Outcome-based Education Model for Computer Science Education

Srividya Bansal, Ajay Bansal
Arizona State University
Mesa, AZ 85212, USA
{srividya.bansal, ajay.bansal}@asu.edu

Odesma Dalrymple
University of San Diego
San Diego, CA 92110, USA
odesma@sandiego.edu

***Abstract*—A well-designed and constructed course plan or curriculum is an integral part of the foundation of effective STEM instruction. This paper presents a framework for outcome-based course design that can be translated into a semantic web-based tool which guides STEM educators through the complex task of curriculum design, ensures tight alignment between various components of a course (i.e., learning objectives, content, assessments, and pedagogy), and provides relevant information about research-based pedagogical and assessment strategies. This framework has been applied to the design of "Object-Oriented Software Development" course and the results are presented.**

***Keywords*—Software Engineering Education; Instruction design; Outcome-based Education.**

I. INTRODUCTION

The Instructional Module Development System (IMODS) is open-source web-based course design software under development that will present a framework for representing curriculum, particularly in the areas of Science, Technology, Engineering, and Mathematics (STEM), and scaffold users through the process of curriculum development. Outcome-based education (OBE) was used as the principal guide for the development of the IMODS framework. The IMODS will (i) guide individual or collaborating users, step-by-step, through an OBE process as they define learning objectives, select content to be covered, develop an instruction and assessment plan, and define the learning environment and context for their courses; (ii) contain a repository of current best pedagogical and assessment practices, and based on selections the user makes when defining the learning objectives of the course, the system will present options for assessment and instruction that align with the type/level of student learning desired; (iii) generate documentation of a course design; (iv) provide just-in-time help to the user; (v) provide feedback to the user on the fidelity of the course design. This paper presents a background on outcome-based education and the motivation for IMODS in section 2. The theoretical framework for IMODS is presented in section 3. Section 4 presents case study of an introductory programming course in B.S. in Software Engineering program and the IMODS framework applied to the design of this course. Results of the course design are presented in section 5 followed by conclusions and future work.

II. BACKGROUND

OBE is a result-oriented approach where the product defines the process. The learning outcomes guide what is taught and assessed [1], [2]. This approach contrasts the preceding "input-based" model that places emphasis on what is included in the curriculum as opposed to the result of instruction. OBE gained great traction at the K-12 level and was adopted by a number of school districts and state systems (including Kentucky, Michigan, Minnesota, Missouri, Pennsylvania and Washington).

The model provides a win-for-all solution; not only has it lead to student success, shown through higher achievement test scores, and improved attendance and motivation [3]; but it provides educators with an empirically driven structure for tracking impact, and identifying problems. There is a growing demand and interest in faculty professional development in areas such as OBE, curriculum design, and pedagogical and assessment strategies. In response to this demand a number of universities have established teaching and learning centers to provide institution-wide, and sometimes program specific support. The IMODS would support these ventures and broaden the impact and reach of professional development in the scholarship of teaching and learning, particularly to STEM faculty. While there are a number of options available to faculty for receiving instructional development training (i.e., training focused on improving teaching and learning), most share similar format, features, and shortcomings. The IMODS will facilitate self-paced instructional development training while the user creates his/her course design with the added benefits of being free to all who are interested, accessible almost anywhere through a web browser, and at any time that is convenient.

III. THEORETICAL FRAMEWORK

Many of the leaders in faculty development programs have identified facilitation by experts as a key ingredient in increasing the effectiveness of instructional development programs [4]. For the IMODS, which will provide professional development with the use of an online tool, expert facilitation is embedded within its design, through the application of a framework that is informed by research in the area of instructional development for STEM disciplines. This framework translates the scholarship into a software platform that supports the development of a rich, meaningful knowledge structure that can be queried to: (1) identify omissions in a course design; (2) identify inconsistencies in the relationships between the elements of the course being designed; (3) identify relevant strategies for instruction and/or assessments; (4) provide just-in-time

guidance to the user on the design process. Development of this software platform is included within the goals of the IMODS project. The framework that will eventually support this development, however, has been conceptualized and a case study to test its feasibility for design of an introductory object-oriented programming course is presented. The structure of that framework and its implementation for design of a course are discussed in the subsequent sections.

A. Previous Models of Outcome-Based Course Design

OBE is an approach where the product defines the process, i.e., the outcomes that specify what students should be able to demonstrate upon leaving the system are defined first, and drive decisions about the content and how it is organized, the educational strategies, the teaching methods, the assessment procedures and the educational environment [1], [5]. This is a contrast to the preceding "input-based" model that placed emphasis on the means as opposed to the end of instruction. OBE was chosen for the following reasons:

1) win-for-all solution – OBE is shown to improve student success, provides a structure to educators for designing instruction, and facilitates reporting to external stakeholders in an accountability education climate;
2) it supports the How People Learn framework for designing learning environments [6];
3) growing adoption of outcome-based program accreditation – Accreditation boards such as ABET, have moved to an outcome focused model (what students learned) to assess the quality of programs in Applied Science, Computing, Engg., and Engg. Technology;
4) alignment with other models that are meant to increase innovation in STEM education – OBE dictates the end and not the means thereby allowing innovation in instruction. It also provides an empirical structure to track impact and identify shortcomings.

A number of models have been developed to represent the application of OBE in design of effective courses. Four key models widely discussed in engineering education are:

1) Effective Course Model by Felder & Brent [4]
2) Integrated Course Design by Fink [7]
3) Understanding by Design Model by Wiggins and McTighe [8]
4) Content Assessment Pedagogy Model [9]

Figure 1 & 2 show a visual representation of some of these models. All of these models either directly or indirectly identify four main elements that must be tightly aligned when defining a course design, i.e., course objectives, content, assessments, and pedagogy. Therefore, one of the main challenges in adhering to an outcome-based approach is maintaining the alignment between course elements. Inconsistencies in the interrelation of these elements can lead to the overall incoherence of the course.

One approach for achieving alignment among course element is through a "backward-looking" design process where the desired results are identified first, and then assessments are designed to verify that these results have been achieved. The learning experiences and instruction are then formulated around the desired results and the assessments. The use of this approach forms the basis of the Understanding by Design model, and is also applied by the other models. One of the key functions the IMODS is expected to perform is the evaluation of the fidelity of the course design. To achieve this, the IMODS framework must include machine processable constructs that can be used to make inferences on the inconsistencies in the relationships between the elements of the course being designed. While the backward-looking process dictates an ideal sequencing of tasks, it is limited in its ability to support automated inferencing on course element coherence. The IMODS framework, therefore, expands on the current models with the inclusion of new constructs.

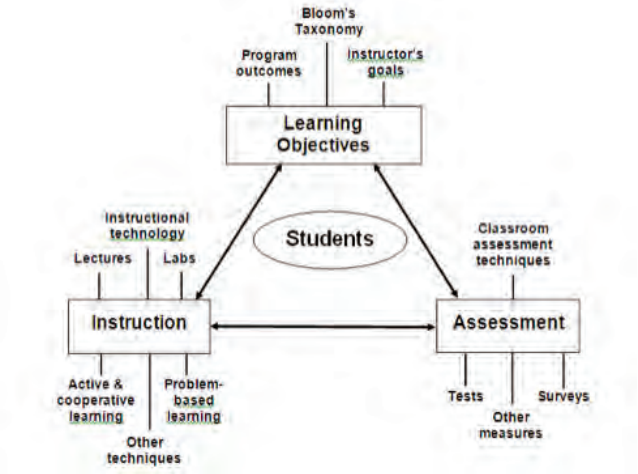

Figure 1: Effective Course Model by Felder and Brent

Figure 2: Integrated Course Design by Fink

B. IMODS Framework

The IMODS framework adheres strongly to the OBE approach and treats the course objective as the spine of the structure. New constructs (not included in the models previously discussed) are incorporated to add further definition to the objective. The work of Robert Mager [10] informs the IMODS definition of the objective. Mager identifies three defining characteristics of a learning objective: Performance – description of what the learner is

expected to to do; Conditions – description of conditions under which the performance is expected to occur; and Criterion – a description of the level of competence that must be reached or surpassed. For use in the IMODS framework an additional characteristic was included, i.e., Content – description of the disciplinary knowledge, skill, or behavior to be attained. Resulting IMODS definition of an objective is referred to as PC^3 model.

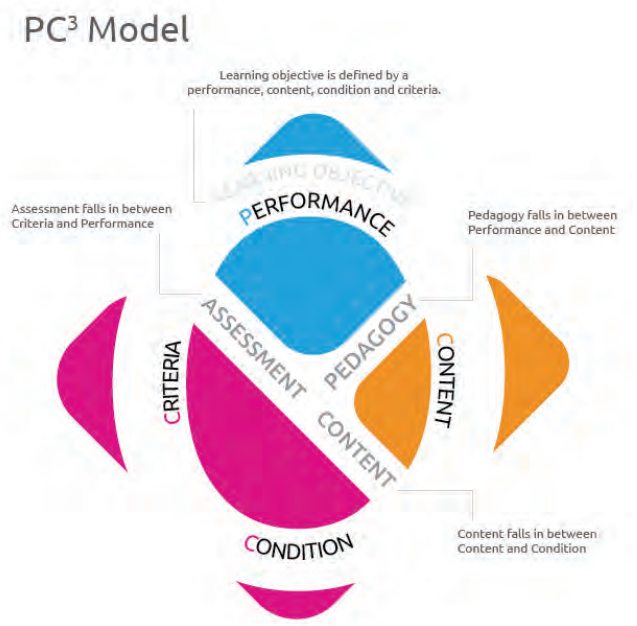

Figure 3: IMODS Framework - PC^3 Model

The other course design elements (i.e., Content, Pedagogy, and Assessment) are incorporated into the IMODS framework through interactions with two of the PC^3 characteristics. Course-Content is linked to the content and condition components of the objective. The condition component is often stated in terms of pre-cursor disciplinary knowledge, skills or behaviors. This information, together with the content defined in the objective, can be used to generate or validate the list of course topics. Course-Pedagogy is linked to the performance and content components of the objective. The types of instructional approaches or learning activities used in a course should correspond to the level of learning expected and the disciplinary knowledge, skills or behaviors to be learned. The content and performance can be used to validate pedagogical choices. Course-Assessment is linked to the performance and criteria components of the objective. This affiliation can be used to test the suitability of the assessment strategies since an effective assessment, at the very least, must be able to determine whether the learner's performance constitutes competency. Figure 3 shows a visual representation of the IMODS framework that emphasizes the alignment of course design elements through the PC^3 model of an objective and the placed on the course design by variables defined in the learning context.

Learning domains and domain categories defined by Bloom's revised taxonomy [11] are used to describe learner performance. Learning domains are categorized into Cognitive, Affective, and Psychomotor, which are further classified under various Domain Categories (Remember, Understand, Apply, Analyze, Evaluate, Create). Each Domain Category has performance verbs associated to it. Learning objective in the PC^3 model is described in terms of Performance, Content, Condition, and Criteria. Performance is described using an appropriate action verb from revised Bloom's taxonomy based on the learning domain and domain category.

Criteria: Learning objective assessment criteria are categorized as quality, quantity, speed, and accuracy. Criteria for learning objectives are described in terms of one or more of these categories with a criteria value defined or determined later when the assessment is defined.

Knowledge Dimensions*:* The revised Bloom's taxonomy introduced an additional dimension called the knowledge dimension that was categorized as Factual, Conceptual, Procedural and Metacognitive.

- *Factual* Knowledge is knowledge that is basic to specific disciplines. This dimension refers to essential facts, terminology, details or elements students must know or be familiar with in order to understand a discipline or solve a problem in it.
- *Conceptual* Knowledge is knowledge of classifications, principles, generalizations, theories, models, or structures pertinent to a particular disciplinary area.
- *Procedural* Knowledge refers to information or knowledge that helps students to do something specific to a discipline, subject, or area of study. It also refers to methods of inquiry, very specific or finite skills, algorithms, techniques, and particular methodologies.
- *Metacognitive* Knowledge is the awareness of one's own cognition and particular cognitive processes. It is strategic or reflective knowledge about how to go about solving problems, cognitive tasks, to include contextual and conditional knowledge and knowledge of self.

Topic Prioritization: The IMODS framework uses a prioritization framework that classifies topics and subtopics of a particular course as one of the following:

- Critical
- Important
- Good to know

All content topics that the instructor feels a student should, at the bare minimum have a basic understanding of are categorized as *good to know*. All those content topics that a student in the course should have a broader understanding of are categorized as *important*. The core ideas of the course are included under *critical*. Content topics that form the base of the course fall under this category and are central to the understanding and application of the subject being taught. Achieving the right mix of the three levels of learning (priorities) is essential to planning a good course. If there are a high percentage of content topics classified

under *Critical*, it's usually safe to assume that the instructor is aiming for too much. On the other hand, if a large percentage of topics were classified under *Good to know*, students may not be able to achieve the desired level of learning. The decision of topic/sub-topic prioritization is completely dependent on the instructors' judgment. It is useful to have a graphical tool to help analyze the priority assignments and change them if required.

IV. CASE STUDY

The IMODS framework was applied to design an introductory programming course titled "Object-Oriented Software Development" in B.S. in Software Engineering program. This section describes the use of IMODS – PC^3 model for course design.

A. About the Course

Objective-Oriented Software Development is a freshman course in the Software Engineering program that introduces problem solving with a state-of-the-art programming language. Expressions, statements, basic control flow, and methods are the broad topics introduced to students. Students are also exposed to data, data aggregation, and usage. This course uses a structured personal software development process to implement solutions representative of common computing applications. Development kits are used for some of the course activities. Basic concepts of object-oriented analysis, design, and programming using Python are covered. The students in the class study basic Python variables, expressions, arrays, statements, loops, functions, methods, and classes. Game development using a Python development kit called Pygame was introduced. A project-based pedagogical model is used for delivery of all our courses in Software Engineering program. Students in this course worked on a game project using Pygame.

B. Learning Objectives

Learning objectives of this course were defined using the PC^3 model. The course has 6 objectives that are categorized under Performance, Content, Condition, and Criteria as shown in the table 1. The objectives are as follows:

LO1: Apply the concepts of sequence, selection, and iteration by constructing algorithms and formal code for problem solving
LO2: Use variables and composite data structures to store and manipulate data by constructing algorithms and formal code to solve problems
LO3: Use modular programming techniques such as functions and objects by constructing algorithms and formal code to solve problems
LO4: Understand concepts of objects and types
LO5: Apply a disciplined problem solving process to the construction of algorithms and formal code to solve problems
LO6: Configure a software development environment for the construction of formal code to solve problems

Table 1: Learning Objectives based on PC^3 Model

Objective	Learning Domain	Domain Category	Action Category	ActionWord Performance	Content	Criteria	Condition
LO1	Cognitive	Apply	Implement	Apply	sequence, selection, iteration, constructing algorithms, constructing formal code, problem solving	Accuracy (85%)*	Given a problem specification
LO2	Cognitive	Apply	Implement	Use	variables, composite data structures, store and manipulate data:, constructing algorithms, constructing formal code	Accuracy (85%); Speed (DPA** summative)*	Given a problem specification
LO3	Cognitive	Apply	Implement	Use	modular programming techniques, functions, objects, constructing algorithms, constructing formal code	Quality (DPA**); Speed (DPA** summative)	Given a problem specification
LO4	Cognitive	Understand	Interpret	Understand	objects, types	Accuracy (DPA**)	-
LO5	Cognitive	Apply	Implement	Apply	problem solving process, constructing algorithms, constructing formal code	Accuracy (85%); Quality (DPA**)	Given a problem specification
LO6	Cognitive	Apply	Implement	Configure	software development environment, constructing formal code	Accuracy (85%); Speed (DPA**)	Given a problem specification

* Collecting data on application of criteria to various types of assessments, e.g., formative and summative
** DPA Determined Per Assessment

C. Content

The list of Content topics and subtopics are listed in table 2. For each topic the knowledge dimension and topic priority is defined. This information is used to find assessments and instructional activities that best fit for delivering a topic.

D. Assessments

Assessments chosen for this course include a mix of both formative and summative assessments. The PC^3 model aligns assessments chosen for the course with the learning objectives by checking compatibility of learning domains, performance, and criteria requirements. Table 3 provides the list of assessments with their corresponding learning domain category, knowledge dimension, and criteria type that each method is suitable for.

E. Instructional Activities

Pedagogical activities used in this course are listed in table 4 along with the knowledge dimension and learning domain category that they are suitable for. The list of activities includes a mix of lectures, lab activities, Q&A discussions, and problem solving activities.

Table 2: Content Topics based on PC^3 Model

Content Topic	Content Sub-Topics	Knowledge Dimension	Priority
Programming Fundamentals	Problem Solving	Procedural (P), Metacognitive (M)	Critical (3)
	Constructing Algorithms	Procedural (P), Metacognitive (M)	Critical (3)
	Constructing formal code	Factual (F), Procedural (P)	Critical (3)
	Variables	Factual (F), Conceptual (C)	Critical (3)
Software Development Environment	Setup and Configuration	Procedural (P)	Important (2)
	Writing code	Factual (F), Procedural (P)	Critical (3)
	Executing Code	Procedural (P)	Critical (3)
	Errors (syntax, semantic, runtime)	Conceptual (C), Procedural (P)	Critical (3)
Program Structure and Flow	Sequence	Factual (F), Conceptual (C)	Critical (3)
	Selection	Factual (F), Conceptual (C)	Critical (3)
	Iteration	Factual (F), Conceptual (C)	Critical (3)
	Functions	Factual (F), Conceptual (C)	Critical (3)
	Modular Programming techniques	Conceptual (C), Procedural (P)	Important (2)
Composite Data Structures	Store and manipulate data	Factual (F), Conceptual (C)	Important (2)
	Lists	Factual (F), Conceptual (C)	Important (2)
	Strings	Factual (F), Conceptual (C)	Important (2)
	Tuples	Factual (F), Conceptual (C)	Important (2)
	Searching and Sorting	Conceptual (C), Procedural (P)	Good to know (1)
Classes and Objects	Objects, Types	Conceptual (C)	Good to know (1)
Application Domain – Problem Specification	Introduction to graphics and drawing	Factual (F), Conceptual (C)	Important (2)
	Introduction to animation	Factual (F), Conceptual (C)	Important (2)

Table 3: Course Assessments

Assessment	Type	Domain Category	Knowledge Dimension	Criteria
Programming exercise (write formal code)	Formative	Understand, Apply, Analyze, Evaluate, Create	Conceptual, Procedural	Speed, Quality, Accuracy
Partially guided programming exercise	Formative	Understand, Apply, Analyze, Evaluate	Conceptual, Procedural	Quality, Accuracy
Guided Lab exercise	Formative	Understand, Apply, Analyze, Evaluate	Conceptual, Procedural	Quality, Accuracy
Quiz	Formative	Remember, Understand	Conceptual, Factual	Accuracy, Speed
Project	Summative	Understand, Apply, Analyze, Evaluate, Create	Conceptual, Procedural	Quality, Accuracy
Exam	Summative	Understand, Apply, Analyze, Evaluate	Conceptual, Procedural	Quality, Accuracy

Table 4: Course Pedagogical activities

Instructional Activities	Domain Category	Knowledge Dimension
Lectures (Face-to-Face, Audio, or Video)	Remember, Understand	Conceptual, Factual
Problem Solving	Understand, Apply, Analyze, Evaluate	Conceptual, Procedural
Partially Guided Programming exercise	Understand, Apply, Analyze, Evaluate	Conceptual, Procedural
Active Reading	Remember, Understand, Apply	Conceptual, Factual, Procedural
Programming Assignment	Understand, Apply, Analyze, Evaluate, Create	Conceptual, Procedural
Guided Lab Exercise	Understand, Apply, Analyze, Evaluate	Conceptual, Procedural
Q&A Forum	Remember, Understand	Factual, Conceptual

V. RESULTS

Object-Oriented Software Development course in the Software Engineering program in School of Computing, Informatics, Decision Systems Engineering (CIDSE) at Arizona State University was designed using the IMODS – PC^3 model framework and offered as a face-to-face section as well as an online section by the same instructor (one of the co-authors). Using the IMODS framework ensured the alignment between various course elements and thereby ensuring high-quality course design.

***Alignment between Learning Objective and Assessments*:** The framework supports the checking of alignment between course assessments and learning objectives. The course assessments are linked to the performance and criteria elements of the learning objective. Figure 4 shows this alignment for learning objective – LO1. The icons used in this figure are explained through a legend shown in Figure 5.

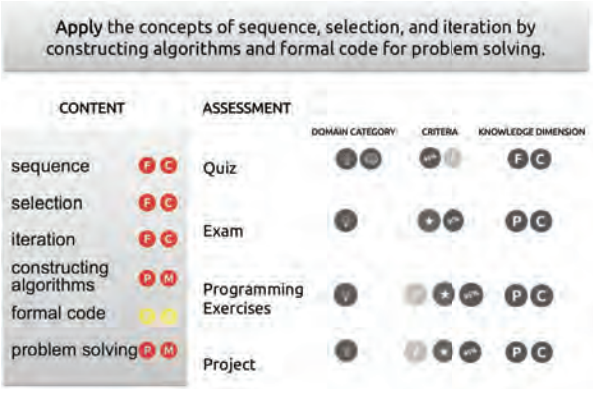

Figure 4: Alignment of LO1 and Assessments

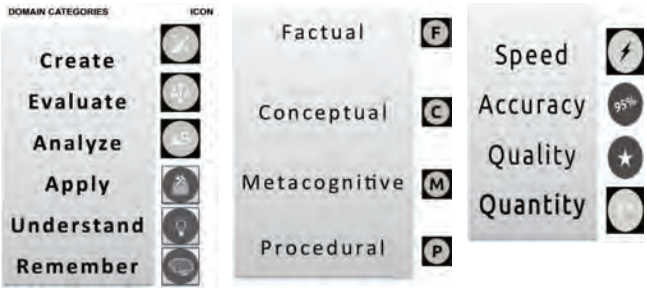

Figure 5: Legend for icons used for Performance, Knowledge Dimension, and Criteria

***Alignment between Learning Objective and Pedagogy*:** The framework supports the checking of alignment between course instructional activities and learning objectives. The course pedagogical activities are linked to the performance and content of the learning objective. Figure 6 shows this alignment for learning objective – LO1. The colors used in this figure for content prioritization are explained through a legend shown in Figure 7.

Syllabus Generation: After the course was designed using the IMODS framework it was fairly easy to generate a syllabus with clear set of learning objectives to be conveyed to the students. The course design also helped with creating a Faculty Course Assessment report (FCAR) for continuous improvement and ABET accreditation documentation.

***Student Evaluations of the course*:** Student evaluations for multiple offerings of this course in recent semesters is shown in table 5. The table also shows the number of students enrolled in each of those sections. The ratings were shown to be high and way above the departmental average.

***Student Feedback*:** Positive feedback was received from the students about the course design. A few comments from the students are provided below:

- *Course itself was also laid out better than any of my other classes, in my opinion. Very simply, it was the most organized and streamlined.*

- *Good learning experience for introductory programming.*

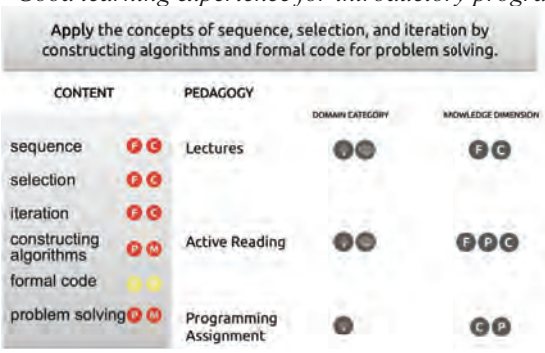

Figure 6: Alignment of Instructional Activities and LO1

***Topic Prioritization*:**

Use of the PC^3 model ensured a balanced distribution of the topics under Critical, Important, and Good to know as shown in figure 8.

Figure 7: Legend for Topic Prioritization

Table 5: Student Evaluations

Semester Course Offered	Summer 2014	Fall 2013	Summer 2013
Delivery Mode	Face-to-face	Online	Face-to-face
Number of students	19	105	19
Overall Course Evaluation	5.0	3.8	4.7
Approach stimulated student thinking	5.0	4.1	4.9
Instructor related course material to its applications	4.89	4.4	4.5
Instructor's methods of presentation supported student learning. 4	4.78	4.4	4.7

Figure 8: Priority Distribution of Content Topics

VI. Future Work

The development of a semantic web-based tool based on the IMODS framework is under progress. The user research phase of the project is completed and functional prototyping is underway. We will conduct usability testing of the prototype using user interviews and other usability testing methods. Semantic web technologies facilitate: the organization of knowledge into conceptual spaces, based on their meanings; extraction of new knowledge via querying; and maintenance of knowledge by checking for inconsistencies. These technologies can therefore support the construction of an advanced knowledge management system [12], [13]. The IMODS software system will use Semantic web technologies to provide intelligent interactions with the users, dictate a course design process in conformance with the underlying framework, check for omissions and inconsistencies in the design, provide feedback to the user on their course design, and recommend relevant assessment and pedagogical approaches along with help on how they are implemented. The IMODS framework will be translated into a rich meaningful knowledge structure in the form of an ontology, i.e., an explicit and formal specification of a conceptualization [14]. During the course design elicitation process, logical inference algorithms will test the course design for consistency and adherence with the ontological model. The software prototype will be used to design further courses in our Software Engineering program including courses on Personal Software Process and Software Testing and Quality and data will be collected to assess the improvement in course design. These courses are delivered both as face-to-face and online courses.

Acknowledgment

The authors gratefully acknowledge the support for this project under the National Science Foundation's Transforming Undergraduate Education in Science, Technology, Engineering and Mathematics (TUES) program.

References

[1] R. M. Harden, J. R. Crosby, and M. H. Davis, "AMEE Guide No. 14. Outcome-Based Education: Part 1--An Introduction to Outcome-Based Education," *Medical Teacher*, vol. 21, no. 1, pp. 7–14, 1999.

[2] W. G. Spady and K. J. Marshall, "Beyond Traditional Outcome-Based Education," *Educational Leadership*, vol. 49, no. 2, pp. 67–72, 1991.

[3] G. C. Furman, "Outcome-Based Education and Accountability," *Education and Urban Society*, vol. 26, no. 4, pp. 417–437, 1994.

[4] R. M. Felder, R. Brent, and M. J. Prince, "Engineering Instructional Development: Programs, Best Practices, and Recommendations," *Journal of Engineering Education*, vol. 100, no. 1, pp. 89–122, Jan. 2011.

[5] W. G. Spady, "Organizing for Results: The Basis of Authentic Restructuring and Reform," *Educational Leadership*, vol. 46, no. 2, pp. 4–8, 1988.

[6] J. D. Bransford, A. L. Brown, and R. R. Cocking, *How people learn*. National Academy Press, DC, 2000.

[7] L. D. Fink, "Creating significant learning experiences: An integrated approach to designing college courses." San Francisco: Jossey-Bass, 2003.

[8] G. P. Wiggins and J. McTighe, *Understanding by design*. Assoc. for Supervision & Curriculum Development, 2005.

[9] R. A. Streveler, K. A. Smith, and M. Pilotte, "Aligning Course Content, Assessment, and Delivery: Creating a Context for Outcome-Based Education," *K. Mohd Yusof, etal (Eds.), Outcome-Based Education and Engineering Curriculum: Evaluation, Assessment and Accreditation. Hershey, Pennsylvania: IGI Global*, 2012.

[10] R. F. Mager, "Preparing Instructional Objectives: A critical tool in the development of effective instruction 3rd edition," *The Center for Effective Performance, Inc*, 1997.

[11] L. W. Anderson and D. R. Krathwol, "A taxonomy for learning, teaching and assessing: A revision of Bloom's Taxonomy of educational objectives." New York Longman, 2001.

[12] G. Antoniou and F. Van Harmelen, *A semantic web primer*. the MIT Press, 2004.

[13] N. Shadbolt, W. Hall, and T. Berners-Lee, "The semantic web revisited," *Intelligent Systems, IEEE*, vol. 21, no. 3, pp. 96–101, 2006.

[14] J. Davies, D. Fensel, and F. Van Harmelen, *Towards the semantic web*. Wiley Online Library, 2003.