UECS2363 SOFTWARE CONSTRUCTION AND CONFIGURATION
CHAPTER 0 : INTRODUCTION

LOO YIM LING
yllo@utar.edu.my

Dr FARIZUWANA
farizuwana@utar.edu.my

TIMETABLE

Group	Day	Timing	Location (MS Teams Code)
Lecture			
1	Friday	10:30am – 12:30pm	**b9v18c6**
Practicals			
P1	Tuesday	1:00pm – 3:00pm	**fko49g2** (Channel: **Practical Group 1**)
P2	Wednesday	11:00am – 1:00pm	**fko49g2** (Channel: **Practical Group 2**)
P3	Thursday	11:00am – 1:00pm	**fko49g2** (Channel: **Practical Group 3**)
P4	Friday	8:30am – 10:30am	**fko49g2** (Channel: **Practical Group 4**)

TEACHING PLAN (I)

Week	Lecture Topic (including sub-topics)	Practical	Assessment
1	**Topic 1: Writing Clean and Effective Code**	-	-
2	**Topic 1: Writing Clean and Effective Code (Con't)**	Practical 1: Using methods and objects properly	-
3	**Topic 2: Code Refactoring**	Practical 2: Working with classes and error handling	-
4	**Topic 2: Code Refactoring (con't)**	Practical 3: Basic concepts of refactoring	Assignment 1 (due on 30 July 2020)
5	**Topic 3: Software configuration basics**	Practical 4: Refactoring patterns I	-
6	**Topic 4: Source code control**	Practical 5: Refactoring patterns II	-

TEACHING PLAN (II)

Week	Lecture Topic (including sub-topics)	Practical	Assessment
7	**Topic 4: Source code control (con't)**	Practical 6: Introduction to versioning	Assignment 2 (due on 14 Aug 2020)
8	**Topic 4: Source code control (con't)**	Practical 7: Operating the versioning tool	-
9	**Topic 5: Building**	Practical 8: Advanced versioning issues	Group Assignment 3 (due on 4 Sept 2020)
10	**Topic 5: Building (con't)**	Practical 9: Building principles	-
11	**Topic 6: Continuous Integration & Deployment**	Practical 10: Scripts for building	-
12	**Topic 6: Continuous Integration & Deployment (con't)**	Practical 11: Continuous Integration (CI)	-
13	**Group Assignment Presentation**	Practical 12: Group Assignment Presentation	-
14	**Revision**	-	-

COURSE SUMMARY

Writing and configuring software properly so that the process of development and deployment is facilitated in a smooth and seamless manner, is an important aspect of the software life cycle.

This course introduces the principles in constructing software. The key activities of source code control, building, continuous integration and deployment are expounded upon along with practical exposure to the most common tools used in this process.

COURSE LEARNING OBJECTIVES (CO/CLO)

CLO 1	**Apply fundamental concepts of continuous integration in software development and operations**
CLO 2	**Develop scripts that automate the software construction and configuration processes**
CLO 3	**Identify tools for improvement of the quality of software construction and configurations**
CLO 4	**Construct a continuous integration project as a team**
CLO 5	**Recognize the issues in development and operations of software solutions in the industry**

COURSE ASSESSMENTS

Assessment	Marks (Total)
Continuous Assessment (CA)	60%
Assignment 1 (CLO 5)	17%
Assignment 2 (CLO 3)	17%
Assignment 3 (CLO 2)	10%
Assignment 3 (CLO 4)	16%
FINAL ASSESSMENT (FA)	40%
CLO 1	10%
CLO 2	10%
CLO 3	10%
CLO 5	10%

Compulsory Pass (CA >= 24% && FA >= 16 && (CA+FA) >= 50)

COURSE MATERIALS/REFERENCES

1) Mukherjee, J., 2015. Continuous delivery pipeline - Where does it choke?: Release quality products frequently and predictably (Volume 1). CreateSpace Independent Publishing Platform.

2) McQuaid, M., 2014. Git in Practice. Manning.

3) Burns, E. & Prakash, W., 2013. Hudson continuous integration in practice. Boston, MA: McGraw-Hill.

4) Loeliger, J. & McCullough, M., 2013. Version control with Git: Powerful tools and techniques for collaborative software development. Sebastopol, CA: O'Reilly Media.

5) Halladay, S., 2012. Principle-based refactoring: Learning software design principles by applying refactoring rules. Principle Publishing.

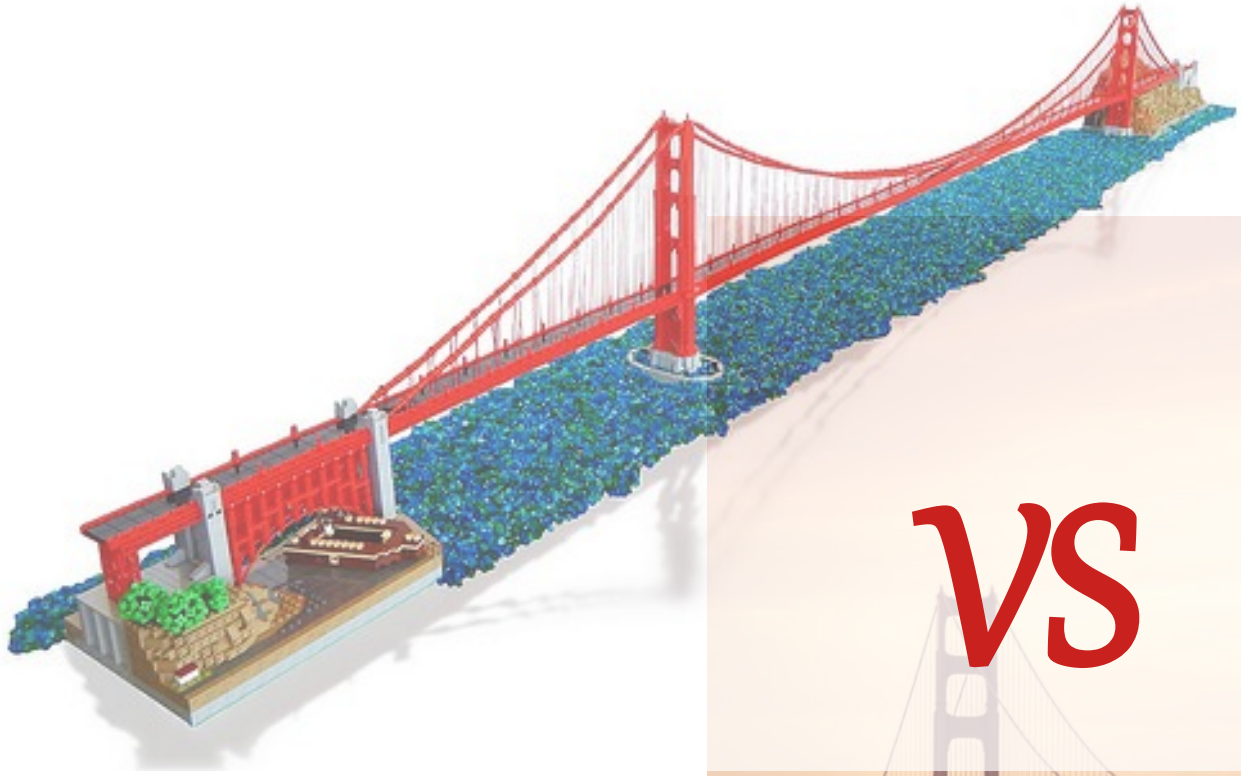
COURSE PLATFORMS AND TOOLS

- Command Line Interface (CLI)
- Git and Github/Bitbucket
- Build Tools: MSBuild, Maven, Gradle
- VM: VirtualBox
- Docker
- CI: Jenkins, Travis

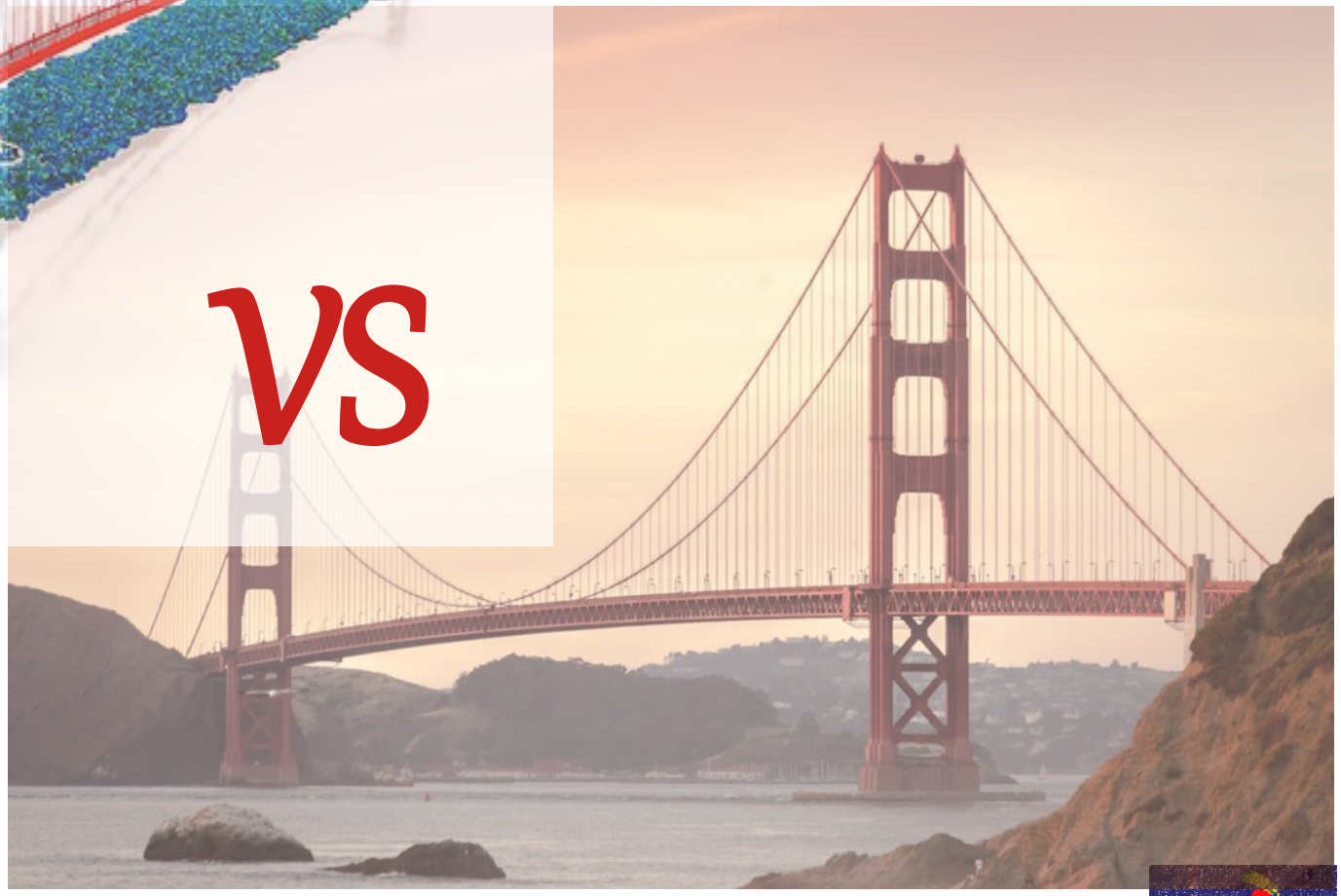
A Little Survey

- **Why Software Engineering?**
- **Interest in Programming?**
- **Programming languages and platforms?**
  - Java/C/C++/C#/VB
- **Programming Experience?**
  - Involved in group programming projects
- **Open Source project?**
- **Source Code Control software?**

Coding vs Software Engineering



VS



Software Development

- **Norm:**
  - Work in team, perhaps remotely
  - 80% “maintenance”: repairing
  - Struggle to produce quality product given limited time and ever-changing requirements
- **Hence:**
  - Elegance code
  - How to keep and retrieve source code and configurations
  - How to ensure the final software products are less buggy
  - How to rollout new software feature without messing up the production

END OF LECTURE 01