SFU Comments pertaining to Timing and Workload of Assessments

I have attended a "Learning and Teaching Workshop on Assessment Guidelines and Timing" which aimed to improve assessment and SFU feedback aspects of our units.

The workshop has reviewed the UWS Assessment Guide for criteria and standard based assessment scheme, the theory of which is primarily based on John Biggs, *Aligning Teaching for Constructing Learn*

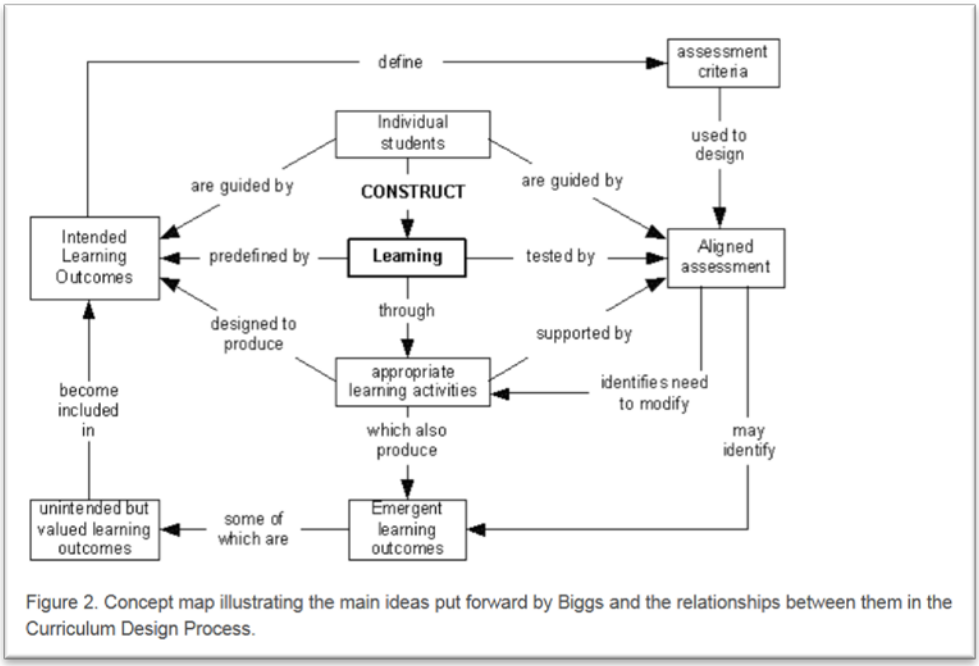


Figure 2. Concept map illustrating the main ideas put forward by Biggs and the relationships between them in the Curriculum Design Process.

The previous SFU results show that, for many units, students commented on the time scheduling, workload allocation, and difficult levels of assessment items.

I'm not surprised that students in this unit have mentioned some assessment items/activities are rather difficult that need for improvements while they have been generous to give many best aspects and high ranking in satisfaction.

SFU results review

Due to the nature of this unit – examining low level details of computer organisation and architecture, students have given many best aspects in their SFU feedbacks:

<u>Best aspects</u>

- The lecturer is very friendly, enthusiastic and has dynamic method of teaching.
- Teacher had a clear idea of what he was talking about.
- The style of teaching kept me interested all times. The teacher doesn't simply read from the lecture slides, and instead elaborates on all relevant tasks.
- There are a lot of good aspects, and I've learned a lot from this unit, not only the technical part, but also logically thinking, and I enjoyed learning it.
- Explaining low level mechanics of computer systems. Gaining a better understanding of SPIM and Computer Organisation.

- Laboratories were very helpful and effective in assisting my learning.
- … …

Also due to nature of this unit – assembly implementation of programming constructs and data structures, students have experienced difficulty in learning this unit and working on assessment activities. The primary issue from SFU that needs to be addressed revolves around time scheduling, workload allocation, and complexity adjustment of assessment items:

Needs Improvement

- The unit was very difficult and the first few weeks are okay then there is a big difficulty jump which leaves many students behind. It is hard to do well without consistent effort.
- Tutorials need to be easier.
- Quizzes need to be easier.
- … …

Difficult, difficult, and difficult. For satisfactory learning outcomes, however, we cannot reduce the learning contents and the study load any more. Due to the level of difficulty and complexity of computer organisation and architecture, we choose MIPS rather than Intel for illustrating the machine structure and instruction set architecture. For now, we will not put more new contents to the unit and keep the unit template unchanged for a period. Actually, a new edition of the text, the 5th edition, is published. It is not adopted this year though in order to remain existing contents (we may choose the new edition from next deliveries). Despite all that, this is a hard unit. This is why I don't normally approve a late enrolment for this unit, which certainly causes a negative impact on the student's academic performance.

If you really want to learn this unit and gain low-level hardware and software experience and skills, prepare to face a challenge.

Assessment structure

When designing the unit assessment items, we have paid attention on alignment of assessment tasks with unit learning outcomes and learning activities:

/* Learning outcomes and learning activities */

Learning Outcome [refer to outline]	Learning Activities		Assessment Tasks		
	Lectures	Practicals	Lab Exercises [40%]	Quizzes [10%]	Final Exam [50%]
1	✓	✓	✓	✓	✓
2	✓	✓	✓	✓	✓
3	✓	✓	✓	✓	✓

… …			… …		

The unit weekly practical task sheets have been revised year by year via students' assessment, commentary, and reflection. For instance, Practical lab-6 task sheet is currently in version 7. I have even given a student an Advanced Project to review the whole sets of unit practicals for writing clear initiative specification and direction. It is believed that initiative and direction can create a great assessment task; however initiative without direction creates a ceiling fan when a brake pad is needed.

To ease students' work and help start up the weekly practical set, sample code has been supplied for nearly every single lab task. I tried to put myself in the shoes of students.

Unfortunately, as the sets of practicals are reused year by year (without much changes), the solutions cannot be publically released.

Learning methods

The unit is difficult is due to the nature of this unit rather than the design of unit materials and assessment structure. This unit examines the low level details of computer organisation and computer architecture, such as, assembly implementation of programming constructs, data structures, and fundamental concepts like memory image for binary patterns, code segment vs. data segment, simple variables vs. arrays, characters packing, procedure calling, memory hierarchy, memory-mapped I/O, datapath and controls, instruction execution and pipelining etc.

The unit is learnt via various teaching and learning components. That is:

Unit = Lectures + Tutes + Quizzes + Exam + Text + …

For better learning results, I draw your attention to the following efficient learning experiences:

<u>Philosophy</u>

Teaching and learning are joined in a process where the goal is to provide the skills and capacity for the student to eventually become independent leaners and practitioners.

<u>Approach</u>

The unit is developed towards university education:

- Meet the unit learning objectives.
- Enhance creative problem solving skills, analytical skills & design skills.
- Expand capacity to motivate students learning for challenging tasks.
- Maintain curiosity for exploration and discovery.

<u>Methods</u>

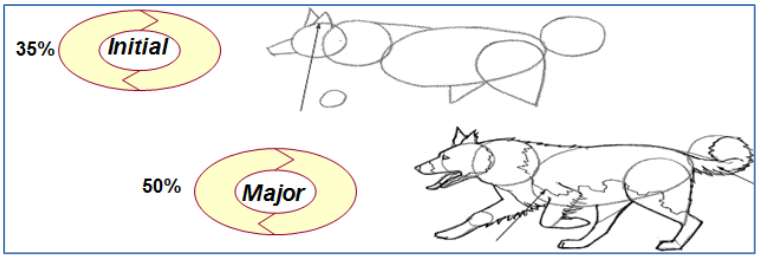
- Learning is Incremental: from knowledge comprehension to application and evaluation.

  Recall Bloom's Taxonomy of Learning Domains:

  {Knowledge, Comprehension, Application, Analysis, Synthesis, Evaluation}

- Learning is Iterative: you gain a certain percent of knowledge in each iteration of repetitive actions.

  Recall x% learning strategy:



- Practicals are even important: you can't learn driving by attending a university course or merely watching others to drive. You have to do it by yourself.
- Not only ‘search’, but ‘research’: discover problem and solve it.
- More analytical than simply descriptive: gain an insight into the problem solving process.