MDDE 619: Trends and Issues in Instructional Design

Assignment #2a: ID Model Application (Collaborative Group Activity)

Group 2: Alex Brierley, Geri De La Mare, Tanya Elias

Athabasca University

November 9, 2009

Abstract

COMP 268 is an introductory course in computer programming (Java) currently offered by the Athabasca University. After reviewing the website and content, we decided that the application of a newer instructional design model, the 4C/ID Model (developed by Van Merriënboer et al in the early 1990's), could offer some fresh insight into ways to enhance its delivery and student engagement in the course. The 4C/ID Model is comprised of four interrelated components: learning tasks, supportive information, just-in-time information and part-task practice. In this paper, we will first briefly describe the 4C/ID Model and its components, identify ways in which the current course is consistent with the model, and also identify areas of the course that could be improved in order to better reflect the model. Based on those areas, we will then present 11 specific recommendations as to how the course could be enhanced.

Commented [CI1]: Very promising

Table of Contents

Commented [CI2]: Well organized struture that supports your arguments

Whole Task Computer Programming: The Redesign of COMP 268 Using the 4C/ID Model

COMP 268 is an introductory course in computer programming (Java) currently offered by Athabasca University. The introduction of this course states that students should be familiar with programming concepts and have some knowledge for setting up and using the Java programming language compiling environment (Athabasca University, 2009). After reviewing the course website and content, we decided that the application of a newer instructional design model could offer fresh insight into ways to enhance its delivery and student engagement in the course.

Commented [CI3]: Promising…

We have selected a four-component instructional design model, entitled the *4C/ID Model*, as the most appropriate for the redesign of COMP 268. Developed by Van Merriënboer and his colleagues in the early 1990s, the 4C/ID Model is typically used for designing and developing courses that require substantial learning and provide a substantial part of a curriculum for the development of competencies or complex skills (Van Merriënboer, and Kirschner, 2008, p. 1).

Van Merriënboer et al. (2002) stated in their discussion that "a range of studies in the computer programming domain, indicated that 4C/ID strategies yielded higher transfer performance than control strategies, and this superiority became more evident on far transfer problems for which learners had to design and construct new computer programs that required solutions not encountered before" (p. 59). We therefore feel that the 4C/ID Model can provide learners with superior transferable skills in problem solving and computer programming than the current course version, COMP 268 Revision 8.

Commented [CI4]: Good rationale. Another reference that describes computer programming as a complex skill would strengthen it even more.

In this paper, we will first briefly describe the 4C/ID Model and identify ways in which the current course is consistent with the model. We will also identify areas of the course that could

be improved in order to better reflect the 4C/ID Model. Based on those areas, we will then present 11 specific recommendations as to how the course could be enhanced.

The 4C/ID Model

Commented [CI5]: Very good description of a model many people find difficult to understand

> *Van Merriënboer (1997) provides the most comprehensive recent model of instructional design that is problem centered and involves all of the phases of instruction. His model integrates more directive approaches to instruction with problem-based approaches all in the context of what is known about cognitive processing.* [italics added] (Merrill, 2001, p. 11)

Developed in the early 1990s, the 4C/ID Model is comprised of four components "for the design of training programs for complex skills…the basic claim is that four interrelated components are essential in blueprints for complex learning: (a) learning tasks, (b) supportive information , (c) just-in-time (JIT) information, and (d) part-task practice" (Van Merriënboer, Clark, and de Croock, 2002, p. 39). These are indicated in Figure 1.

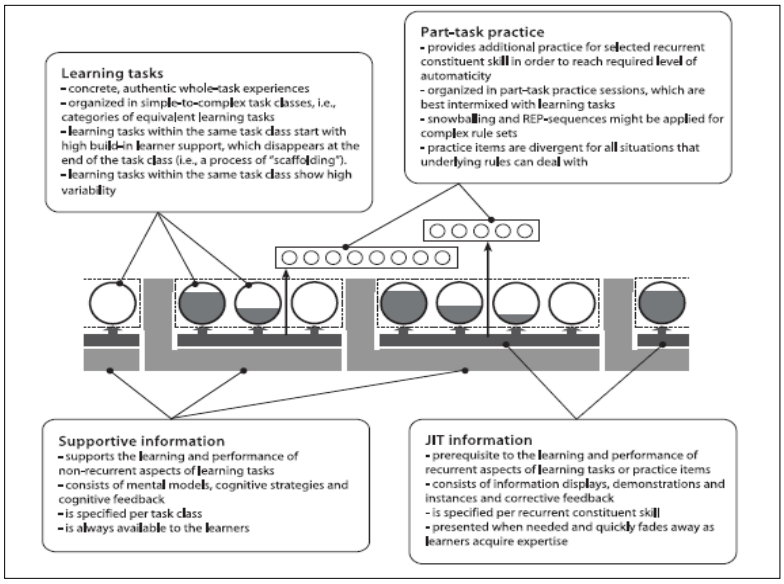

Figure 1: A graphic view of the 4 components of the 4C/ID Model. (Van Merrienboer et al., 2002)

Learning tasks. Van Merriënboer et al. (2002) state that a sequence of learning tasks is the backbone of every training program aimed at complex learning. The learning tasks are typically performed in a real or simulated task environment and provide whole-task practice and should engage learners in activities that require them to work with the constituent skills, as opposed to activities in which they have to study general information about or related to the skills (p. 43). They go on to state:

> A whole-task approach is taken, whereby the first task class refers to the simplest version of whole tasks that experts encounter in the real world. For increasingly more complex task classes, the assumptions that simplify task performance are relaxed. The final task class represents all tasks, including the most complex ones that professionals encounter in the real world. (p. 44)

Because the 4C/ID Model focuses heavily on the whole-task approach, it also stresses learning goals that go beyond a list of highly specific objectives. These learning goals seek to integrate sets of objectives (Van Merriënboer, 2007) in which skills from the cognitive, psychomotor, and affective domains (Bloom, 1956) are combined in order to give learners the skills required when dealing with transfer tasks requiring both specific and general, abstract knowledge.

In order to achieve this required knowledge, Van Merriënboer (2007) advocates the use of *mathemagenic models* which "give birth to learning" (p.77) by the random sequencing of learning tasks such that learners construct knowledge "that eventually better allows them to diagnose new malfunctions that they have not encountered before" (p.76).

In COMP 268, the learning tasks do adhere to the principles of the 4C/ID model in several ways. The current course does:

- Seek to teach knowledge, skills and attitudes related to computer programming within a single introductory course; and
- Provide units that build on one another and thus demonstrate a progression from simple to more complex task classes (V. Kumar, personal communication, October 30, 2009).

Despite~~,~~ the above consistencies, several other components of the COMP 268 learning tasks are inconsistent with the 4C/ID model. Current course weaknesses include,

1. Active programming tasks are only included as lab activities in the later stages of the unit and, thus, not the core component of each unit;
2. Programming tasks in the early units that are not presented within the greater problem-solving context of real world applications and the whole-task approach;
3. Demonstrations are often included at the end of the unit in the Optional Activities and/or Exercise Solutions rather than at the beginning of the unit;
4. Units with 8-10 highly specific, separate objectives corresponding to a single knowledge, skills or ability with virtually no integration.
5. Use of a compartmentalized, fragmented approach to identifying and solving problems rather than mathemagenic models encouraging the use of randomized and variable practice enabling the variable, real-world practice more likely to result in far transfer. (i.e., in the final exam, one question asks students to look for syntax problems and in the next question, they are told that the syntax is correct but they must identify the logic problem.)

Commented [CI6]: Nicely (positively) worded

Commented [CI7]: Do you mean across the units?

Supportive information. Supportive information provides the bridge between what learners already know, and their work on the learning tasks. It includes the information that teachers typically call "the theory", which is often presented in study books and lectures. The primary purpose of this information should be to promote schema construction through elaboration, that is, help students to establish arbitrary relationships between newly presented information elements and their prior knowledge. These relationships between new skills and prior knowledge can be developed through the reasoning process within mental models or by using cognitive strategies, such as a systematic approach to problem-solving including "rules of thumb or heuristics that guide the problem-solving process" (Van Merriënboer et al., 2002, p.47).

Commented [CI8]: I like the nested structure wherein your analysis of the course is within the discussion of each component of the model. It avoids repetition and is very engaging.

An important aspect of supportive information "relates to feedback that is provided on the quality of performance…which should promote schema construction…Well-designed feedback should stimulate learners to reflect on the quality of their personal problem-solving processes and found solutions, so that more effective mental models and cognitive strategies can be developed" (Van Merriënboer et al., 2002, p. 50).

As with the learning tasks, three elements of the supportive information offered in the current COMP 268 course are consistent with the model:

- Tips and techniques for problem-solving, non-recurrent constituent skills are taught;
- Information is provided that is directed at the construction of general schemas including requisite terms and concepts through an extensive Java Tutorials Website that is linked to the course;

- The course assessment also consists of a series of Tutor Marked Exercises (TMEs), each of which will provide the learner with feedback on progress in a unit before proceeding to more complex tasks. This feedback should be very helpful for learners as long as it is received in a timely fashion. Delays in the marking of TMEs can slow the progress of students as they wait for feedback before continuing with the next unit.

Commented [CI9]: Always a concern – sometimes tutors don't understand the importance of this, which is why there are guidelines for turnaround times.

We have also identified two areas for improvement in terms of the development and delivery of supportive information in keeping with the 4C/ID Model:

1. Large amounts of supportive information are delivered at the beginning of the each unit. Because concepts are presented in isolation, there are few chances for learners to make the required connections between both concepts and their own prior knowledge; and
2. There are no assignments directly related to the development of the mental models and metacognitive skills required to monitor and regulate task-related activities.

Commented [CI10]: Are there any activities related to this? Are assignments necessary to ensure the learning?

Procedural or *Just-in-Time* (JIT) Information. JIT information pertains to a skills component that "provides learners with the step-by-step knowledge they need to know in order to perform recurrent [routine] skills, [such as someone looking over their shoulder and giving them small pieces of information that will assist them]…it is typically provided during the first learning task for which the skill is relevant…and then fades away as learners gain more expertise" (Van Merriënboer et al., 2002, p. 51).

JIT information can also occur within a feedback process after the completion of a task "where learners learn to recognize their errors and how to recover from them. Well designed feedback should then inform the learner why there was an error and provide a suggestion or

hint [example or demonstration] of how to reach the goal." (Van Merriënboer et al., 2002, p. 52).

In terms of the 4C/ID Model, COMP 268 does:

- Provide a good amount of information for learning recurrent constituent skills which is provided in each successive unit of the programming process.
- Employ tutors who can provide JIT support within a reasonable amount of time. When the students encounter difficulties, they may access the support of tutors who can help them by offering correction and, when necessary, expert guidance.

Commented [CI11]: This phrase is not very specific – it would help if you describe what information is provided.

The current course does not:

1. Offer procedural information "in the form of directive, step-by-step instruction" (Van Merriënboer , 2007, p. 79) in the form of a job-aid, but rather offers information in a way disjointed from the actual learning task.
2. Explicitly encourage students to seek feedback from other learners or offer opportunities for students to reflect on their own mistakes and improve their metacognitive and problem-solving skills.

Part-Task Practice.

> *JIT information presentation aims at restricting encoding of newly presented information in rules; supportive information presentation aims at elaboration of existing schemata with new information. However, if a very high level of automaticity of particular recurrent aspects is required, the learning tasks may provide insufficient repetition to provide the necessary amount of strengthening. Only then, it is necessary to include additional part-task practice for those selected recurrent aspects in the training program.* [italics added] (Van Merriënboer et al., 2002, p. 53).

Part-task practice can be a slow process, as it focuses on aspects of learning that would require a repeated performance of a task until it can become automatic (over-training). Van

Merriënboer et al (2002) stress that, “Only for highly complex algorithms, represented by large rule sets, it may be necessary to work from simple to complex practice items. The whole algorithm is then decomposed into parts, and learners are extensively trained on each part separately before they begin to practice the whole recurrent skill” (p. 54).

A lot of part-task practice is currently offered in the form of Exam Preparation questions and Lab Exercises throughout the COMP 268. Part-task practice is also provided through the exercises at the Java Tutorial Site. Generally, however, this practice is currently provided after concepts have been discussed, but *before* whole-tasks have been introduced to students. To remain consistent with the 4C/ID Model, this part-task practice should instead be included *after* whole task has been introduced. In Unit 1, for example, learning to edit code may involve part-task practice.

Recommendations

Commented [CI12]: Well-written summary introduction

Given the above analysis of COMP 268 using the 4C/ID model, we have developed a list of 11 course redesign recommendations. The first nine recommendations are directly related to the areas of weakness identified in the previous section. The final two recommendations, while not directly related to the weaknesses identified, involve adjustments required to realign course evaluation and student assessment as a result of the above changes. Our recommendations include:

1. Developing a sequence of learning tasks in a simulated environment that provide whole-task practice as the backbone of the course;
2. The introduction of “pair programming”;
3. Replacing highly specific objectives with a single integrated learning goal for each unit;
4. Using mathemagenic models for variable whole-task practice to enhance far transfer;

5. Introducing a final group project intended to assess the ability of learners to address an unfamiliar situation using general, abstract knowledge;
6. Integrating supportive information into whole-task sequenced tasks.
7. Using reflective journaling and an "Experts Exchange" as ways to assist students in the development metacognitive skills and integration of supportive information;
8. Including just-in-time, step-by-step job aids when new recurrent procedures are introduced;
9. Using part-task practice only after the constituent skill has been introduced in a whole-task problem, and only when required to develop automaticity;
10. Adjusting student assessment to take into account new course elements; and
11. Introducing a more robust course evaluation process in order to assess how the above recommendations are improving learning in COMP 268.

In the following section, we will review each of the above recommendations in more detail and use specific examples from COMP 268 in order to demonstrate how they could be implemented.

Develop a sequence of whole-task practice problems as the backbone of the course. In COMP 268, a waterfall approach typical in introductory programming will be introduced and used in the first few units. The waterfall approach is a systematic approach to computer programming in which, given a problem, programmers: (1) Analyze the problem and identify an abstract solution; (2) Develop and implement a solution by writing code; and (3) Test and evaluate the program's ability to solve the original problem. Once established, more complex whole task practice involving other options available to them at each step including optimization of run time, interoperability of the data structure, the capacity to maintain the solution over time, and the possibility of parallel/distributed solutions (V. Kumar, personal communication, October 30, 2009).

Commented [CI13]: Should / could?

Commented [CI14]: Should also be added to references

Example of the application of whole-tasks in COMP 268. This section demonstrates how the whole-task approach could be applied to COMP 268 by using specific examples from the

Units 0 and 1 of Comp 268, Revision 8. Please bear in mind that these examples have been developed to demonstrate a design approach, rather than to accurately reflect content.

In Unit 0, after some basic introductory information, students could be directed to Java in Action , a site which demonstrates products that use Java programming. Students could be asked to consider, "What was the problem that each of these products solved?" From there, students would be presented a new, simple problem. In order to solve that problem, students will need to download and install Java SDK and the course compiler--instructions and a link to the help desk should be included--and then run the program. By successfully running the program, they will have both participated in their first whole task and ensured that their software is correctly installed; they should have the skills required to proceed to Unit 1.

For Unit 1, we have devised five types of sample problems designed to teach students to edit, save, print, run programs using whole task problems (see Appendix A). Because students have not yet learned how to write their own code, all problems in this unit will provide students with the actual Java code required to run programs. Students will however, be required to identify problems and abstract solutions associated to code. In several problem types, students will be expected to identify more than one possible abstract solution as "students are expected to understand and be aware that for any given problem… there could be hundreds of solutions" (V. Kumar, personal communication, October 30, 2009).

Although we have developed five sample problems, the actual number of problems will be determined by the Subject Matter Experts. More than one type of each problem and/or other combinations of similar whole task problems may be required in order to ensure adequate practice of all elements.

Commented [CI15]: Link not live

Commented [CI16]: Good use of SME in guiding your recommendations. I assume you are willing to share your paper with him. Perhaps you already have?

Commented [CI17]: These specific examples help support your proposal.

Introduce "pair programming". We also recommend that students work through the course in groups of two. In this way, their work will simulate a real-world task environment known as "pair programming". In use since the 1980s, it requires coders to work in pairs on a single piece of source code instead of on differing tasks. Also known as "Extreme Programming", Ron Jeffries (date) explains that software is built by two programmers, sitting side by side, at the same machine, ensuring that all production code is reviewed by at least one other programmer which results in better design, testing and code. He goes on to state that research into pair programming shows that this approach produces better results in approximately the same amount of time as a single programmer working alone.

Further reinforcing the value of working in pairs, a paper published by Blaschke, Brindley, and Walti (2009) found that participants seemed more active in smaller groups and yielded better learning outcomes and increased skill acquisition. The authors cite Shaw (2006) who states that "skills gained from the experience of collaborative learning are highly transferable to team-based work environments". By working throughout the course in groups of two, it is our hope that learners might learn how to successfully interact in an online environment, including the acquisition of new skills and behaviors.

In order to facilitate the learner readiness, we suggest that course materials provide guidelines as to how the pair might work together to solve a common problem by listing the tasks and roles. If the group cannot resolve their conflicts, tutors would need to be available to intervene. Careful management of groups is required in order to ensure a successful learning experience for all (Blaschke, Brindley and Walti (2009). Appendix D outlines "Directives for Fostering Effective Group Collaboration Environments" by Blaschke, Brindley and Walti (2009).

Commented [CI18]: Could be difficult in most unpaced courses, but in this case, there are approximately 200 students per year enrolled in 268, so it might work

Commented [CI19]: Compare with listing in references

Commented [CI20]: Add to references or add page number from citation

Commented [CI21]: Should be B, since it is the second appendix to be mentioned (APA)

Students will begin to work in pairs in Unit 1, where they can work together to identify abstract solutions. As the course progresses and learners begin to write their own code, they could meet synchronously with their partner via Skype or Elluminate which allow learners to screen share, or asynchronously via emails or shared document software (i.e., Google docs).

Replace highly specific objectives with a single integrated learning goal for each unit. In Unit 1, there are currently eight highly specific learning objectives. We recommend moving to the single learning goal: "Given a simple problem, learners will be able to use a waterfall approach to identify, modify and correct a solution in the form of a simple Java program." This new single, integrated learning goal helps learners make the connections between cognitive abilities associated with problem-solving and the skills involved in actually manipulating code Clearly, as the course progresses, the learning goals will also become more complex. In some units it may not be as easy to identify a single learning goal. Whether, the outcome is one, two or three goals, the objective of this exercise is to begin to consider how the knowledge, skills, and abilities required in programming are interrelated and dependent on one another.

Commented [CI22]: Were we looking at the same version of the course? I found 12 in unity 1

Use mathemagenic models for variable whole-task practice to enhance far transfer. As the course progresses, more complex whole-task practice utilizing the mathemagenic methods could help to build general, abstract knowledge that will help students when they encounter a previously untaught problem. In the case of debugging programs, for example, instead of practicing the identification and repair of errors in and orderly and systematic way, (m1c1, m1c2, m2c1, m2c2) Van Merriënboer (2007) advocates randomized practice (m1c2, m2c1,m1c1…). He explains that such random sequencing will require students to activate their routine knowledge and access it in an abstract manner in order to solve the problem. As a

result, when students are faced with a new, previously untaught problem (m3c4) they have the abstract knowledge required to solve it.

Although this randomized practice may be less efficient in instruction, long-term results make the extra time invested in the method worthwhile. Using a mathemagenic model when teaching considerations related the optimization of run time, interoperability of the data structure, the capacity to maintain the solution over time, and the possibility of parallel/distributed solutions should improve transfer in unfamiliar situations.

Commented [CI23]: This should have a reference, preferably something you haven't already cited.

Introduce a final group project In order to assess the ability of learners to address an unfamiliar situation. Having already recommended that students work throughout the course in pairs, it seemed fitting that they also complete their final projects with their partner in order to, again, reflect the collaborative nature of the programming industry. In this project, students will be given a real-world problem for which they have not yet been explicitly taught a solution. Drawing on their general, abstract knowledge developed throughout the course, and with the support of their teammate, tutors, and "Experts Exchange" learners will demonstrate their abilities to transfer what they have learned in the course.

Integrate supportive information into whole-problem sequenced tasks. In the later units, as the problems and options available become more complex, heuristics in the form of flowcharts and concept maps with interactive links to resources, or roll-overs as explanations (e.g., links to optimization tools, procedures; reminder of the best practice of providing accurate comments within the code itself; requirements for distributed computing) should be introduced as required to support learning in the whole-task environment. At the moment most of this information is presented apart from the active tasks and most of it is built in a static fashion. We

would recommend the use of more interactive elements such as links to definitions and examples within the context of the learning task in order to ensure that learners can easily access information when it is required.

Use reflective journaling and an "Experts Exchange". As students progress through their education, instructors hope that their students have time to not only retain but also reflect on the knowledge learned. The introduction of journaling and the use of an "Experts Exchange" are two ways that instructors can facilitate retention and reflection.

Below, we have outlined an example of what journaling in COMP 268 might look like. For more information on the purpose of journaling, and how to integrate it into a course, please see Appendix B.

Because the purpose of the journal is to provide motivation and encourage critical thinking about computer programming, to help the student to observe changes in patterns, and to reflect on the way that they view problems and their potential solutions, several focusing questions should be provided in order to help students focus their writing and feel satisfied about what they are writing. Among those questions could be: What did you struggle with; what did you learn; what do you wonder about OR what sorts of problems did you encounter that could not be resolved. It is also essential that learners not feel threatened about how the journal will be graded. To this end, we have developed the following criteria for journaling,

- Journals/blogs will be created and kept on the Athabasca University Me2U platform . This platform allows users to select who they want to share their journals with. Moreover, through the use of an internal AU site, privacy and security concerns should be minimized, and students can quickly access the site using their existing AU Student

Commented [CI24]: Very useful for understanding what you are proposing

Commented [CI25]: Provide references for this strategy

Commented [CI26]: Reference?

Numbers and Moodle passwords. A direct link from the COMP 268 Moodle site to Me2U should be included.

- Students will maintain their journals throughout the course, and the minimum requirement will be one journal post per unit.
- Journals will be reviewed on an ongoing basis by course tutors who will provide feedback accordingly. Students will have the option of sharing their entire journal/blog with their classmates, and even the Athabasca Community at large if they wish, or simply take sections from their journals and post those directly into the discussion forum where appropriate.

The "Experts Exchange" forum is the effectively the discussion board, located in the Me2U interface. It would allow learners to post questions, observations and discoveries that would draw upon, or benefit the community. By moving to this platform, the forum would remain accessible to learners after their course was completed allowing student to return to the site to either refresh their memories or to contribute additional information as the student progresses along their path towards their career as a programmer. The tutors would be members of the board, and would provide the initial "expertise" in the forum. It would be expected that as the forum grew over time, the students would become the "experts" themselves.

Include just-in-time, step-by-step job aids when new recurrent procedures are introduced. JIT resources can be provided within the course in the form of heuristics, flowcharts, step-by-step procedures and hints when a student is experiencing difficulty. In Unit 1, for example, in the first problem where editing code is introduced, students should be provided with a step-by-step procedure to follow. As students become more comfortable with

the procedure, those steps should be replaced with a link back to the procedural information whereby students can still access the information if required. In later questions, a single "Need Help?" link could be introduced which would in turn link to definitions, concepts and examples.

Use part-task practice only after a whole-task problem to develop automaticity. Part-task practice of constituent skills should be introduced *after* the new skill has been introduced. The course currently contains ample questions which involve part-task practice. These questions could be entered into a computerized quiz bank for students to access after a new constituent skill has been introduced within the context of a whole-task. In Unit 1, for example, practice editing and correcting errors in code may require extra practice. Using the quiz bank, students would be able to access a five-question quiz and receive automatic feedback on their skills in these areas.

Moreover, the quiz bank would allow students to self-regulate and set their own level of practice; some students may complete only five questions, others may choose to repeat the quiz several times accessing 15-20 different questions from the test bank. Students' scores could also be ranked in order to help students assess their level of learning in comparison to class averages.

Adjust student assessment to take into account new course elements. We have made nine recommendations for changes in the course structure. We have suggested the addition of journaling, a final project, and Experts Exchange. We have also indicated that small TMEs completed throughout the course support learner needs for feedback, and thus should remain an important element of the course. Given these additions, we recommend the following assessment for course activities:

50% - TMEs to be completed in pairs
25% - Final far transfer project
15% - Journaling and contributions to Experts Exchange forum
10% - Self-assessment and partner-assessment

It is important that learners have a good understanding of materials in the earlier units before moving on to more complex tasks. As a result, the TME assignments comprise the most substantial portion of the course grade. The final far transfer project was also weighted heavily because through this one project, the ability of students to successfully complete most or all of the constituent skills taught in the course and their abstract, generalized knowledge can be measured. The heavy weighting of these whole task, real-world applications of computer programming replaces the heavily weighted final exam in the current course. Like the final exam, these assignments aim to directly measure the knowledge, skills and abilities of students.

Commented [CI27]: Well reasoned justification for the proposed new grading scheme. What if the final exam can't be dismissed entirely due to policy?

We suggest journaling and contributions to the Experts Exchange forum be worth 15 per cent of the final grade in order to reflect the importance of reflection, the development of metacognitive skills and self-regulation in computer programming. Finally, we have allotted 10 per cent of the final grade to a self- and partner-assessment at the end of the course. These marks are offered as an incentive for students to do their best in helping their pair programming succeed.

Introduce a more robust course evaluation process. Throughout the paper, we have made a series of recommendations for significant changes to COMP 268. Implementing these changes will also involve significant resources in the form of time and money. In order to determine the effectiveness of these changes and to facilitate on-going improvements in the course, the introduction of a more robust course evaluation is recommended. Hanna, Gowacki-Dudka and Conceicao-Runlee (2002) advise: "You can assess your course's effectiveness before

(pilot test), during (formative evaluation) and after (summative evaluation) a module, unit or lesson. To evaluate your instructional improvement on a continuous cycle, you might consider answering the following questions:

Before Instruction:

- How well is the instruction likely to work?
- Will the instruction hold learners' interest?
- Is there an alternative way to organize the instruction to make better use of available time and resources?

During Instruction:

- What obstacles are learners encountering, and how can they be overcome?
- What can be done to maintain learner motivation?
- How can these learners be helped to better progress through the instruction?

After Instruction:

- What improvements could be made in the instruction for future use? What revisions have the highest priority?
- Did learners find the instruction interesting, valuable and meaningful?
- Were the selected instructional methods, media and materials effective in helping learners learn? (p. 39).

We encourage the course revision team to reflect on these questions as they consider the recommendations presented in this paper and to refer to Appendix C for ~~more information~~ ~~on~~further details of what the evaluation might look like for COMP 268.

Commented [CI28]: Suggestions for clearer understanding…

Conclusion

Commented [CI29]: Very effective summary of your paper. You have convinced me

We have attempted to demonstrate how the 4C/ID model might be applied to an Introductory Java programming course through the following: A reflective journal activity that will measure their perceptions and ideas about Java at the start of the course, reprised again as a capstone activity which will be used as a comparison and measure of their growth in skills and understanding; the integration of active programming/learning tasks, in paired groups in a simulated environment that take a whole task approach from tasks ranging from simple to

complex with the addition of the random sequencing of problems as part of the mathamagenic model approach to learning; supportive information provided in the form of more interactive elements such as links to definitions and examples and flash based graphics to make the materials more engaging and provide just in time information to facilitate automaticity; and the introduction of journaling that will help track their progress, concerns and provide information for their group exchange. It is hoped that this more holistic approach to the course will encourage students to undertake more programming courses and challenges in the future, and provide the basis for approaching the task with a more comprehensive view of what it means to be a programmer.

References

Commented [CI30]: Good use of resources outside course readings

Athabasca University (2009*). Computer Science (COMP) 268: Introduction to computer programming (Java) (Revision 8) – Course Syllabus.* Retrieved on October 14, 2009, from: http://www.athabascau.ca/html/syllabi/comp/comp268.htm

Bloom, B. S. (1956). Taxonomy of educational objectives: Cognitive Domain. New York: David McKay.

Balschke, D., Brindley, J. E., Walti, C., & Blaschke, L. M. (2009). Creating Effective Collaborative Learning Groups in an Online Environment. *International Review of Research in Open and Distance Learning*, *10* (3), 1-18.

Broadbent, B. (2002). *ABCs of e-learning: Reaping the benefits and avoiding the pitfalls*. San Francisco, CA: Jossey-Bass/Pfeiffer.

Chen, Sue-Jen. (2007). Instructional Design Strategies for Intensive Online Courses: An Objectivist-Constructivist Blended Approach*. Journal of Interactive Online Learning, Volume 6, Number 1, Spring 2007.*

Hanna, D.E., Glowacki-Dudka, M. Conceicao-Runlee, S. (2000). *147 practical tips for teaching online groups: Essentials of web-based education*. Madison, WI: Atwood Publishing.

Harrington, C.F., Reasons, S.G. (2005). *Online student evaluation of teaching for distance education: A perfect match?* The Journal of Educators Online, 2(1).

Jeffries, R. (2009). What is Extreme Programming. Retrieved from http://xprogramming.com/xpmag/whatisxp#test) on November 7, 2009.

Van Merriënboer, J.J.G. (1997). *Training complex cognitive skills: A four-component instructional design model for technical training*. Englewood Cliffs, NJ: Educational Technology Publications.

Van Merriënboer, J.J.G., Clark, R.E., de Croock, M.B.M. (2002). Blueprints for complex learning: The 4C/ID-Model. ETR&D., 50(2), 39-84.

Van Merriënboer, J.J.G. (2007). Alternate Models of Instructional Design approaches and complex learning. In R.A. Reiser & J.V. Dempsey (Eds.) *Trends and issues in instructional design and technology* (pp. 72-79). Upper Saddle River, NJ: Pearson Prentice Hall.

Van Merriënboer, J.J.G., Kirschner, P.A. (2008). Four Component Instructional Design (4C/ID). Retrieved on October 24, 2009, from: http://www.scitopics.com/Four_Component_Instructional_Design_4C_ID.html.

Walker, S.E. (2006). *Journal Writing as a Teaching Technique to Promote Reflection.* Journal of Athletic Training, 41(2), 216-221.

Appendix A

Prob. #	Course materials provide….	Student will….	Support./JIT Resources
1	Description of a simple problem A possible abstract solution (AS)	 Compile and run program	 Provide support info re: objects, classes, etc. Highlight sections of code and link to above
2	Description of a simple problem Implemented code	 Find possible abstract solution Highlight code sections related to AS Compile and run program	 Provide support and cues to AS
3	 Abstract solution Choices of code	Identify problem associated with AS Select code for AS from several options Compile and run program	Provide support and cues to the problem Provide support and cues to correct code
4a	 Code	Identify associated abstract solution Identify possible problem being solved Compile and run program	Provide links to support & cues if required
4b	A change in the problem	 Identify change in abstract solution Make changes to code Compile and run program	 Provide info on ‘how to edit code’
Editing part-task practice if required			
5	Description of a simple problem Code choices (with errors)	Student identifies several possible AS Select code that relates to AS Compile and run program Identify and correct errors Compile and run corrected program Student to print and save program	 Hints to locate errors & procedure ‘how to fix’ Procedure ‘how to print & save’
Error correction part-task practice if required			

APPENDIX B – Journal Writing

Reflection has been defined as a process regarding thinking about and exploring an issue of concern, which is triggered by an experience (Walker, 2006, p. 216). Leaver-Dunn and Harrelson (2002) stated that reflection distinguishes expert practitioners from their peers. An expert [programmer] uses information from previous experiences as well as the insights gained from the reflective process to improve decision-making ability. As students progress through their education, they must practice, enhance, and habitually use their reflective skills. Walker (2006) goes on to explain that, "Although many strategies exist to promote this process, one teaching method that has been used to encourage reflection is journal writing" (p. 216), which is a student's written reflections of their own unique past experiences, , challenges, frustrations, etc. related to a course, and how they might handle those experiences differently if faced with the same or similar situations in the future. Once a student has knowledge and becomes proficient at a skill…that student possesses a *knowing-in-action.* This is the 'know-how' a practitioner reveals while performing an action. They are showing competency. Knowing-in-action assists a student except when a familiar routine produces an unexpected result. When students come across a new situation…it would be beneficial for them to *reflect-on-action*, or reflect on that experience after it has happened. Unfortunately, more often than not, no time is designated for students to engage in the activity of reflection (Walker, 2006, p. 216).

> Journal writing can have many different applications based on the goals of the instructor and student…Holmes stated that by recording and describing experiences, feelings, and thoughts, students are able to recreate their experiences for additional exploration…As educators, we must push our students to reflect more deeply. Pushing students to continuously ask themselves why a decision was made or why they feel the way they do about a topic or situation will cause them to look deeper for answers. [For example, why did they choose a particular application over another?] (Walker, 2006, p. 217)

It is important for instructors to experiment with their students and classes to determine which methods encourage reflection in students (Walker, 2006, p. 218).

However the instructor chooses to integrate journal writing into a course, unless the journals have an effect on the grades, students will put very little effort into their writing. Adding a grade to the journals puts value to them and establishes their importance. Although 10% to 20% of a grade has been reported in the literature, it is up to the individual instructor to weigh the journals accordingly or in some way to ensure that students feel the journal writing assignments matter and are of significant value (Walker, 2006, p. 219). Instructors' comments will be important in this process, as it shows their commitment to the students' efforts. Comments such as the ones that follow, should lead the students into even further reflection, should they make the extra time to reflect on them: how did you form this opinion? Where did you learn this information? How did you know this was the right action? How will you handle this in the future? (Walker, 2006, p. 220). Walker (2006) indicates that, as stated by Kobert:

> One drawback to journal writing is what makes it so valuable: students may be reluctant or unable to explore and share intimacies of their own lived experiences with others. They may be more concerned with writing what they think the instructor wants to hear than writing about what is true to them. Writing about issues and feelings puts the student in a very vulnerable position. To promote reflection, he or she must express weakness and insecurity to grow. Students must feel comfortable exposing this vulnerability…[There is] significant responsibility of both the student and instructor to accept differing views while searching for understanding and meaning. Part of encouraging this truthful writing is not only through feedback procedures but also by maintaining confidentiality to encourage truthfulness. If students:
>
> i) are familiar with the instructor and know him/her to be nonjudgmental, they will, more than likely, be more willing to self-disclose in their journal writing;
>
> ii) [are not familiar with the instructor,]they will need evidence that the instructor will remain true to his or her word before disclosing too much in a journal entry (p. 220).

Such trust takes time to develop, but if journal writing is seen as a work in progress, this is all part of the journey (Walker, 2006, p. 221). In the case of Comp268, the journal entries need not be of a personal nature; they might simply outline challenges faced and how they were overcome, as somewhat of a reference. When the student becomes more comfortable, they can discuss the emotion connected to their experience as well if desired.

APPENDIX C – Assessment

Harrington and Reasons (2005) indicate that: "Distance education as a vehicle for the delivery of course content, as well as complete academic programs, continues to grow rapidly" (p. 2).

"There is no shortage of research on the subject of student evaluation of teaching (SET). Just searching the ERIC database for 'student evaluation of teaching performance' reveals [almost 7,000] citations…Not surprisingly, colleges and universities differ in their approaches to the student evaluation of teaching in distance education courses" (Harrington et al. 2005, p. 3): some have the instructors do it, some have the department heads do it, and some don't have a mandatory requirement for it. This would definitely raise some fundamental questions such as:

> How are institutions measuring and ensuring the quality of their distance education curricula? How thoroughly is teaching in distance education courses being evaluated? Are standardized evaluations of teaching forms being used? Who is responsible for developing the forms, and have the forms been evaluated for statistical reliability and validity? Who oversees survey and instrument administration, data collection processes and dissemination of results? (Harrington et al. 2005, p. 2).

[It is important] to develop a consistent, effective, and appropriate means for student evaluation of teaching [and learning] in distance education courses, which recognizes the unique necessity of combining both formative and summative measures…Historically, there are usually only end-of-term course evaluations that are very similar to the paper-and-pencil method in traditional classrooms. (Harrington et al. 2005, p. 4).

They also provide information regarding a pilot project conducted in the Spring of 2003 for a mid-western university in the USA. This project was commissioned with a view to discuss a series of fundamental questions related to course evaluation and assessment: "Why are we doing this in the first place? What do we need to measure? What would we like to measure? What do the results need to look like? How will they be used?" (p. 8). The following are lists of the "project logistics" that they needed to explore for both the formative and the summative measures:

Formative Issues:

- *I found course objectives and assignments to be clearly stated and easily understood.*
- *I felt connected to the instructor and other students in this course.*
- *The instructor used a number of teaching techniques to involve me in learning.*
- *The way in which this course was taught required me to think in new and different ways.*
- *The instructor was able to clearly explain the relationships among the various course topics.*
- *The instructor makes difficult course material understandable.*
- *Parts of this course were designed to make difficult material thoroughly understandable.*

Summative Items:

- *In this course, the instructor's teaching required me to do my best work.*
- *In this course, the instructor was able to explain difficult materials in ways that I can understand.*
- *In this course, the assignments given in class were challenging.*
- *This instructor is one of the best I've had at this University.*
- *Of all the courses I've taken at this university, this course is one of the best I've had.* (p. 8) (italics added)

APPENDIX D. Directives for Fostering Effective Group Collaboration Environments

Commented [CI31]: Good idea to put this in an appendix. It supports your proposal, but isn't directly enough related to it to include in the main text.

Blaschke, Brindley, and Walti (2009) suggested that students expressed dissatisfaction with group work when they felt they lacked a sense of full control over the quality and grade assigned. The authors suggest that grading should not be emphasized, and to instead allow the students to learn how to deal effectively with a group mate who might not be performing. The authors provided the 10 following directives for fostering effective group collaboration environments.

1. Transparency of expectations – The importance of collaboration will be clearly posted in the syllabus and mapped to the learning objectives of preparing the student to work in teams in industry.
2. Clear instructions – Tasks and timelines including length of time estimated to complete each task will set out by the instructor. Specific tasks (writing, testing, optimization, compatibility) will work on a rotation through the group. For the final group project, there will be choices from which the students can determine their final project. The choices will be accompanied by a complete description of the usability of the final product in order to alleviate the load of trying to clarify the task and develop a common group understanding of the project as they approach it.
3. Appropriateness of task for group work – the task of programming is already deemed suitable for group work, since this is the way that teams function in the work world.
4. Meaning-making/relevance – The concepts learned in each module are understood and applied to programming problems, thereby providing meaning and purpose.-

5. Motivation for participation embedded in course design – the groups must work together to complete the project, so the motivation is to produce well written and tested code.

6. Readiness of learners for group work – By beginning the group work at the start of the course, and accomplishing simple tasks, the group has the opportunity to develop a working relationship and style as they approach each new module.

7. Timing of group formation – Group work begins immediately, and progresses in complexity over the course to allow time. Each group project will have a shared reflective component, where students will draw on their own personal journals and share their impressions and learning with one another in a group setting.

8. Respect for the autonomy of learners –Students are free and encouraged to participate in the larger group forum and poll the class through the Me2U Java Exchange Students group. This allows them to go outside their immediate group to get feedback or support.

9. Monitoring and feedback – The Instructor and tutors will closely monitor the groups and provide direction, resources where needed.

10. Sufficient time for the task – Simple module assignments will be allotted sufficient time to accommodate differing schedules and commitments. Tasks will be assigned and students will rotate on each task (eg. writing, testing, compatibility checks, etc).

The group work is built in a manner that can take advantage of the scaffolding of knowledge and experience, as the students work through more modules which will manifest in the acquisition of skills.