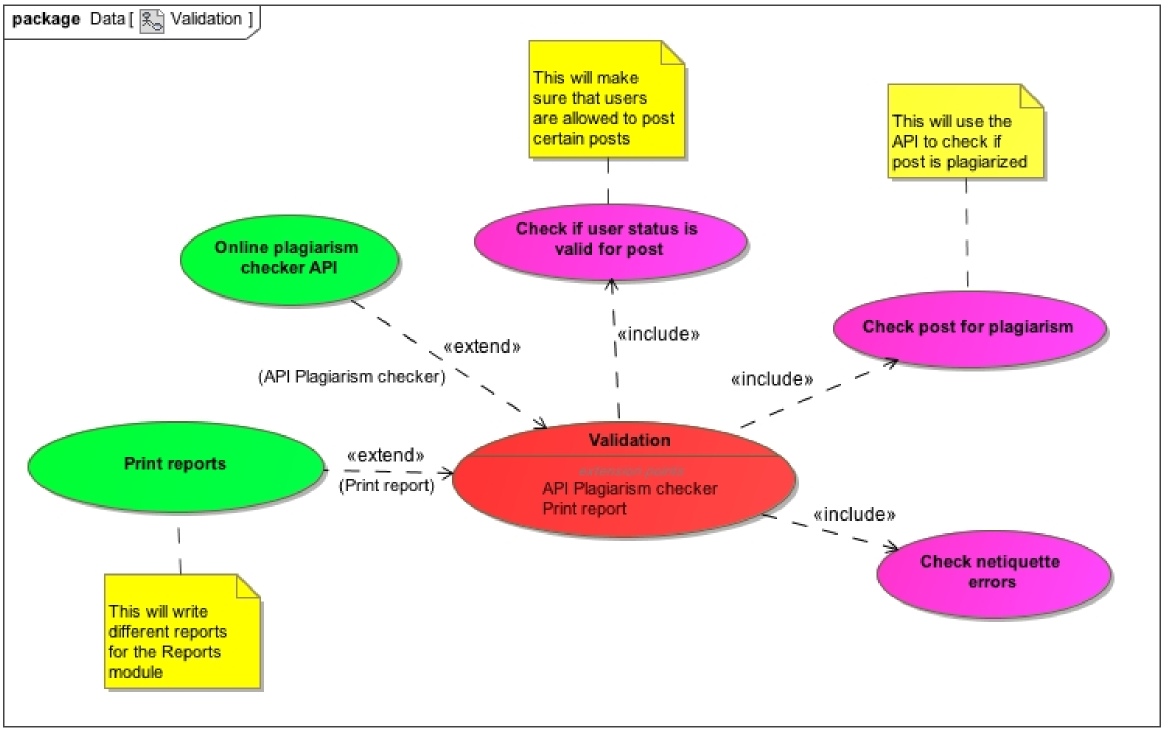
Use Case	Pre Condition	Post Condition	Description
BuzzSpace	There must be a valid user	User must still exist	This use case provides an interface that facilitates management of threads
Information Management			
Communication	A user needs to be registered in order to have notifications sent to his profile inside the application. For e-mails to be sent, a valid and up-to-date e-mail address is needed on the user database.	A notification should visibaly be highlighted in the application with appropriate messages. In some cases, an email is sent out from the system.	This use case specifies all the functions that the Buzz system needs to have in order to communicate important information with the user.
Summaries			
Achievement Rewards System	A user's level requires Achievements to be allocated and/or rewards to be awarded	Achievements are allocated and/or rewards are awarded	This use case provides a system that allocates achievements to users based on their levels and the votes they aquired. it also provides a system that awards rewards to users based on their achievements.
Access	The user will need a browser to view the website.	Threads and posts are displayed in descending order by date.	Details how an end-user will be able to access the Buzz system.
Validation	Post is palgiarised and/or does not follow netiquette	Post is valid against rules	This module will be the interface that other modules use in order to validate any post and help generate certaain reports.
User Management	If the User is not inlvolved in the course (not a registered student/ tutor/ teaching assistant).	The User is still involved in the course	This is a basic system which manages the User's Login and logout.
Reporting	Data must be available to report on.	Data must not be corrupt.	This use case generate report for all actors

1.4 Required functionality

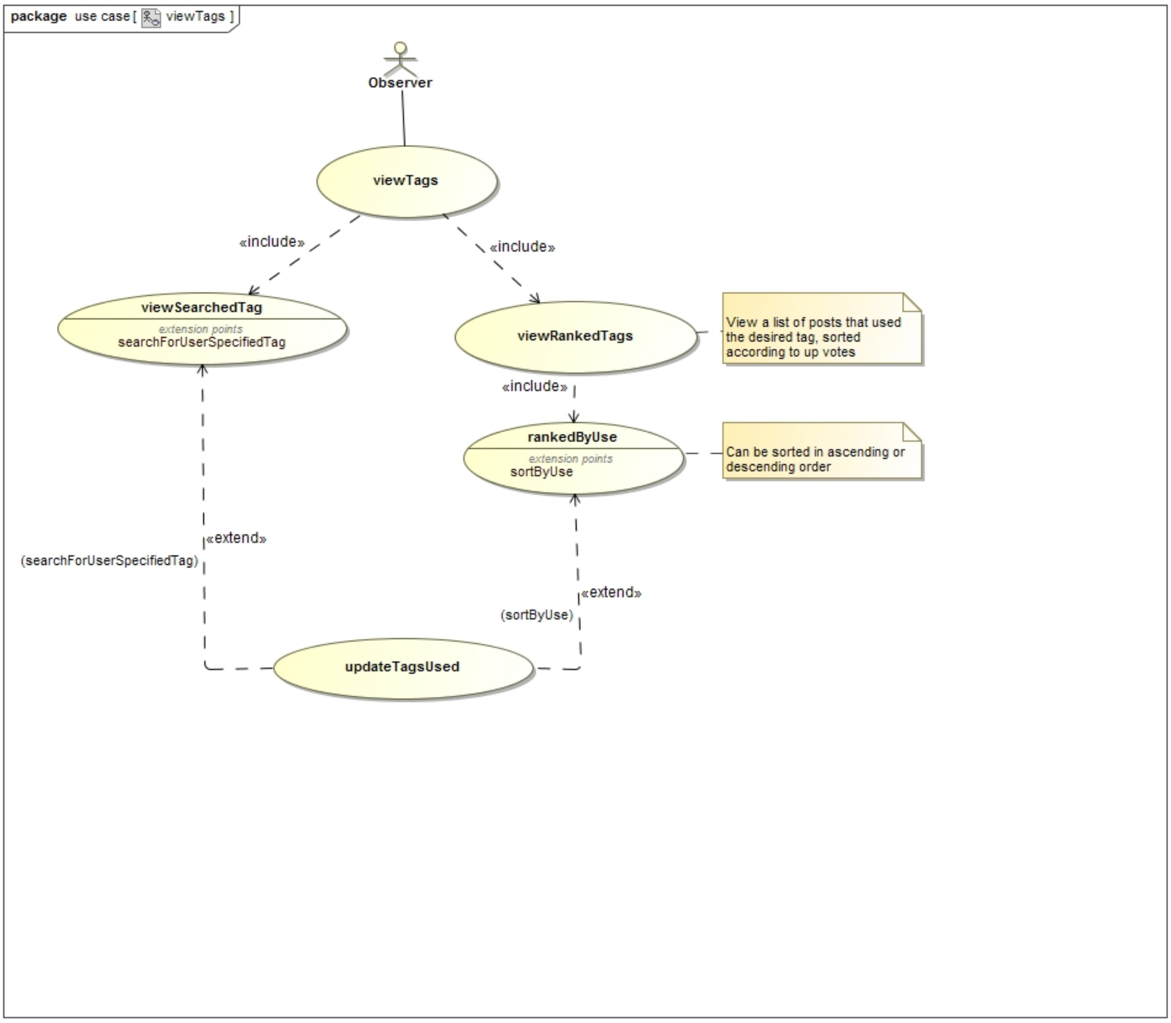
- **BuzzSpace**. A Buzz Space is a integral component of the Buzz System which facilitates the management of threads added by its users. Buzz Spaces may be created for each active module in the Computer Science Department in order to promote intuitive communication between the Computer Science staff and its students.



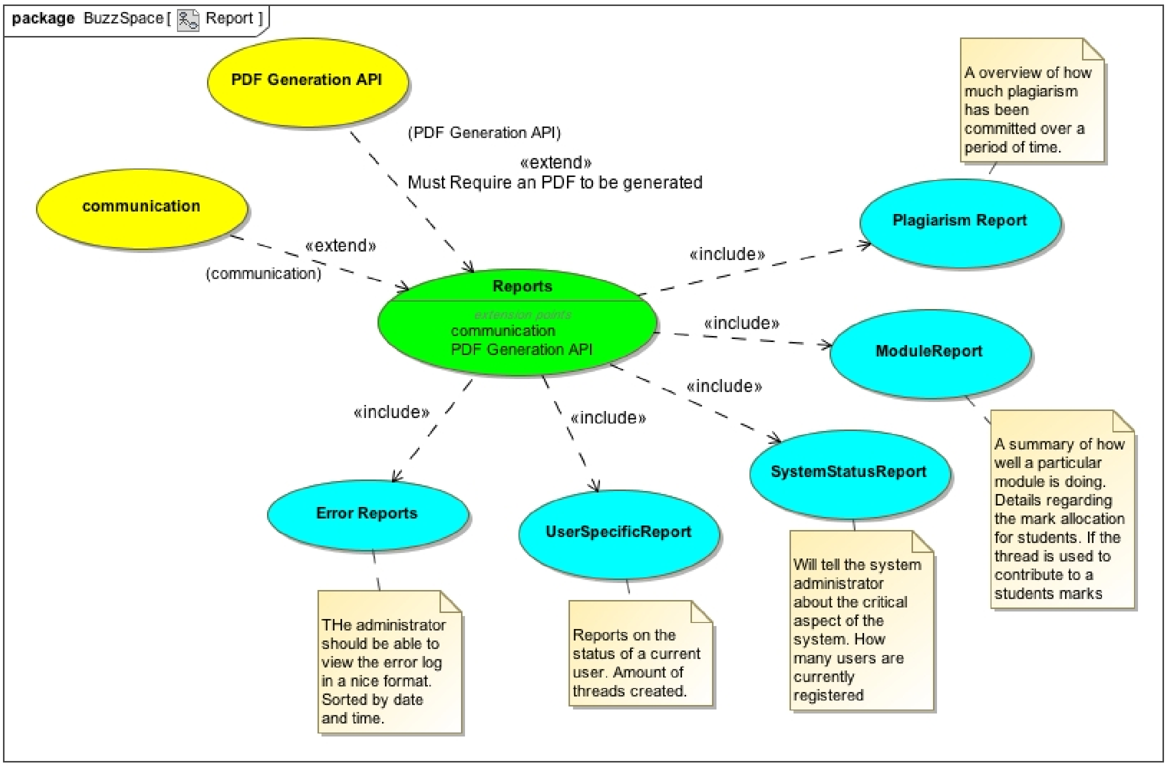
- **Validation**. This module will validate the any post send to it in terms of plagiarism check and also to check to see if the post follows certain netiquette rules. The post will also make sure of valid user acces by using the Usermanagement system. This system will also help the Report system to generate certain reports regarding the above validation. This module will only ever be used bu other modules and not directly by the user. It will simply just make sure that all the rules that must be followed for a post to be sufficient for the Buzz space threads.


package Data [Validation]
This will make sure that users are allowed to post certain posts
This will use the API to check if post is plagiarized
Online plagiarism checker API
Check if user status is valid for post
Check post for plagiarism
«extend»
(API Plagiarism checker)
«include»
«include»
Print reports
«extend»
(Print report)
Validation
extension points
API Plagiarism checker
Print report
«include»
Check netiquette errors
This will write different reports for the Reports module

- **Information Management**

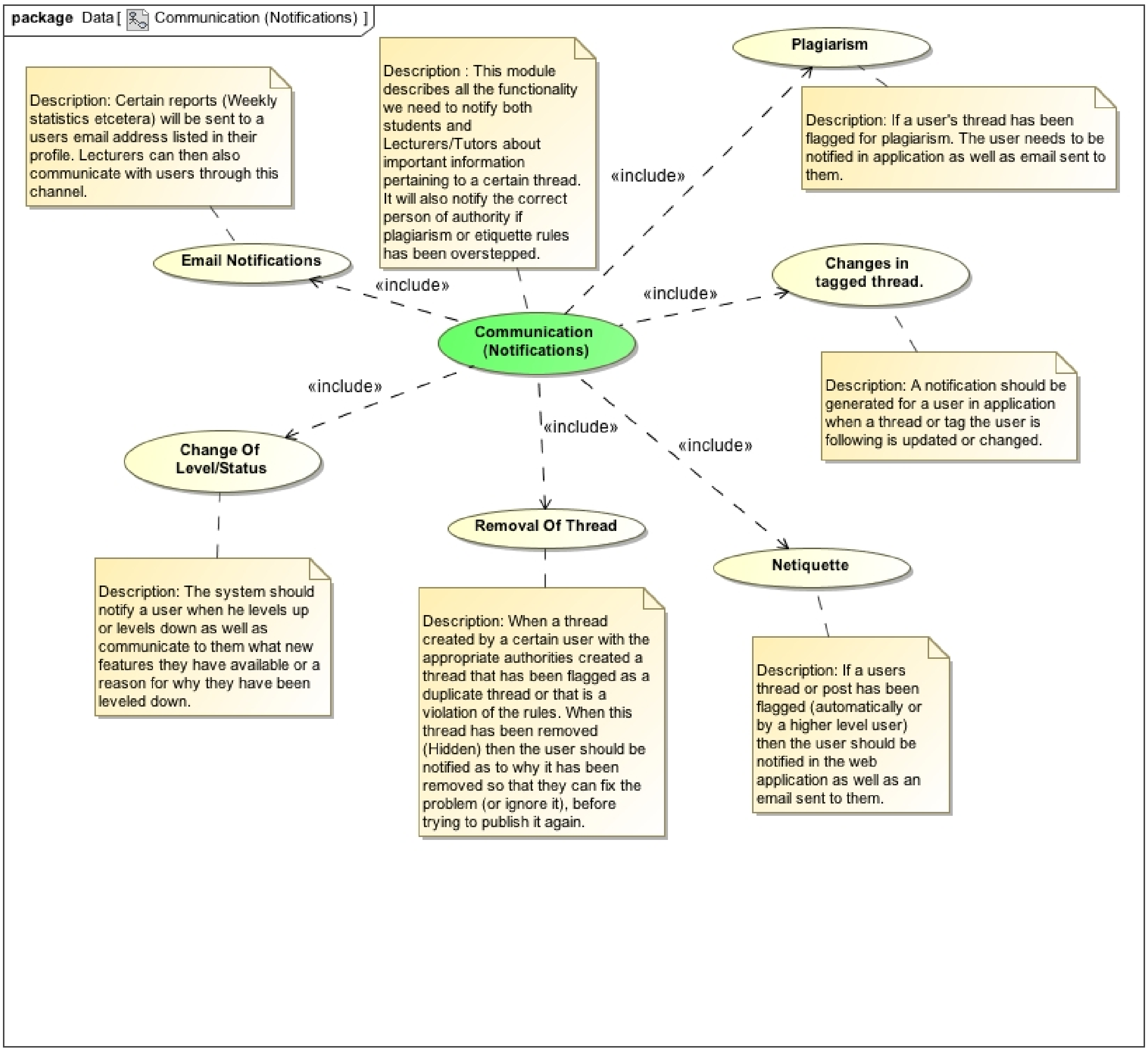

package use case [viewTags]
Observer
viewTags
«include»
«include»
viewSearchedTag
extension points
searchForUserSpecifiedTag
viewRankedTags
View a list of posts that used the desired tag, sorted according to up votes
«include»
rankedByUse
extension points
sortByUse
Can be sorted in ascending or descending order
«extend»
(searchForUserSpecifiedTag)
«extend»
(sortByUse)
updateTagsUsed

- **Reporting**. We will use the reporting module to generate quite a few reports regarding the Buzz Space system. It will be a key player in adding value to lecturers and students. Each student can easily general a report regarding their own contributions towards a Buzz Space. Lecturers will be able to grade student performance and see how much plagiarism has occurred. The system administrators will be able to check for system bugs and see error logs.


package BuzzSpace [Report]
PDF Generation API
(PDF Generation API)
«extend»
Must Require an PDF to be generated
communication
«extend»
(communication)
Reports
extension points
communication
PDF Generation API
«include»
Plagiarism Report
A overview of how much plagiarism has been committed over a period of time.
«include»
ModuleReport
«include»
SystemStatusReport
A summary of how well a particular module is doing. Details regarding the mark allocation for students. If the thread is used to contribute to a students marks
«include»
Error Reports
«include»
UserSpecificReport
THe administrator should be able to view the error log in a nice format. Sorted by date and time.
Reports on the status of a current user. Amount of threads created.
Will tell the system administrator about the critical aspect of the system. How many users are currently registered

- **Communication (Notifications)**

  This module describes the way the Buzz System will communicate with its users inside of the application as well as sending information and/or reports from the Buzz system to an external system such as email notifications.



Description: Certain reports (Weekly statistics etcetera) will be sent to a users email address listed in their profile. Lecturers can then also communicate with users through this channel.

Description : This module describes all the functionality we need to notify both students and Lecturers/Tutors about important information pertaining to a certain thread. It will also notify the correct person of authority if plagiarism or etiquette rules has been overstepped.

Description: If a user's thread has been flagged for plagiarism. The user needs to be notified in application as well as email sent to them.

Description: A notification should be generated for a user in application when a thread or tag the user is following is updated or changed.

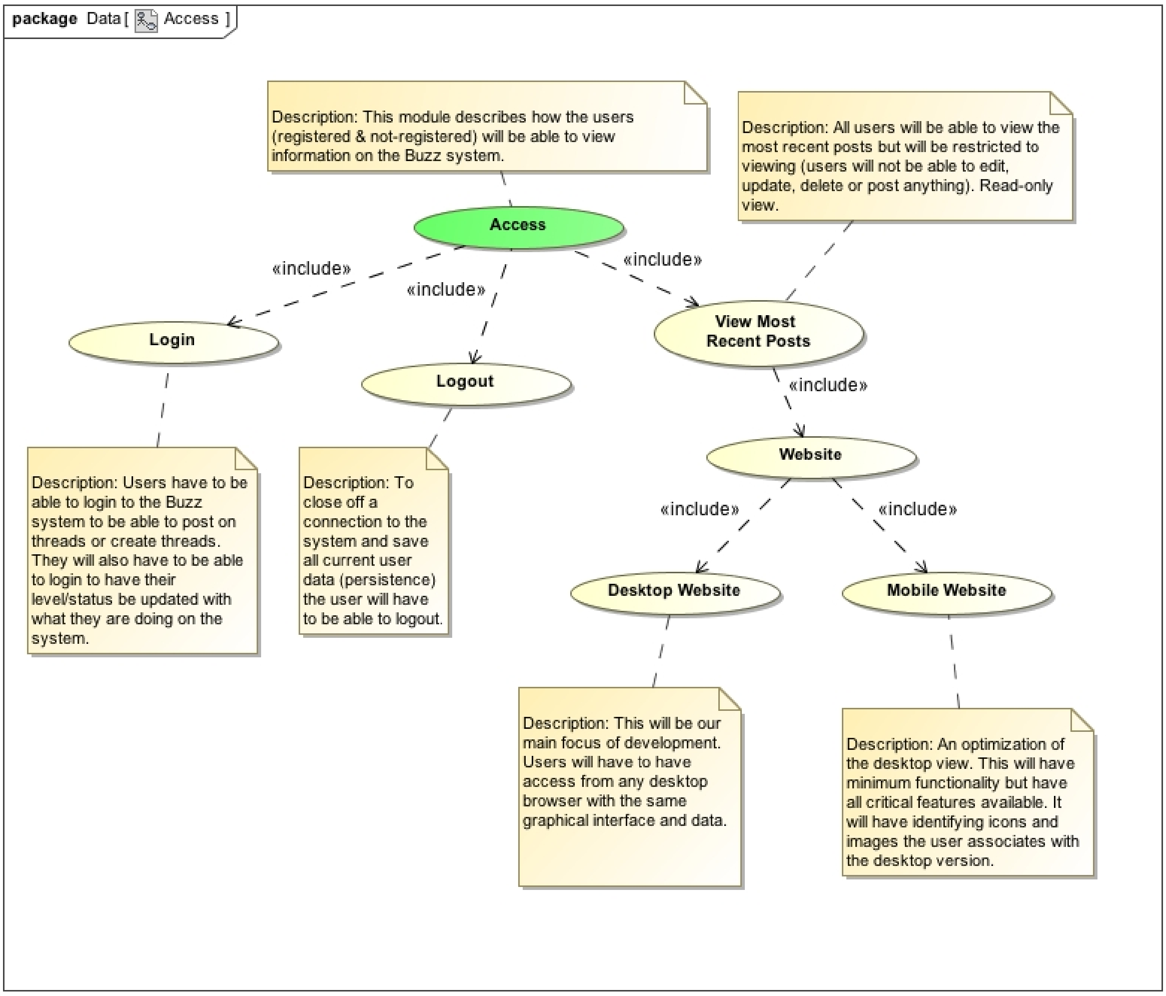
Description: The system should notify a user when he levels up or levels down as well as communicate to them what new features they have available or a reason for why they have been leveled down.

Description: When a thread created by a certain user with the appropriate authorities created a thread that has been flagged as a duplicate thread or that is a violation of the rules. When this thread has been removed (Hidden) then the user should be notified as to why it has been removed so that they can fix the problem (or ignore it), before trying to publish it again.

Description: If a users thread or post has been flagged (automatically or by a higher level user) then the user should be notified in the web application as well as an email sent to them.

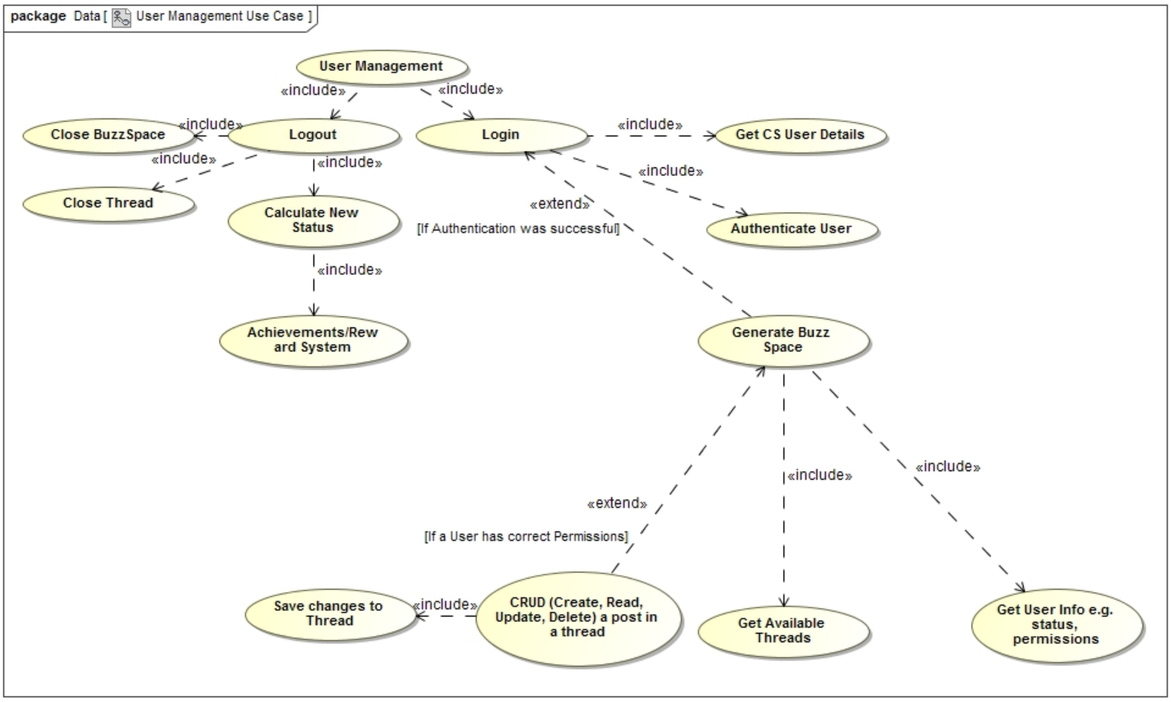
- **Access**

  The use case below shows how a end-user will be able to access the Buzz system. Although a user may not be registered, they should still be able to view and read threads and posts.


package Data [Access]
Description: This module describes how the users (registered & not-registered) will be able to view information on the Buzz system.
Access
«include»
«include»
«include»
Login
Logout
View Most Recent Posts
Description: All users will be able to view the most recent posts but will be restricted to viewing (users will not be able to edit, update, delete or post anything). Read-only view.
«include»
Website
«include»
«include»
Desktop Website
Mobile Website
Description: Users have to be able to login to the Buzz system to be able to post on threads or create threads. They will also have to be able to login to have their level/status be updated with what they are doing on the system.
Description: To close off a connection to the system and save all current user data (persistence) the user will have to be able to logout.
Description: This will be our main focus of development. Users will have to have access from any desktop browser with the same graphical interface and data.
Description: An optimization of the desktop view. This will have minimum functionality but have all critical features available. It will have identifying icons and images the user associates with the desktop version.

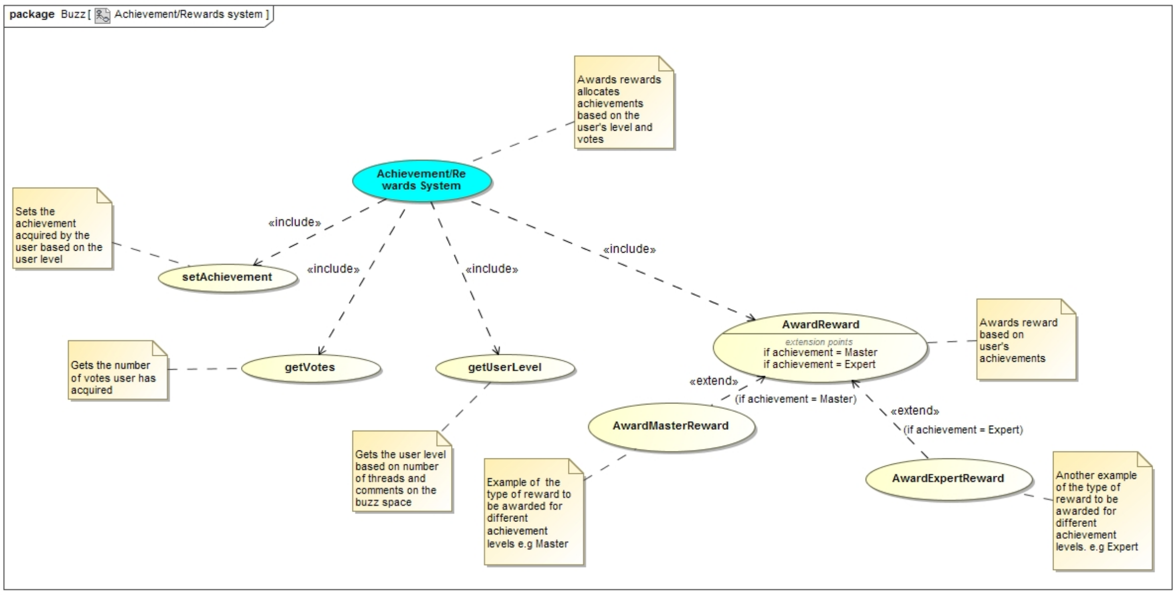
- **User Management**

  This use case specifies how the user's themselves will be managed (not their data). This refers to who will be allowed to firstly login, access a buzzspace and then a specific thread and eventually logout. This displays what services help acheive User Management and also which servies are subsets of other services.



- **Achievement/Rewards system**

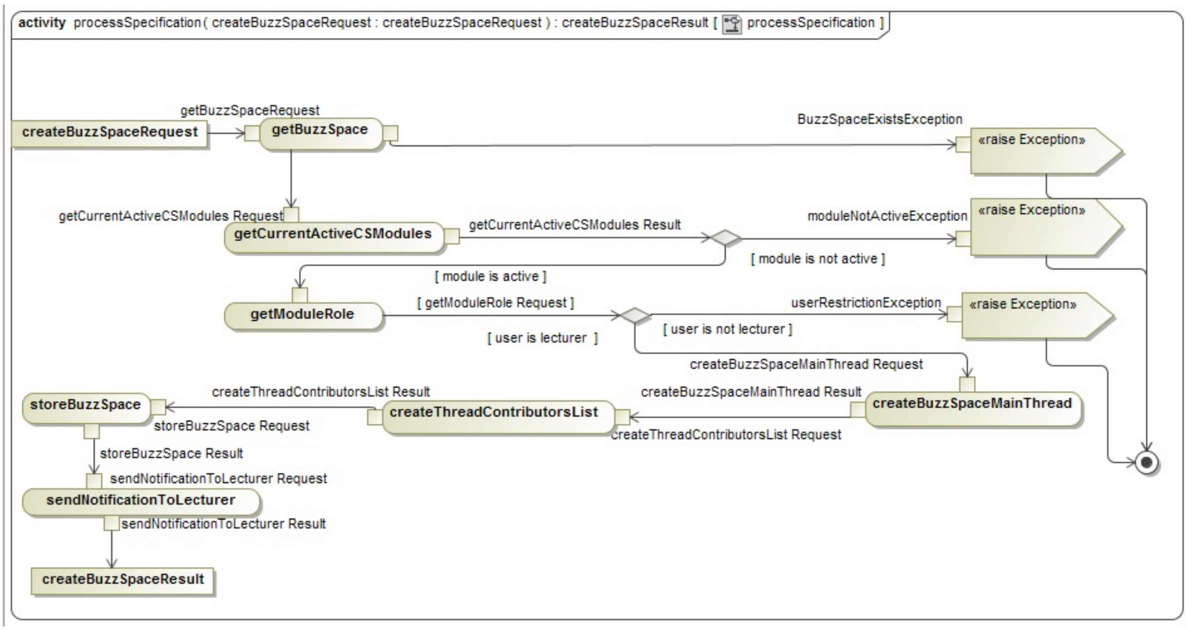
  The Achievement/Rewards system use case component shows how the Buzz System generates and awards rewards to users, based on their different achievements. The achievemnet is derived from each user's level of participation on the Buzz Space as well as the number of votes they acquire. The Achievement/Reward system incorporates the gamification functionality of the Buzz System. Therefore, forcing the users of the system to participate more often as there will be rewards for this.


package Buzz [Achievement/Rewards system]
Awards rewards allocates achievements based on the user's level and votes
Achievement/Rewards System
Sets the achievement acquired by the user based on the user level
«include»
«include»
«include»
«include»
setAchievement
Gets the number of votes user has acquired
getVotes
getUserLevel
AwardReward
extension points
if achievement = Master
if achievement = Expert
Awards reward based on user's achievements
«extend»
(if achievement = Master)
«extend»
(if achievement = Expert)
AwardMasterReward
AwardExpertReward
Gets the user level based on number of threads and comments on the buzz space
Example of the type of reward to be awarded for different achievement levels e.g Master
Another example of the type of reward to be awarded for different achievement levels. e.g Expert

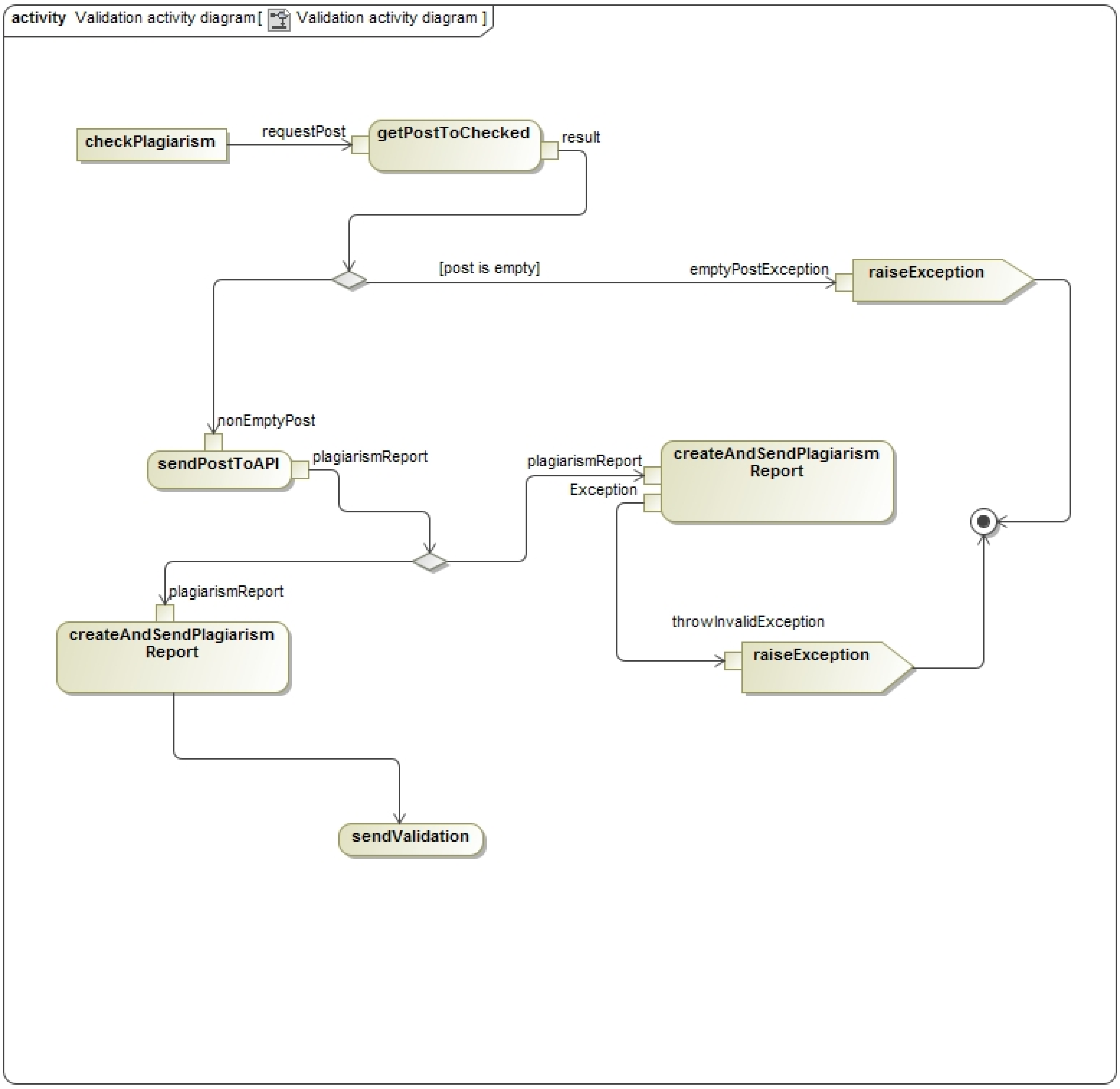
1.5 Process specification

We want to show various important process specification of our recommendation.

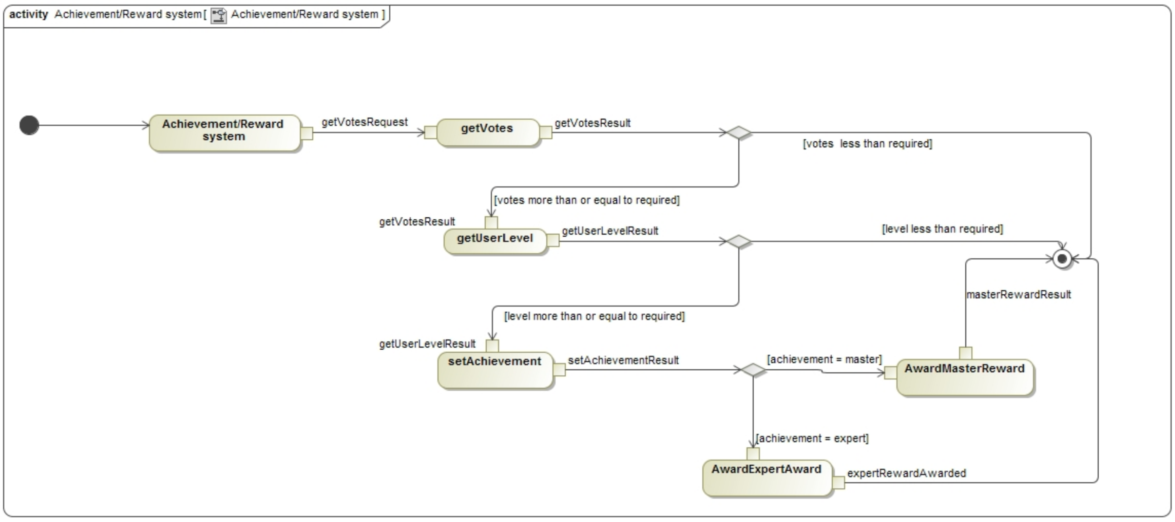
- CreateBuzzSpace


activity processSpecification(createBuzzSpaceRequest : createBuzzSpaceRequest) : createBuzzSpaceResult [processSpecification]
createBuzzSpaceRequest
getBuzzSpaceRequest
getBuzzSpace
BuzzSpaceExistsException
«raise Exception»
getCurrentActiveCSModules Request
getCurrentActiveCSModules
getCurrentActiveCSModules Result
moduleNotActiveException
«raise Exception»
[module is not active]
[module is active]
getModuleRole
[getModuleRole Request]
userRestrictionException
«raise Exception»
[user is not lecturer]
[user is lecturer]
createBuzzSpaceMainThread Request
createBuzzSpaceMainThread Result
createBuzzSpaceMainThread
createThreadContributorsList Result
createThreadContributorsList
createThreadContributorsList Request
storeBuzzSpace
storeBuzzSpace Request
storeBuzzSpace Result
sendNotificationToLecturer Request
sendNotificationToLecturer
sendNotificationToLecturer Result
createBuzzSpaceResult

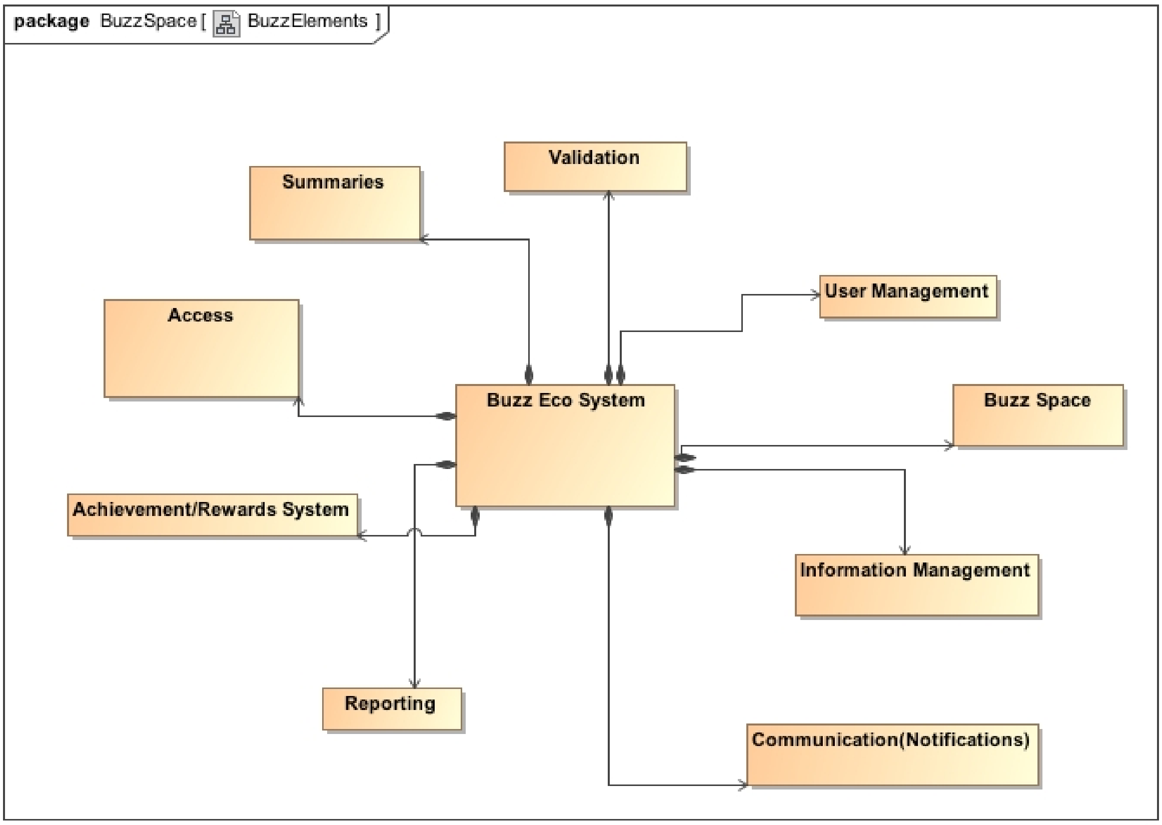
- Validation



- Achievement/Rewards system



1.6 Domain Model


package BuzzSpace [BuzzElements]
Validation
Summaries
User Management
Access
Buzz Eco System
Buzz Space
Achievement/Rewards System
Information Management
Reporting
Communication(Notifications)