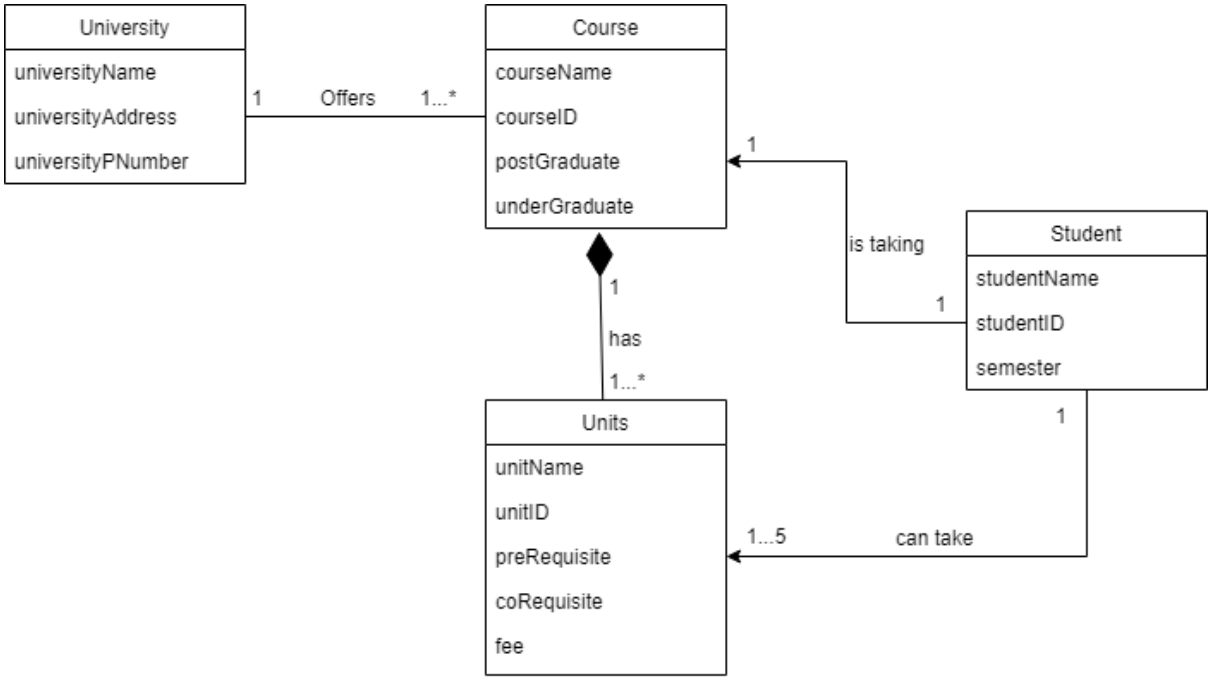
Credit task 6.2 : System Modelling

Part 1: Domain Model



Part 2: Models in software development project.

Software development models are different processes or methodologies that are used in a project's task or function and the method that is used is dependent on the software development team's goals and objectives. There are different types of models like visual, non-visual, static and dynamic. An example of static would be the waterfall methodology and an example of dynamic would be the scrum method.

Models in software development project is important because it helps a software development team to plan, design, build, release and review a project. Choosing the right model is important as it can help a company or software development team to save time and money but still achieve the result that they want according to their objectives. Other than that, modelling also helps to draw out the blueprint of the project that we want to create. Modelling also allows us to design the pattern or how we want a system to be according to our objectives. Finally, modelling is important because it allows us to record down the decisions or changes that we have made on a system.

Part 3: common / domain vocabulary

Common / domain vocabulary is required as it helps to highlight the important elements, facts, relationships, and procedures in a software development project so that people who are not familiar with programming has a better understanding of what is happening in a software development project.

Example of 5 words from the context of the case study and their definition:

a) Course: A plan of study on a particular subject.
b) Unit: A single object or something part of a bigger object.
c) Student: A person who is studying in a school, college or university.
d) Semester: One of the periods into which a year is divided at college or university
e) Fee: A specific amount of money that is needed to be paid off for a particular object in this case a particular unit.

Part 4:

a) Eddie is enrolling for units that he wants in Swinburne University of Sarawak through their website.

b)

1) Alternatives: Eddie can go to Swinburne University of Sarawak physically and fill out the hard copy form from there to enroll into a unit.

2) Extensions:

1a) Eddie enters his information wrongly.

I. System will check the information on the form submitted by Eddie and cross reference it to the database in Swinburne University of Sarawak.

II. System will signal an error to Eddie and request that he enter his information again.

2a) Website is down for maintenance

I. System will inform Eddie that the website to fill out the form for enrolling into a unit is under maintenance and request that Eddie try again later.

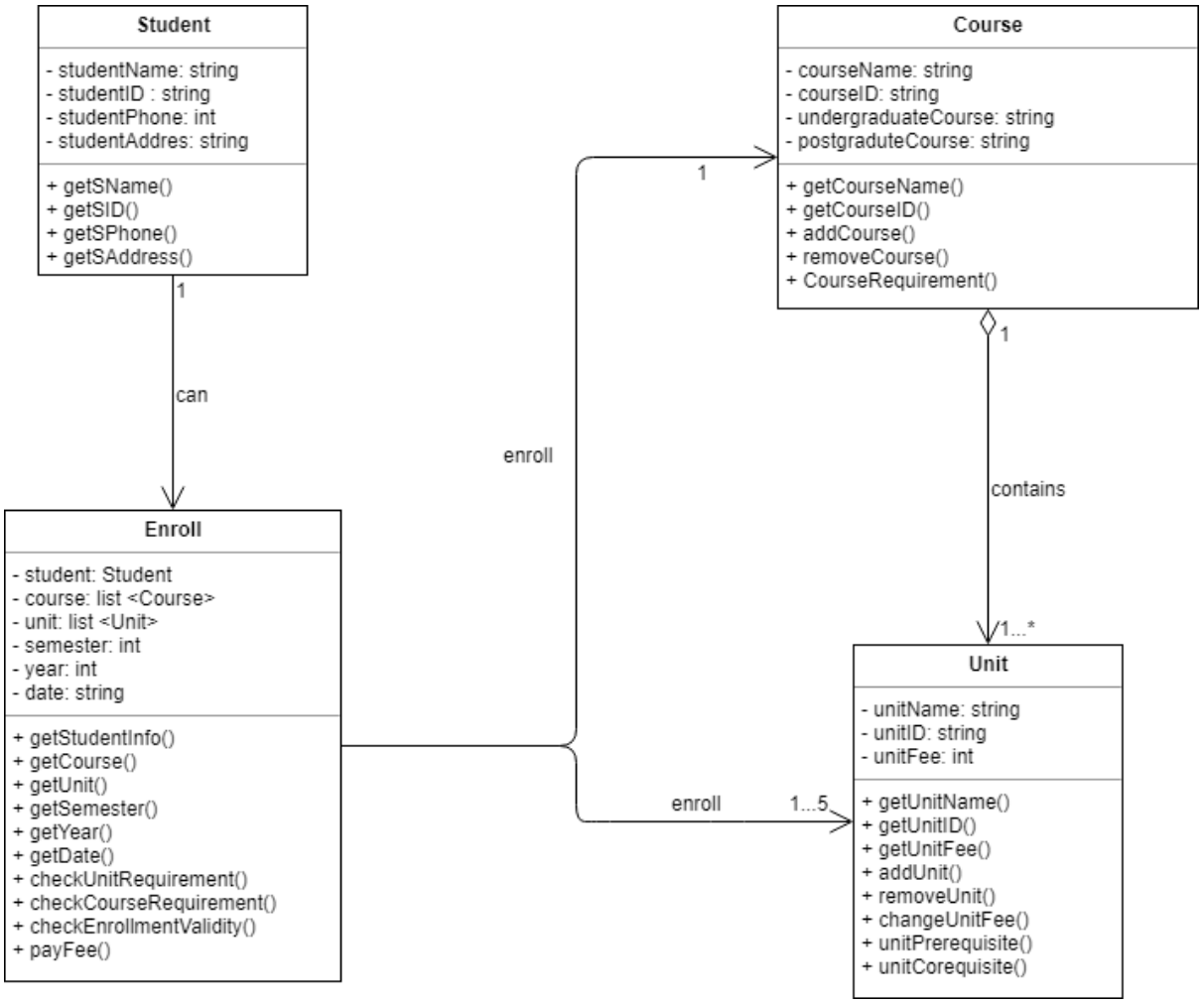
c) UseCase:

a. Name: Eddie it trying to enroll into a unit.
b. Primary actor: Eddie.
c. Pre-condition: Eddie is a student in Swinburne University of Sarawak and his information is authenticated.
d. Success guarantee: Eddie's information is stored. A success message that says Eddie has successfully enrolled into the unit that he wants after all the information that Eddie entered has been checked and is valid.
e. Main success scenario:
    i. Eddie fills out the online form that he obtained from the Swinburne University of Sarawak's website to enroll into a unit.

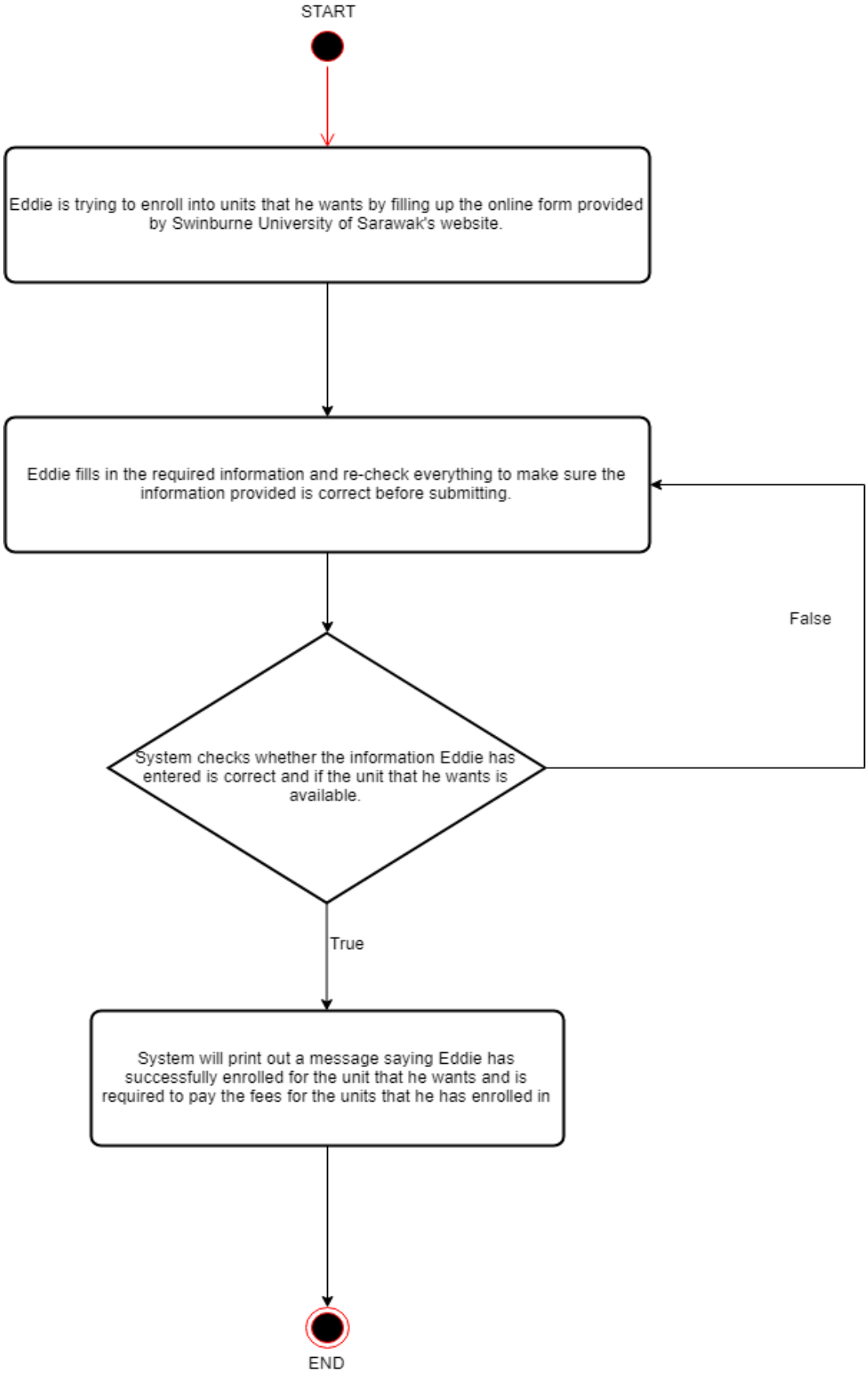
ii. Eddie makes sure that his student information, units that he wants to enroll in, and the semester that he wishes to enroll in is correct before submitting it.
iii. System will check the information on the form that Eddie has submitted and verify all the information.
iv. System will then print out a success message that indicates Eddie has successfully enrolled into the unit that he wants.

f. Extensions:
    i. Unit is not available for the semester
        1. System will alert Eddie that the unit that he wants to take is not available for the semester.
        2. System will prompt Eddie to refill the enrolment form again but with a different unit this time.
    ii. Eddie submits the enrolment form after the due date
        1. System will check the enrolment form that is submitted by Eddie and verify all the information on it.
        2. System will print out a success message that indicates Eddie has successfully enroll into the units that he wants but with a note saying Eddie is required to pay the late penalty fee.
    iii. Enrolment form is not filled out completely.
        1. System will print out an error message saying some of the required information is not filled out by Eddie.
        2. System will show Eddie which information that he has not filled out yet by highlighting the text box with the color red.
    iv. Eddie has not paid the remaining fees from last semester.
        1. System will reject the enrolment form submitted by Eddie with a warning message saying that Eddie must pay the remaining fees from last semester before he could enroll into new units for this semester.
    v. Eddie could not load the enrolment form page.
        1. System will print out a message saying that the enrolment form page is not available.
        2. System will bring Eddie back to Swinburne University of Sarawak’s main website.

d) Class Diagram:



e) Activity Diagram:



Resources used:

Boyd, NS 2009, *Domain Vocabulary*, Educery, viewed 20th April 2019,

<http://educery.com/educe/patterns/domain-vocabulary.html>

Activity Diagram - Activity Diagram Symbols, Examples, and More **n.d.**, *Activity Diagram*, smartdraw, viewed 20th April 2019,

<https://www.smartdraw.com/activity-diagram/>

SWE20001 Development Project 1: Tools and Practices 1
Credit Task 6.3 Version Control

Aldalton Choo Chien Khin (Student ID: 101212783)

Credit Task 6.3 Version Control

Version Control:

Version control system helps a software team to handle the changes that have been made on to the source code over time. It helps to keep an eye on changes that have been made to the source code in a special database storage. If an error has been made by one of the members in a software team, they can retrieve an older version of the source code and this can help to identify and fix the errors without causing a huge disturbance to the progress of the other software team members. There are several kinds of version control systems available in the market but the ones that I am going to talk about is the centralized and distributed version.

CVS or also known as Centralized Version Control System is where the single master copy of a project or a source code is stored in a central server and all version history or changes that has been made on a project or source code is also stored in the central server. This means that all developers must push their changes to the central server and other developers working on the same project can see the changes that has been made once it is committed. CVS only allows one developer to edit or make changes to the project at a time and once the developer is done editing and commits the changes to the central server, CVS then allows other developer to edit the project. One of the disadvantages of CVS is if the central server is down, developers won't be able to save their changes to the central server.

DVCS or also known as Distributed Version Control System on the other hand does not rely on a central server. Instead, it allows the developers to clone or copy the repository and this means that the developers will have the entire history of a project on their local computer. Developers will then be allowed to make changes on the project without being locked out by the central server unlike in CVS only one developer is only allowed to make changes at a time. Once the developers are satisfied with the changes that they have made, they will have to ask the developer or person that created the master copy to push the changes that they have made into the master copy as the owner is the only one with the power to do so. One of the disadvantages of DVCS is that when developers work on the same project at the same time, merge conflicts tends to happen.

How does Version Control work?

Version control systems uses a database of changes called repository and a working copy of where a developer is doing their work. On the repository, version control system usually stores the information of all the edits that a developer has made on their projects and historical/older version of the developer's projects.

A working copy is also known as the private or individual duplicate of all the files in a developer's project that is usually located on the developer's local computer. The developer can make changes to their working copy without having the fear of disrupting the progress of the other members in the software team and once they are done with the editing or changes made to the project, the developer can commit these changes to the repository. Once a commit is pushed, the other team members can update their own working copy from the repository and see the changes that has been made by a developer of the software team.

Version control system also allows different members in a software team to work simultaneously on the same project but if two or more team members pushed their changes on the same line in the same file of the same project at the same time, a conflict occurs, and this is called a merge conflict.

To avoid or resolve merge conflicts, sharing changes that has been made frequently is one of the ways to do it. Once you are done editing or making changes to the project and have committed the changes, share it with your software team members as soon as possible. Your team members can then update their working copy with the changes you have made it in and this helps to avoid conflicts and manual intervention needed to undo the merge conflict. Another method to prevent merge conflicts is splitting different tasks to different team members. This means two or more developers cannot make changes to the same line of codes because they have different tasks assigned to them. This helps to avoid merge conflict and at the same time increase the quality of code written. The last method to avoid merge conflicts is keep the changes you have made small and frequently commit them to the repository. This helps to lessen the chances of merge conflicts from occurring and even if a merge conflict did occur, it will not take much time and effort to fix the merge conflict as the changes made to the code is small.

Different version control software in the existing market

The three version control software available on the existing market that I have chosen are:

- Git
- Bazaar
- Apache Subversion (or also known as SVN)

All these version control software are available free to everyone and is open source but each version control software provides different features to its users. For Git, it is known to have super-fast and efficient performance when compared side by side with the other two version control software. For Bazaar, it allows the developers to work with or without a central server. For SVN, it supports atomic commits, where either all edit, or changes made to the master copy are applied or none are applied and this helps to prevent data corruption in the database.

Git

Advantages:

- Available on multiple platforms
- A powerful version control software and easy to maintain
- Changes made on codes can be easily tracked
- Fast operational speed and more efficient.

Disadvantages:

- More complicated and larger history log is harder to understand
- Timestamp preservation is not supported.
- Linux has more support compared to windows

Bazaar

Advantages:

- Directories tracking is supported well in Bazaar
- Plugin system is easier to understand and use
- High storage efficiency and speed
- Available on multiple platforms

Disadvantages:

- Does not support cloning
- Does not support Timestamp preservation

SVN

Advantages:

- Easier to set up and administer
- Have better windows support unlike in Git
- Has a benefit of good GUI tools like TortoiseSVN
- Integrates well with Windows.

Disadvantages:

- Does not record down the time of when a file is modified
- Does not manage the filename normalization properly
- SVN uses centralized version control system so this mean one person is only allowed to edit their code at a time.

Resources used:

Ernst, M 2012, *Version control concepts and best practices*, University of Washington Computer Science & Engineering community, viewed 20 April 2019,

< https://homes.cs.washington.edu/~mernst/advice/version-control.html>

Manandhar, G 2016, *3 simple rules for less or no git conflicts*, Geshan's Blog, viewed 20 April 2019,

< https://geshan.com.np/blog/2016/04/3-simple-rules-for-less-or-no-git-conflicts/>

Ramos, J 2018, *Recommendations to avoid merge conflicts*, ITNEXT, viewed 20 April 2019,

<https://itnext.io/recommendations-to-avoid-merge-conflicts-845ec133676e>

Ecker, R 2016, *4 simple tricks to avoid merge conflicts*, The Team Coder, viewed 20 April 2019,

<https://team-coder.com/avoid-merge-conflicts/>

Rawson, R 2019?, *2019 version control software comparison: SVN, Git, Mercurial*, Time Doctor, viewed 20 April 2019,

<https://biz30.timedoctor.com/git-mecurial-and-cvs-comparison-of-svn-software/>

Gupta, L 2012?, *How distributed version control system works?*, HowToDoInJava, viewed 20 April 2019,

<https://howtodoinjava.com/vcs/how-distributed-version-control-system-works/>

Johnson, E 2014, *CVCS & DVCS: The needs That Version Control Systems Serve*, INTLAND SOFTWARE, viewed 20 April 2019,

<https://content.intland.com/blog/sdlc/the-needs-that-version-control-systems-serve>

Lionetti, G 2012, *What is version control: centralized vs. DVCS*, Atlassian, viewed 20 April 2019,

<https://www.atlassian.com/blog/software-teams/version-control-centralized-dvcs>

What is version Control 2012?, Atlassian, viewed 20 April 2019,

<https://www.atlassian.com/git/tutorials/what-is-version-control>

15 BEST Version Control Software (Source Code Management Tools) 2018, Software Testing Help, viewed 20 April 2019,

<https://www.softwaretestinghelp.com/version-control-software/>