International Journal of Engineering Education Vol. 27, No. 1, pp. 128–137, 2011
Printed in Great Britain

0949-149X/91 $3.00+0.00

A 3-D Pyramid/Prism Taxonomy for Viewing Knowledge When Teaching Language-Focused, Undergraduate Simulation Courses*

CHRIS POYNER, MARY COURT, HUONG PHAM, AND JENNIFER PITTMAN
School of Industrial Engineering, University of Oklahoma, 202 West Boyd, Suite 124, Norman, OK 73019.
E-mail: mcourt@ou.edu

We developed a 3-D knowledge pyramid/prism model to structure the relationships of lower-level learning, 'optional' knowledge bases, concurrent knowledge, and new knowledge; so we may view learning needs of a higher-level learning objective. Our paradigm stems from Bloom's taxonomy, but has the advantage of supporting 'just-in-time' and 'learn-by-doing' delivery, teaching and learning styles. We illustrate the paradigm through the BMMKP (3-D knowledge pyramid/prism model of the highest-level, batch-means-method learning objective for our language-focused, undergraduate course). The BMMKP reveals how highly dependent and fully integrated this learning is to calculus, probability, statistics, and queuing theory—regardless of the simulation modeling language chosen.. The BMMKP is then used to develop a set of lower-level learning objectives for the undergraduate course. The 3-D pyramid/prism approach should lend itself well as a communication tool for visualizing other simulation learning objectives.

Keywords: learning models, just-in-time learning, concurrent learning

1. Introduction

A simulation study involves the execution of approximately eight high-level iterative steps (as shown in Fig. 1). Ideally, simulation course content should be developed to cover all steps of the study; particularly if we expect our industrial engineering (IE) undergraduates to be capable of utilizing simulation as an analysis tool in practice—i.e. our programs will generate entry-level, well-versed 'practitioners'. There are currently 93 industrial engineering programs within the United States with the vast majority requiring a simulation modeling course for their undergraduate degree program However, most IE undergraduate programs have recently reduced their degree credit hours (perhaps as a means for recruiting students) and few offer more than one course in simulation. The standard IE undergraduate curriculum now has one semester of an introductory simulation course with a major learning objective of having the student learn discrete-event logic via a simulation language—as is the case at the University of Oklahoma's School of Industrial Engineering.

A conventional requirement for the student in these one-semester courses is to show that s/he can take supplied descriptions of systems of study and encode those descriptions into a simulation language of choice (almost always chosen by the instructor). Some may also require the students to understand the issues surrounding simulation input modeling and output analysis; e.g. have the student be able to employ the method of independent replications and perhaps the batch means method.

Mainstream introductory course textbooks for teaching simulation languages provide systems descriptions and problems sets, where the arrival processes and service mechanisms are entirely described in terms of their probability distributions, schedules, etc. The student is then left with the abstraction tasks (encoding the system description into simulation language); and performing the simulation study steps of verifying (the model works as encoded), and validating (the model/code accurately reflects the behavior of the pre-

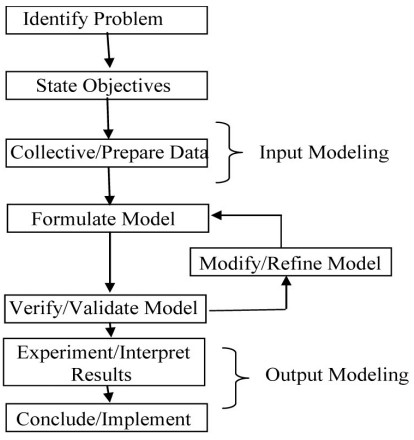

Fig. 1. Eight iterative steps of a simulation study.

* Accepted 10 September 2010.

described system) their model. Whether the textbook covers verification/validation techniques is questionable.

1.1 Provision for students

So, referring back to Fig. 1, problem identification, the objectives(s) of the study, and the input models (e.g. data distributions) are provided for the students; and the reiterative simulation study steps of formulating, verifying/validating, and modifying their simulation model(s) are for them to learn. This equates to the students being 'handed' nicely-worded problem definitions, well-behaving and complete data sets, and clearly identifiable/measurable performance parameters—a situation rarely found in practice! And, the last step of the study, implementation, is usually not encountered by students until they are able to utilize simulation in practice, or are allowed to implement the results of simulation study through an internship or capstone course.

But then again, most of the language-focused textbooks do 'progress' in terms of what is asked of the learner—e.g., identifying the problem(s) shift(s) from being supplied in the textbook's problem descriptions, to a task for the student to perform.

Some textbooks [1] progress even further and ask the student to perform some type of experiment on the model (see Fig. 1; Experiment/Interpret Results)—such as, obtain a confidence interval on a parameter of interest or perform what-if analysis on various system levels (e.g., the number of resources available or their scheduling schemas). One textbook [1] provides an excellent guide for performing the batch means method when using the Arena simulation modeling language and a thorough set of exercises requiring the student to do output analysis.

And yet, we continue to observe through e-mails, assignments, tests, etc. that one of the most difficult topics for our undergraduate IE students is output analysis—particularly output analysis for non-terminating systems. At issue is the student's ability to understand that output data generated from a non-terminating system's simulation will have both transient and steady-state data (i.e. the data are not iid—independent and identically distributed data). Additionally, they are uncomfortable or inexperienced with utilizing approximation tools (simulation) that rely on ad-hoc methodologies (e.g., graphical techniques to distinguish between transient and steady-state behavior) and statistical laws (e.g. the central limit theorem) for parameter estimation. Adding to the difficulty is that more often than not, the student has only had experience using mathematical modeling techniques that were 'guaranteed' to generate 'one-and-only-one' (or hopefully, the optimal) solution to a problem. Compounding their confusion is that if they do recall or master pre-requisite knowledge (e.g., what a confidence interval means), that knowledge is not readily applicable—the data violate an underlying assumption. Somehow, it must be made clear to the student that:

- simulation is a statistical experiment—an approximation tool—it will not automatically provide the optimal solution for you—it is not like prior modeling tools the you have utilized;
- simulation analysis is a statistical experiment and yet, the simulation data tend to violate the assumptions of classical statistical analysis techniques;
- there are only ad-hoc methodologies available for manipulating the data generated from non-terminating simulation models, so that parameter estimation may occur.

1.2 Checking simulation

Complicating the matter for the undergraduate IE student is the inability of students to 'check their answers'. Remember, these students are not comfortable with the amount of judgment/skill/experience required to evaluate their findings (e.g. confidence intervals about the parameters of interest). Now, let's take away their ability to check their results. One justification for using simulation is that the system is too complicated to be captured mathematically—so how are students able to judge the results of their simulation analysis? One approach is to draw upon their prior knowledge of queuing theory, so they may look at a more simplified system with closed-form solutions. The simplified system's steady-state parameters may provide some guidance. A simple example is—if they have just simulated an M/M/1 queue (single server system with exponentially distributed arrival and service times) but the server breaks down, they should expect that the average time-in-queue for their simulated model to be greater than the 'closely-related' M/M/1 queue (without breakdowns). Another approach is to remind them about the definition of a confidence interval and they should expect some degree of 'movement' about the parameter.

But we make it very clear that there are no guarantees in simulation output analysis—they cannot actually 'check' their results. The inability to know that they have the correct answer tends to 'pull the rug right out from under the student's feet'.

Initially, simulation output analysis (particularly simulation output analysis of non-terminating

systems) tends to be 'too ad-hoc' for the 'typical' undergraduate IE student. Simulation output analysis is viewed as a complicated, higher-level learning activity on the part of the student, since it requires them to draw upon several other 'older' knowledge bases (e.g. queuing and statistics). But does it require/draw-upon every topic in statistics, probability and queuing theory? If the answer is 'yes', then this may be why mainstream introductory course textbooks for teaching simulation languages either omit or do not provide much depth on the topic. But, does the student really need all of the topics, or can the course content or course textbook concentrate on only a few key concepts? And, how can that content and underlying knowledge be made more 'viewable' for the student?

1.3 Establishing learning objectives

One approach widely used by instructors to identify and help develop course content and assessment tools is to establish learning objectives. Learning objectives are active statements and involve some type of demonstrative/assessment 'product' (assignment or test), or activity (e.g. generate a graph) to show/prove the learning objective has been met. Educational research has shown faculty (instructors) who teach using learning objectives provide their students with learning advantages, since they communicate to the students what deliverables are expected of them. Students also obtain a 'view' of the underlying knowledge required for meeting the learning objectives.

The roots of learning objectives go back to Bloom [2]. However, in Bloom's hierarchical taxonomy, no higher-level learning can occur without lower-level learning being mastered. Previous work [3] has used Bloom's taxonomy in empirical investigations to determine course objectives. Additionally, [4] combined Bloom's with cooperative learning as a framework for filling the need for providing feedback within hierarchical levels of learning. But previous works [5], [6], [7], [8], and [9] with Bloom's taxonomy do not provide tools for modeling or visualizing complex learning (e.g. concurrent learning) or delivery systems (e.g. just-in-time teaching) and hence, the ability to develop learning objectives for complex learning paradigms such as those found in engineering curricula.

We now outline a derivative of Bloom's taxonomy, a 3-D knowledge pyramid/prism model that supports features not supported in Bloom's taxonomy: learning-by-doing, concurrent and just-in-time delivery, teaching and learning styles. We feel that our proposed view of knowledge is more correlated to the needs of model simulation knowledge; and more applicable for today's interdisciplinary curriculum and accelerated degree programs.

2. The knowledge pyramid model approach to viewing knowledge/learning

One of the most well-known outcomes in learning stemmed from the research conducted by a team of educational psychologists under the direction of Dr Benjamin Bloom. The team believed that learning could be separated into three domains: intellectual (cognitive) domain, emotional (affective) domain and physical (psychomotor) domain. The research is known today as 'Bloom's taxonomy of learning'. In [2], learning is best viewed as a hierarchical classification of learning objectives; where the student is expected to complete the lower level of learning before moving onto the next learning objective. The six learning objectives from the lowest to the highest level are:

1. Knowledge. The ability to recall the information presented.
2. Comprehension. The ability to restate the knowledge in different words.
3. Application. The ability to apply the knowledge appropriately to solve a problem.
4. Analysis. The ability to break a problem into its components and note the relationships between them.
5. Synthesis. The ability to rearrange component knowledge into a new 'whole'.
6. Evaluation. The ability to make decisions based on the whole situation.

There are some correlations between Bloom's taxonomy, simulation as an analysis tool, and the steps of a typical simulation study, as presented in our language-focused undergraduate simulation course: Bloom's Comprehension and Knowledge. There is a fundamental knowledge base required for students to learn simulation (e.g. probability, statistics, and queuing) and a new knowledge base of simulation for them to build and comprehend. Bloom's Application. A simulation study requires a certain set of simulation skills (new skills) and prior, 'older' skills (e.g. statistical analysis) on that of the student—e.g.

1. s/he must be able to represent the system (real or non-existent) via the appropriate amount of details
2. (abstraction and conceptualization),
3. s/he must also be able to select the appropriate mathematical and logical tools/algorithms,
4. s/he must be able to code the conceptual model within a particular simulation language.

Bloom's Analysis. Simulation is used for system analysis when you wish to study components, and

or their relationships. Simulation is a systems integration tool—allowing the parts (components) to be studied, as well as the whole.

Bloom's Synthesis. Modifying models and 'rearranging' systems and their components resulting in a new model is expected in simulation studies and is an integral step of a simulation study (see Fig. 1).

Bloom's Evaluation. Simulation allows you to study the system as a 'whole', as well as the system components. Additionally, an industry-wide expected deliverable is that the simulationist performs 'what-if' and output analysis for the purpose of comparing alternative (or competing) system designs, so that the 'best' solution can be identified and justified.

Note also that we take a 'just-in-time' delivery and a 'learn-by-doing' teaching approach for the language-focused undergraduate simulation course. That is, while we require the student to have some knowledge of the simulation modeling language and the underlying discrete-event logic;—we do not delay output analysis until the end of the course, but teach it in conjunction with the discrete-event logic and language topics. So while students are building a simulation-language knowledge (new knowledge) through the building of more and more complicated models, they are also required to learn output analysis techniques (parallel, new learning). The student simultaneously requires prior (old) knowledge (e.g. statistics and queuing theory). The just-in-time delivery and 'learn-by-doing' teaching and learning styles may be viewed as 'concurrent learning; where separate learning objectives (knowledge) are being achieved (built) in a synchronous or asynchronous manner. The learning we have just described is in violation of Bloom's taxonomy.

While the six levels of Bloom's taxonomy has been modeled in prior published research as successive levels within a 2-D pyramid's framework, we now propose (as shown in Fig. 2) a 4-level modification to the 2-D pyramid model due to:

- the correlation between simulation knowledge and Bloom's learning objectives;
- our need for viewing concurrent learning.

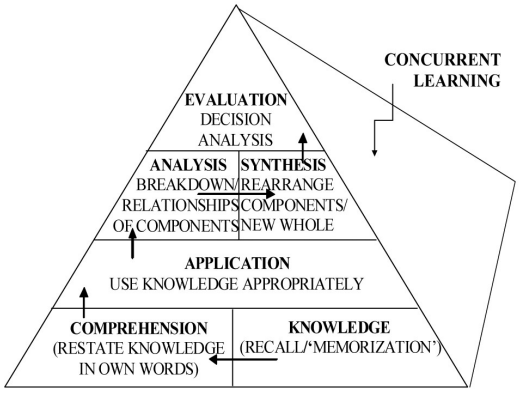

Fig. 2. Proposed 3-D knowledge pyramid/prism Model (KP/PM) based on Bloom's (1956) taxonomy.

Our 3-D Knowledge Pyramid/Prism Model (KP/PM) combines Bloom's Knowledge and Comprehension levels into one integrated level, and supports the viewing of concurrent, 'just-in-time' and 'learn-by-doing' teaching/delivery methodologies and learning styles.

Specifically, the lowest level of our KP/PM is knowledge recall (perhaps just memorization or the ability to locate the information in a textbook); and then (as indicated by the arrow), comprehension. We suggest that comprehension supports the capability of learners to remove themselves from the physical source of the knowledge base (e.g. having to refer to the textbook or having to search for the knowledge); and hence, improves their application skill set. But, we do not believe that lacking comprehension will necessarily deter the learner from successful application of the knowledge. That is, we have observed/noticed that students can be quite successful at memorizing even the applications of knowledge, without fully comprehending the knowledge they are utilizing. For example, if asked to calculate the average time-in-queue for an M/M/1 queuing system; students are quite capable of memorizing the formula, selecting the appropriate data to 'plug' into the formula, and solving for the unknown parameter—but, they are often not capable of 'stating in their own words' what the average time-in-queue means for a steady-state queuing system. So, in contrast to what Bloom supports, we do allow for learning to 'skip'—a student may not master comprehension but will (i) go from knowledge to application, or (ii) from application to knowledge (the 'learn-by-doing' style).

2.1 More physical than Bloom

If we view application in a more 'physical' sense, it is the identification, obtaining and manipulation of the correct data into the appropriately selected tool (correctly chosen formula). We believe the mastering of application learning cannot take place without some degree of knowledge (data) in place—similar to a formula without data—nothing can be calculated. So our definition of application knowledge in the KP/PM allows for a more physical interpretation than Bloom's; and we must therefore, allow for misconceptions at this level. That is, we allow for three misconceptions:

1. wrong knowledge/data in combination with the correct application/formula,
2. correctly chosen knowledge/data with the wrong application/formula,
3. wrong knowledge/data with wrong application/formula.

Note the arrows shown in the KP/PM are only used to indicate the hierarchical taxonomy Bloom professes; and when our paradigm is employed, arrows may or may not be present. Our paradigm is not as restrictive or hierarchical as Bloom's taxonomy. Our model supports learning within levels and between levels—in any direction. For example, if to solve a problem the learner is having difficulty with breaking down the problem into smaller more manageable components (analysis), the learner may need to increase their comprehension skills. In response to this need, the learner may 'self-test/evaluate' their comprehension via the application of new or old knowledge—or the instructor may require the learner to 'revisit' lower-level problem sets (application), and or the instructor may try to identify the knowledge gaps/misconceptions. The reverse is also true. Our paradigm allows an instructor to use application knowledge to support comprehension learning—i.e. our paradigm supports 'learning-by-doing' delivery of knowledge.

So, in our proposed model, the instructor and learner are free to draw from any level below—even skip levels—so as to meet learning objectives (or for the learner to reach higher levels of learning).

We also propose that our model is more supportive of viewing concurrent learning environments, co-enrolled (concurrent) course knowledge/learning/content, and even interdisciplinary degree programs. For example, due to the complexity of the new accelerated baccalaureate and masters degree programs, some of the courses that were once prerequisites are now being taken as co-requisite courses. The student is still expected to have that prerequisite knowledge; but it now becomes the responsibility of the student to build 'parallel'/'concurrent' knowledge bases for both courses. Under the accelerated program, it becomes highly likely that application learning requirements for one course come before the comprehension/knowledge level learning of the other course—i.e. the student needs to meet a higher-level learning objective without having the lower-level learning accomplished—again a violation of Bloom's taxonomy. This violation is less controllable since it may involve two courses and two instructors; across disciplines.

Our KP/PM has an added benefit of being able to support the visualization of concurrent learning within and between courses; particularly if the timing of concurrent learning is not synchronized.

We are not suggesting that our KP/PM guarantees knowledge gain, or that all learners will attain the appropriate amount of knowledge in this manner (there are always 'exceptions to the rule'); or even how to measure knowledge gain. We foresee the KP/PM may be used as a tool to assist the instructor in viewing the relationships between and among knowledge requirements for complex learning topics; and eventually, the development of specific learning objectives, assessment and misconception tools. The KP/PM is used in the next section for viewing the knowledge required for meeting the batch-means-method learning objective of our language-focused undergraduate simulation course.

3. A knowledge pyramid model of batch means method (BMMKP)

Before we reveal our batch-means-method knowledge pyramid (BMMKP), we present in Fig. 3 the 3-D course-based knowledge pyramid model (CBKP) of a student enrolling into our language-focused, undergraduate simulation course: IE4663, Systems Analysis Using Simulation. While the CBKP has only three levels (the KP/PM has four), a KP/PM could be developed for each level/topic. We only introduce the CBKP here to assist the reader in our development of the BMMKP.

The CBKP allows us to view the expected (required) and possible (elective or co-enrolled) undergraduate courses, as well as other knowledge pyramids a student may need, or might draw upon, when enrolled in IE4663. For example, while it is highly advisable for the student to have taken IE4553, Experimental Design, before IE4663; several of our undergraduate students are in co-

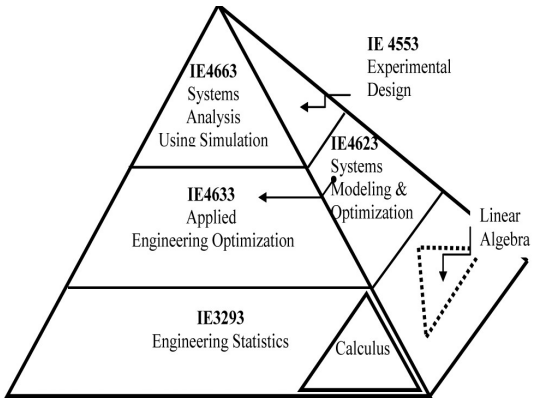

Fig. 3: 3-D, course-based knowledge pyramid model (cbkp) of undergraduate simulation course.

op or accelerated degree programs (BS/MS or BS/MBA). We allow those students to co-enroll in IE4663 and IE4553. IE4553 is then depicted at the same level as IE4663, but along the z-axis—i.e. the knowledge bases for the two courses may be concurrent.

The first level of the CBKP represents our calculus-based engineering statistics course, (IE3293, Engineering Statistics), other required courses for the IE undergraduate program, and the various types of other coursework/experience gained by an individual student. We also note that while linear algebra is no longer a requirement for our undergraduate program, our undergraduate students are advised to take the course. Since the majority does—we have this 'optional' (but helpful) linear algebra knowledge pyramid shown with dotted lines to indicate that it may or may not be present. The calculus knowledge pyramid is expected (required) for IE3293; so it is shown within IE3293 as a pyramid with solid lines to emphasize its importance (it could just as well be placed below IE3293).

The IE4633 course is the stochastic operations research course (Applied Engineering Optimization). It is in this second-semester, junior level course that the student is expected to build knowledge in Markov chain analysis and queuing theory; and then if time permits, have some experience with (exposure to) Monte Carlo simulation and discrete-event simulation logic.

IE4623, Systems Modeling and Optimization, is our deterministic operations course and required for IE4633 (as depicted by the arrow in Fig. 3). IE4623 is not directly correlated with our undergraduate simulation course (IE4663); but since it assists the students in building abstraction, conceptualization, and modeling experience, it is noted on our pyramid along the z-axis.

We now state the highest-level learning objective for the batch means method in our undergraduate IE4663 course:

At the end of the course, the student should be able to evaluate parameter estimates and parametric tests obtained via the Batch Means Method in order to identify and justify the 'best' alternative system among those competing.

We consider this learning objective to be at the evaluation level—our highest level of learning—i.e. the student will be tasked with having to demonstrate (either through their assignments, project, oral, and or written exams) that they are able to make and can justify their decision(s) based on the whole situation (a thorough systems analysis with appropriate 'what-if' exploration and supporting simulation output analysis).

Some of the tasks the student must perform in their demonstration can be immediately identified by listing the 'mechanical' steps involved with performing the batch-means method—a 'new' knowledge base. However, there are several other 'older' comprehension/analysis skills and tools the student must appropriately perform (or select) in order to achieve the goals of the learning objective (e.g. generate confidence intervals or perform hypothesis tests). The student must also be able to determine the viability of alternative solutions (competing designs—encoded into their new simulation modeling knowledge—and perhaps select the designs or design criteria themselves), by comparing the parameter(s) estimated (via the batch means method) against performance measure(s) established for the simulation study (e.g. identify the design that minimizes the average time-in-queue).

This higher-level learning objective is complicated since there are 'new' as well as 'old' knowledge bases (recall) the student must attain (know and comprehend) and utilize (apply).

At the application level, the student is expected to select and utilize the appropriate old (e.g. confidence intervals and correlation) and new tools (batch means method); and depending on the student, perhaps choose and employ concurrently-learned tools.

To demonstrate competency of this learning objective, requires the student to 'analyze', 'synthesize', and resolve the old, new and concurrent knowledge and applications; frame the analysis within a simulation-study context; and then, 'evaluate' the results and document a cohesive ('whole') argument supporting their recommendation(s).

As expected, our batch-means-method knowledge pyramid (BMMKP) for the undergraduate course is more complicated than our prior knowledge pyramid. The BMMKP of Fig. 4 has four 'sides' to represent expected and possible concur-

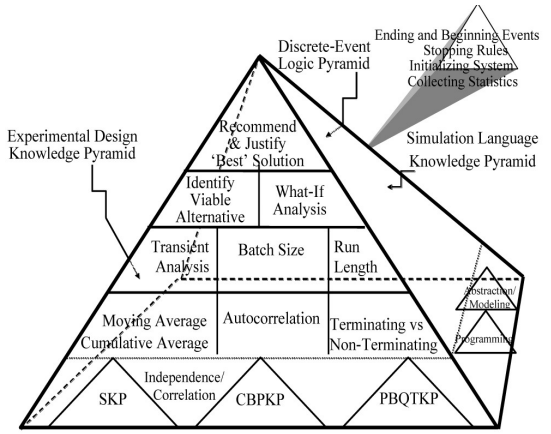

Fig. 4. Batch-means-method knowledge pyramid (BMMKP) for the language-focused undergraduate course, IE4663.

rent learning; and a foundation (comprehension/knowledge) level dependent on at least five other knowledge pyramids.

Concurrent learning is represented by the faces/sides of the BMMKP—i.e. the student may be co-enrolled in our IE4553, Experimental Design course. Plus, our delivery and instructional methodology employed for IE4663 has the knowledge bases for discrete-event logic and simulation language knowledge concurrent as well.

3.1. 'Learn-by-doing' and 'just-in-time'

Recall that for our course, we take a 'learn-by-doing' and 'just-in-time' approach to teaching, so we only require the student to have some knowledge of the simulation language and the underlying discrete-event logic topics; therefore, we do not detail the discrete event and simulation language knowledge pyramids here. We leave them for future research. We do, however, note a joint/merged knowledge pyramid generated from the discrete-event and simulation language topics, with the specific learning that is linked to the batch-means method (and other output analysis techniques as well). For illustrative purposes, a small protruding pyramid in Fig. 4 lists the lower-level knowledge for the intersecting sides of the discrete-event and simulation knowledge pyramids specific to our higher-level learning objective: ending and beginning events, stopping rules, initializing the system and (initial state), and collecting statistics.

The calculus-based probability knowledge pyramid (CBPKP, Fig. 5), the statistics knowledge pyramid, (SKP, Fig. 6) and the probability-based queuing theory knowledge pyramid (PBQTKP, Fig. 7) are specific to the BMMKP. The remaining two (abstraction/modeling and programming) knowledge pyramids are also not detailed here, since they will vary from student to student and are not necessarily required prior knowledge. In fact, the abstraction/modeling knowledge pyramid is expected to expand along the simulation language (face) knowledge pyramid (concurrent learning).

Observe in Figs 5–7 that again, we do not strictly follow our KP/PM. We found that most learning for the topics is foundational (knowledge recall and comprehension) and course assessment tools (assignments, quizzes, tests, etc.) focus on applica-

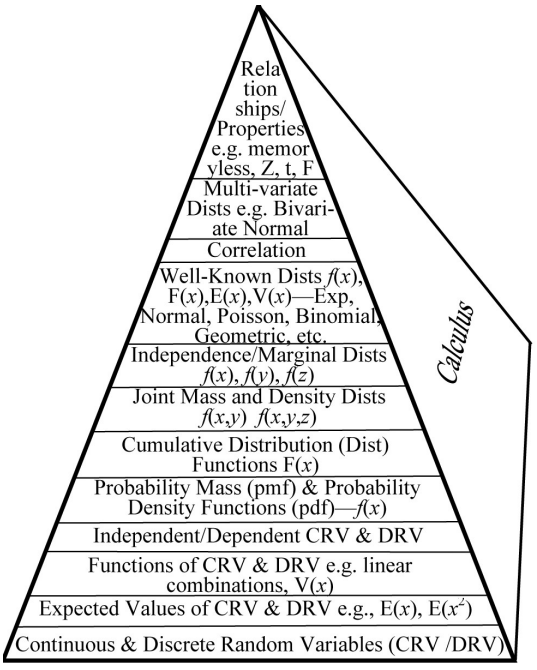

Fig. 5. 3-D Calculus-based, probability knowledge pyramid (CBPKP) required for learning the batch means method at the undergraduate level.

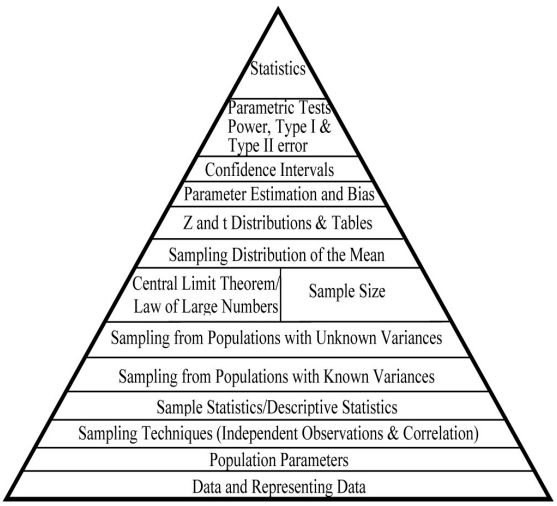

Fig. 6. Statistics knowledge pyramid (SKP) required for learning the batch means method at the undergraduate level.

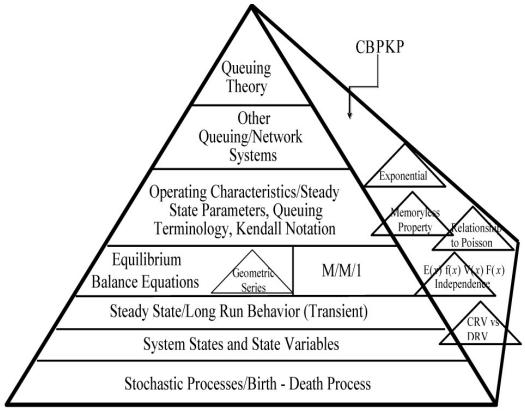

Fig. 7. 3-D Probability-based queuing theory knowledge pyramid (pbqtkp) required for learning the batch means method at the undergraduate level queuing knowledge pyramid.

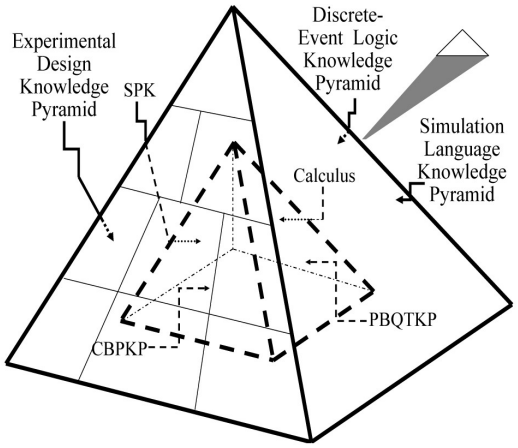

Fig. 8. Optional-BMMKP prism.

tions. As a result, we chose to only identify the knowledge most directly 'linked'/critical to the batch means methodology. Hence, the CBPKP, SKP and PBQTKP contain more of a 'suggested order' of learning/teaching topics. For example, in the SKP we do not see how anyone can truly 'comprehend' the sampling distribution of the mean without knowing the central limit theorem.

The CBPKP (Fig. 5) has a concurrent face, Calculus. But after reviewing the knowledge needed of statistics for our learning objective, we saw no justification for having calculus as a required knowledge pyramid in (or for) the SPK (Fig. 6). Calculus however, is required for knowledge gain in probability (although some students co-enroll). Likewise, probability knowledge is constantly called upon for queuing theory (PBQTKP, Fig. 7); and the particularly emphasized probability knowledge is shown as pyramids along the CBPKP face (e.g. the exponential distribution and its memoryless property).

Now, one can see that by having a 'common face', the PBQTKP (Fig. 7) and the CBPKP (Fig. 5) can be 'coupled', and then joined to the (SKP Fig. 6), to yield a four-sided pyramid—or prism. This allows the reconfiguring of the BMMKP (Fig. 4) into an optional-BMMKP, as presented in Fig. 8; with the prism of the foundational knowledge internal to the BMMKP. The details on the face of the optional-BMMKP and the small protruding knowledge pyramid are omitted for clarity since they remain the same as in Fig. 4.

For both the BMMKP (Fig. 4) and optional-BMMKP (Fig. 8), the levels supporting our batch-means-method learning objective are as follows:

- The Moving Average and Cumulative Average level is considered 'knowledge' that must be applied at the Transient (application) level for the student to show they can perform an ad-hoc transient analysis technique.
- The student will perhaps call upon 'older' graphing and spreadsheet-analysis knowledge to achieve the learning. For some students the calculation of these averages is 'new knowledge'.
- The comprehension of the graphs assists the student in determining steady-state (transient).

The Autocorrelation is also considered at the 'knowledge' level, since the student must be able to apply this knowledge at the Batch Size (application) level to demonstrate they can determine 'lag 0'. They will need to call upon prior knowledge in statistics (correlation) and perhaps other knowledge (e.g. graphing). They also need to comprehend that Autocorrelation knowledge will not necessarily 'guarantee success' (i.e. it is an ad-hoc methodology and some systems do not reach steady state).

Terminating versus Non-Terminating is at the knowledge and comprehension level since students must understand they are analyzing non-terminating systems, where the initial state and Run Length (application level) have impact on their parameter estimation. They also need to be able to identify and classify transient and non-transient systems based on the simulation study's objective(s). They will need to call on 'new' knowledge from the discrete event and simulation language knowledge pyramids; and 'old' knowledge from the PBQTKP and SKP.

The Transient Analysis, Batch Size, and Run Length are all at the application level. Students are applying their knowledge (using the tools) to obtain results. As with the knowledge level, the application level is also connected—i.e. if lag '0' cannot be determined, perhaps transient data still exist in the output data; or the Run Length was not long enough. Run Length coupled with Batch Size and Transient Analysis, will impact the number of batches generated, their independence and the 'strength' of the confidence statements.

At the Identify Viable Alternative level, students investigate/use parameter estimates from the batch means method (for those systems that do reach steady state) to determine feasible alternatives (e.g. they must answer the question, 'do the designs meet the simulation study's objective?'). They will call upon older knowledge (statistics), perhaps concurrent knowledge (designs of experiments); and new knowledge (simulation language pyramid) model.

The What-If Analysis level is where the student synthesizes and re-arranges the information of the Identify Viable Alternative level to determine the 'best solution'. The student may call upon 'older' knowledge (e.g. paired t-tests) and will need 'new' knowledge (e.g. simulation language knowledge pyramid).

At the highest level, the results are presented in a

cohesive argument with the 'best' solution identified. The student will also provide the statistical analysis and modeling techniques they employed to justify their recommendation.

The development of our BMMKPs led to the following lower-level learning objectives to support our higher-level learning objective (—all begin with the statement—'at the end of the course the student should be able to . . .':

- analyze a system and the objectives of the simulation study to identify the system as terminating or non-terminating;
- identify transient versus steady-state behavior using the moving average and cumulative average method;
- produce an autocorrelogram from output data and identify 'lag0';
- calculate the batch size when using the batch means method for steady-state parameter estimation;
- generate and test for approximately, normally distributed batches;
- apply the batch means method to obtain confidence intervals on the mean of system parameters (e.g. average queue time);
- identify and justify the 'best' of competing system designs in terms of designs of experiments or other parametric tests, where the data for the statistical tests are obtained via the batch means method.

4. Conclusions and future research

We have presented a 3-D knowledge pyramid/prism approach (the KP/PM) to viewing knowledge based on Bloom's taxonomy of learning. Bloom's taxonomy does not allow for today's classroom environment where a just-in-time or learn-by-doing approach to teaching and delivering content is popular; or for the increasing number of interdisciplinary and accelerated degree programs (concurrent and asynchronous learning).

Our KP/PM has an added benefit of being able to support the visualization of concurrent learning within and between courses; particularly if the timing of concurrent learning is not synchronized. We developed the model for viewing the relationships of

1. lower-level learning;
2. 'optional' knowledge bases;
3. concurrent knowledge;
4. new knowledge; in terms of a higher-level learning objective.

Since knowledge requirements for simulation output analysis of non-terminating systems is directly correlated to higher-level learning, we illustrated the paradigm through the BMMKP and the optional-BMMKP (the 3-D knowledge pyramid/prism models of the highest-level, batch-means-method learning objective for our language-focused, undergraduate course).

The BMMKPs reveal how highly dependent and fully integrated this learning is to calculus, probability, statistics, and queuing theory—regardless of the simulation modeling language chosen to teach in the course. The BMMKP is then used to develop a set of lower-level learning objectives for the undergraduate course. Educational research has shown faculty (instructors) who teach using learning objectives provide their students with learning advantages, since they communicate to the students what deliverables are expected of them. The students also obtain a 'view' of the underlying knowledge required for meeting the learning objectives.

We are not suggesting that our KP/PM guarantees knowledge gain, or that all learners will attain the appropriate amount of knowledge in this manner (there are always 'exceptions to the rules'); or even how to measure knowledge gain. We foresee that the KP/PM may be used as a tool to assist the instructor in viewing the relationships between and among knowledge requirements for complex learning topics; and eventually, the development of specific learning objectives, assessment and misconception tools. With over ninety three industrial and systems engineering programs in universities over the United States alone, this work can serve as a useful tool to numerous instructors.

Future research will be aimed at developing other simulation KP/PMs. We also hope to compare the usefulness of the KP/PM against concept maps [10]. [11] developed a high-level concept map for output analysis but not specifically for the batch means method. Concept maps also show learning relationships but can be difficult to ascertain when many relationships exist (they almost become 'spaghetti diagrams'). They too suffer from what we found in Bloom's taxonomy—they do not visualize concurrent learning or provide for asynchronous learning environments.

References

1. W. D. Kelton, R. P. Sadowski and D. T. Sturrock, Simulation with Arena, 4th ed., McGraw Hill, Inc., New York, 2007.
2. B. Bloom, Taxonomy of Educational Objectives, Handbook 1: Cognitive Domain, Longman, New York, NY, 1956.
3. S. Goel and N. Sharda, What Do Engineers Want?: Examining Engineering Education through Bloom's Taxonomy. In: Snook, Chris (Ed.); Thorpe, David (Ed.), Creating Flexible Learning Environments: *Proceedings of the 15th Australasian Conference for the Australasian Association for Engineering Education and the 10th Australasian Women in Engineering Forum*, Toowoomba, Qld: Australasian Association for Engineering Education, 2004, pp. 173–185

4. R. M. Felder and R. Brent, The ABC's of Engineering Education: ABET, Bloom's Taxonomy, Cooperative Learning, and So On, *Proceedings of the 2004 American Society for Engineering Education Annual Conference & Exposition*, American Society for Engineering Education, CD-Rom, 2004.
5. R. Carter, A Taxonomy of Objectives for Professional Education, *Studies in Higher Education,* **10**(24), 1985, pp. 135–149.
6. C. W. Starr, B. Manaris and R. H. Stalvey, Bloom's taxonomy revisited: specifying assessable learning objectives in computer science**,** *Proceedings of the 39th SIGCSE technical symposium on Computer science education,* 2008, pp. 261–265.
7. A. J. Swart, Evaluation of Final Examination Papers in Engineering: A Case Study Using Bloom's Taxonomy, *IEEE Transactions on Education*, **53**, 2010, pp. 257–264.
8. M. H. W Hoffmann, *Using Bloom's taxonomy of learning to make engineering courses comparable*, European Association for Education in Electrical and Information Engineering Annual Conference Formal Proceedings, 2008, pp. 205–209.
9. K. O. Jones, J. Harland, J. Reid, M. V. Juliet, and R. Bartlett, Relationship between examination questions and Bloom's taxonomy, *Proceedings Frontiers in Education Conference*, CD-Rom 2009.
10. J. Turns, C. Atman and R. Adams, Concept Maps for Engineering Education: A Cognitively Motivated tool Supporting Varied Assessment Functions, *Institute of Electrical and Electronics Engineering Transactions on Education*, 43, 2000, pp. 164–173.
11. M. C. Court, The impact of using Excel macros for teaching simulation input and output analysis, *International Journal of Engineering Education,* **20**, 2004, pp. 966–973.

Christopher Poyner is a doctoral student at the University of Oklahoma's School of Industrial Engineering (OUSIE). His research is on formulating and implementing a reverse simulation modeling approach for supporting lean philosophies. He received his BS and M. S. from the School of Industrial Engineering.

Mary Court is an Associate Professor at OUSIE. She has researched and taught simulation analysis for over 10 years. Her current funded research programs are with the Federal Transit Administration.

Huong Pham is working on her doctoral degree at OUSIE. Her research focuses on developing the framework for simulating the impact transportation infrastructure design has on large-population evacuations. She received her accelerated BSIE/MSIE in the IT program of the School of Industrial Engineering.

Jennifer Pittman is a doctoral student at OUSIE. Her research is on developing an agent-based simulation paradigm to study the impact human-behavior has on large population evacuations. She received her accelerated BSIE/MSIE in the IT program of the School of Industrial Engineering with minors in computer science and philosophy.