Graduate Management Admission Council®

Adaptive Computer-Based Tasks Under an Assessment Engineering Paradigm

Richard M. Luecht

The University of North Carolina at Greensboro

Presented at the Item and Pool Development Paper Session, June 3, 2009

Abstract

This paper presents the theoretical foundations for self-contained, computerized adaptive performance tasks (CAPTs). Some important links are introduced between cognitive dimensions that affect item difficulty, measurement information along a latent scale, and features of assessment engineering task models, templates, and template nodes associated with particular CAPTs. A node-selection algorithm for task generation and a Bayesian calibration framework for calibrating the features associated with template nodes are also presented.

Acknowledgment

Presentation of this paper at the 2009 Conference on Computerized Adaptive Testing was supported in part with funds from GMAC®.

Citation

Luecht, R.M. (2009). Adaptive Computer-Based Tasks Under an Assessment Engineering Paradigm. In D. J. Weiss (Ed.), *Proceedings of the 2009 GMAC Conference on Computerized Adaptive Testing.* Retrieved [date] from www.psych.umn.edu/psylabs/CATCentral/

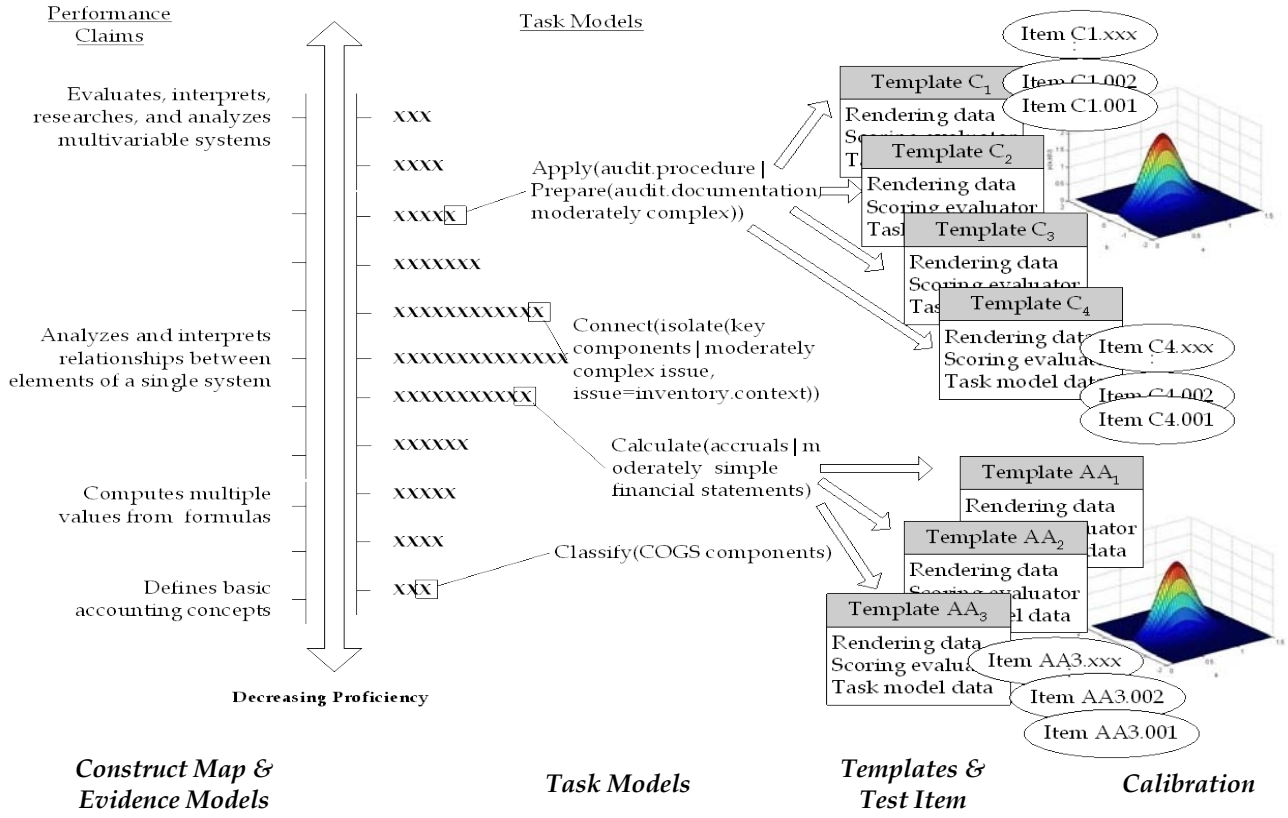
Author Contact

**Richard M. Luecht, The University of North Carolina at Greensboro,
209 Curry Building, 1109 Spring Garden Street,
Greensboro, NC 27412, U.S.A. Email: rmluecht@uncg.edu**

Adaptive Computer-Based Tasks
Under an Assessment Engineering Paradigm

Assessment engineering (Luecht, 2007, 2008a, 2008b; Luecht, Gierl, Tan, & Huff, 2006) is a highly structured way of designing constructs and building instruments and associated scales that measure those constructs. A construct map describes ordered performance expectations at various levels of a proposed scale. Empirically driven evidence models and cognitive task models are then developed at specified levels of each construct, effectively replacing traditional test blueprints and related specifications. Multiple assessment task templates are engineered for each task model to control item difficulty, covariance, and residual errors of measurement. Finally, psychometric models and procedures are used as statistical quality assurance mechanisms that can directly and tangibly hold item writers and test developers accountable for adhering to the intended test design at the item level. By using construct maps, evidence models, task models and templates, assessment engineering (AE) makes it possible to generate extremely large numbers of test forms with prescribed psychometric characteristics (e.g., targeted measurement precision). Figure 1 presents a view of the entire AE framework, from the construct map at the left through items and hierarchical calibration at the right.

Figure 1. Assessment Engineering at a Glance



This paper presents an extension of AE to include computerized adaptive performance tasks (CAPTs). In a traditional CAT, each item is selected to maximize the measurement precision relative to a provisional estimate of some latent trait. CAT requires every item to be calibrated using an appropriate item response theory (IRT) model so that estimates of item difficulty (location) and other characteristics can be used in the item selection process. CAPTs attempt to change this paradigm by presenting self-adapting assessment exercises.

Under AE, task models and templates can generate large numbers of items that form *classes*. In turn, individual items *inherit* the estimated psychometric characteristics of the class, via the task models and/or templates. A hierarchical Bayesian framework is used for calibration and to quantify uncertainty associated with the class of items sharing estimated item parameters (cf. Glas & van der Linden, 2003) . Assessment engineering works similarly with CAPTs. With CAPTs, features or components of the task models and/or templates are *altered in real-time* to actually vary the task difficulty in a systematic way. By applying a maximum information criterion to an item generation algorithm scripted as part of an AE template, the task features can be selected to create *highly variable computer-based performance tasks (i.e., items) that effectively adapt themselves to the proficiency of the examinee*. In this sense, the ensuing performance task or items become semi-intelligent measurement agents.

The theoretical foundations for CAPTs will be presented in this paper, describing the links between cognitive dimensions that affect item difficulty and features of AE task models, templates, and nodes. It will also present a node-selection algorithm for task generation and a Bayesian calibration framework for calibrating the features associated with template nodes.

Computerized Adaptive Performance Tasks

For CAPTs to be viable, we need to re-conceptualize the idea of an *item* from being a fixed entity to instead represent a set of assessment data objects that offer a distinct collection of manipulable assessment features capable of producing a large number of items, each with predictable difficulty and other psychometric characteristics. The term "task model**"** has been used to refer to as a large class of test items that share a common interpretation of the key declarative knowledge and procedural skills. In the present context, task models must include interpretative, performance-based elements (i.e., task performance expectations and conditions under which the task was to be performed); have a relative or absolute location on a particular proficiency scale; and include data-oriented specifications that link the required task actions, knowledge objects, contexts, and auxiliary tools to features and data objects that can be organized by a "template**"** that ultimately generates an entire family of test items. The templates are formal database structures that make it possible to control specific instantiations of each task model and create many replications, all of which operate similarly in a psychometric sense—that is, there are usually multiple templates designed for each task model and all templates and associated items for a given task model operate as isomorphs. Each task model has a unique *location* on an underlying measurement scale; therefore, all templates and items (assessment tasks) created from the templates share that location.

This idea of locating a task model on a measurement scale accomplishes three important goals: (1) it provides the basis for subsequently calibrating the task model to the scale; (2) it allows us to evaluate the measurement precision associated with a particular task model; and (3) it provides strong, empirically based quality control mechanisms to ensure that items created to represent a common task model perform similarly in a psychometric sense. In practice, each task

model is represented by multiple, exchangeable templates. In turn, each template can produce multiple items. However, CAPTs can actually adapt themselves to the examinee by modifying distinct, cognitively based features of the template and essentially generate a task that is ideally challenging for the examinee.

Underlying each task model and its family of templates is a cognitively based set of "dimensions" that can be empirically related to the psychometric operating characteristics of each CAPT. Three viable cognitive dimensions were introduced by Luecht, Burke, and Devore (2009) for purposes of mapping the difficulty of complex computerized performance exercises (CPEs) used on a national licensing examination in accounting. The first dimension is *task-complexity* and reflects the number and complexity of skills/task actions required by the examinee to complete the salient components of the task. These countable skills and actions must have a logical ordering that is maintained over context (e.g., "analyze" is *always* a more complex task than "identify"). In addition, the complexity must be logically and directly connected to specific, scored measurement opportunities associated with the template-based scoring evaluators[1]. In the case of CAPTs, it is probably not advisable to actually manipulate the task complexity in real-time because, while is possible, such manipulation could easily change the underlying statistical dimensionality and interpretation of the scales. The second dimension is *information density*. Cognitive research (e.g., Embretson & Wetzel, 1987; Kirsch & Mosenthal, 1990) has consistently confirmed the somewhat obvious intuition that highly dense text and graphics (e.g., using uncommon words, phrases, and complex constructions) will tend to make any assessment or learning task more difficult by increasing the cognitive load. This dimension merely attempts to capture the relative ordering of each task based on important features that increase or decrease information density. In the present context of CAPTs, we can go a step future to actually manipulate the information density in a prescribed way to effectively adapt the task to the examinee's apparent proficiency. The third dimension introduced by Luecht et al is *context complexity*. Highly complex settings with a great deal of distractive information ought to be more difficult than simple, well-structured settings. This context complexity dimension can likewise be manipulated as part of CAPT design.

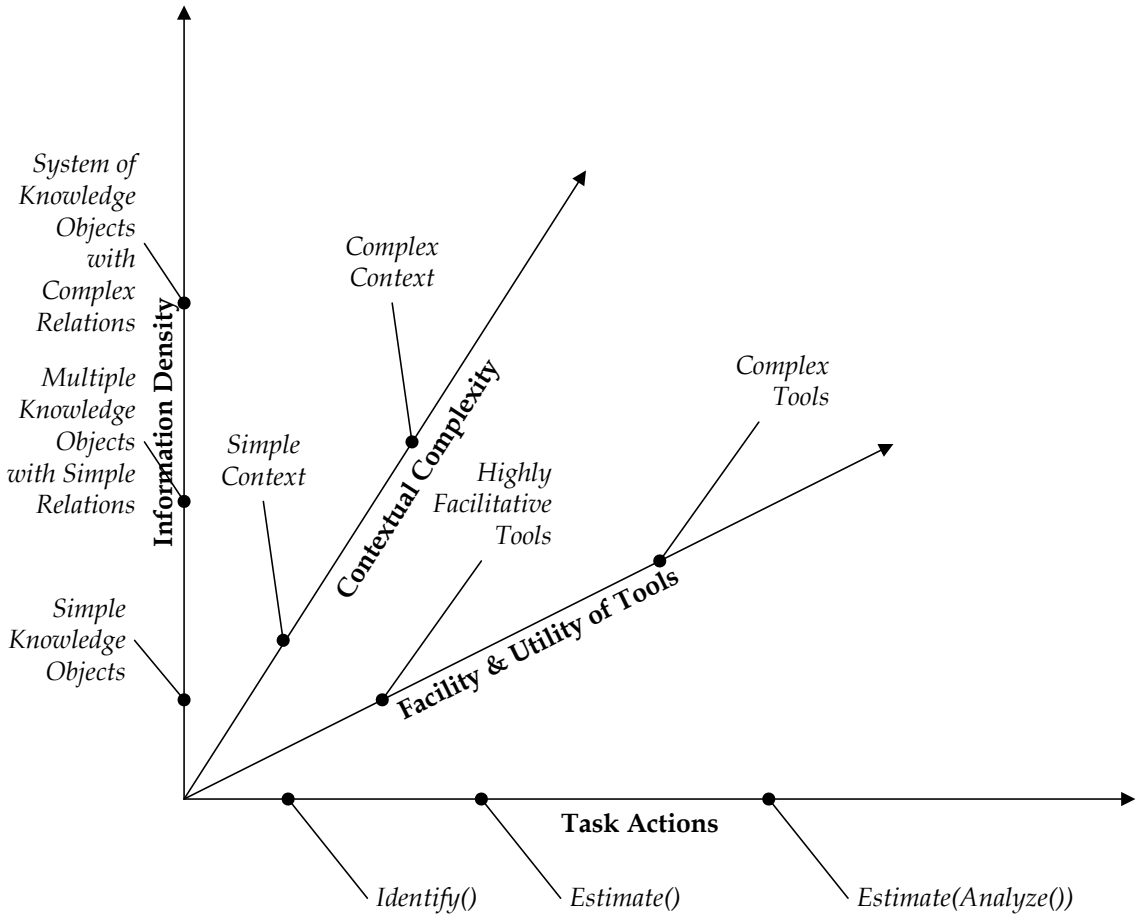
Luecht, Burke and Devore demonstrated that these types of cognitively-based complexity dimensions, based on subject-matter judgments, could be used to effectively order the CPEs and produce an underlying scale. That is, they *located* 14 CPEs being pilot tested for a national accounting examination on a composite metric, based on each exercise's complexity ratings. The surrogate statistical location parameters for the 14 CPEs produced almost identical scores as IRT-calibrated item parameter estimates, obtained using a polytomous Rasch model. The product-moment correlation between the complexity-based scores and the IRT-based proficiency estimates was .997.

A final dimension—and one not specifically cited by Luecht, Burke and Devore (2009) is the *facility and utility of auxiliary tools and information* (see Luecht, 2007, 2008a, 2008b) This dimension relates to specific components such as calculators, search engines, sorting tools, or other software components/agents that can facilitate completion of a particular task. For example, using a spreadsheet will tend to facilitate adding together a long list of numbers.

[1] This point about linking the task-model and template complexity components to aspects in the scoring is subtle, but extremely important. It does not matter how complex a task is if we are not scoring the very aspects of the response that make it complex.

These four cognitively based task dimensions are depicted in Figure 2. As we increase cognitive complexity along any of the dimensions, we are likely to correspondingly increase or decrease the difficulty and operating characteristics of the assessment task[2]. Ultimately, the task model acquires a central *location* on a scale by specifying and controlling the cognitive level of the action(s) or manipulation(s) required by each task model, controlling information density and context complexity by manipulating the number and complexity of knowledge objects for each task, as well as identifying key properties of the objects relevant to the task (facilitative or distractive) and constraining the number and properties of the relationships among objects, and by constraining the use of auxiliary tools and facilitative task components. This degree to which these manipulated dimensions actually alter the location of the task on a scale is empirically determined. Features that do not relate to changes in location are ignorable and exchangeable. AE therefore replaces traditional content blueprints with a cognitively based task-model specification that can be directly instantiated and manipulated via the templates, and that links the "content" to "scale location."

Figure 2. Cognitive Dimensions Affecting Task Location on a Scale



[2] Dimensionality and, in a related sense, the discrimination of an assessment task relative to the underlying scale(s), must also be considered via principled AE task-model and template design (see Luecht, Gierl, Tan & Huff, 2006).

AE-based task design creates direct links, first from task-model templates features and components of the rendering models and scoring evaluators to the four cognitive dimensions: (1) task model complexity; (2) information density; (3) context complexity; and (4) facility and utility of auxiliary tools. In turn, these four cognitive dimensions are linked to the empirical location of a particular task model on a scale. If we can systematically alter the cognitive dimensions by manipulating the task-model template features and components, we can therefore *change* the location of the task (subject to a caveat noted earlier regarding the changes to the task complexity possibly affecting statistical dimensionality of the scale).

We can use a *task-model grammar* (TMG; Luecht, 2007, 2008a, 2008b) to represent each assessment task with respect to the four cognitive dimensions. A TMG is essentially a predicate calculus of the form

$$action_2\left[action_1\left(is.related\left(object_1, object_2\right), object_3 \middle| context, aux.tools\right)\right].$$

This type of predicate calculus form has been successfully used in cognitive text processing research to denote semantic complexity (Kintsch, 1988). The "actions" can differ in *ordered* complexity (e.g., *identify* is a lower complexity action than *analyze*) or functionally nested such as $action_1($ $action_2((\ldots\ action_m)))$ to denote complexity. "Objects" can likewise differ in complexity by assigning more or less relevant properties to them. Those properties are defined external to the predicate clause. "Objects" can also differ in number, where including more objects implies greater cognitive load and, correspondingly, greater complexity. "Objects" can also be made more complex by incorporating "relations" that vary in type (unidirectional, bidirectional, hierarchical) and can take on additional properties that might affect the complexity of the "relations". Finally, the "context" and "auxiliary tools" are included in the predicate clause and both have properties and controls that can alter the complexity of the task in predictable ways. An important advantage of this predicate calculus representation of task complexity is that it also lends itself nicely to modern, object-oriented database designs. The importance of connecting the task complexity representation to specific, manipulable, data-based features of each task should become more apparent below.

A CAPT can be tailored to the examinee's proficiency by manipulating one or more of the four dimensions. The actual manipulation, however, is done by systematically changing the features of the CAPT *template* to essentially modify the difficulty of the CAPT, itself. Each CAPT template has three concrete subsystems of structured data components and procedural agents (i.e., formal database structures and programmed software components) that can be manipulated in real time: (1) the rendering model, (2) the scoring evaluator, and (3) the object data model.

An AE template *rendering model* elaborates on the notion of an "item shell" (e.g., Haladyna & Shindoll, 1989; LaDuca, 1994; Case & Swanson, 1998). Item shells present a generic item stem that has some fixed text and a small number of variable slots. The variable slots are filled by substituting words, phrases, or other item features that alter the surface appearance of the item, but not the essential task that the examinee is required to do. Similarly, an AE rendering model provides a detailed presentation format data and constrained interactive components for each task that capture the intended response(s) from the examinee (Luecht, 2001, 2006). The content presented in the rendering model will differ by item, but the essential manner in which

the information is displayed and the nature of the interactivity between the examinee and the assessment task generate a stored response (or other stored information such as response times, series of actions, etc.) is essentially the same for all items created from a particular template. The rendering model incorporates three types of components: (1) the semi-passive component(s); (2) the interactive, response-capturing components and controls; and (3) interactive tools, applications and agents. Each of these components has defined properties that may be modified, via the object data model (explained below) to alter the amount and type of information displayed and the nature of the interactivity between the examinee and the assessment task, as well as activate or deactivate any facilitative or punitive auxiliary tools and features.

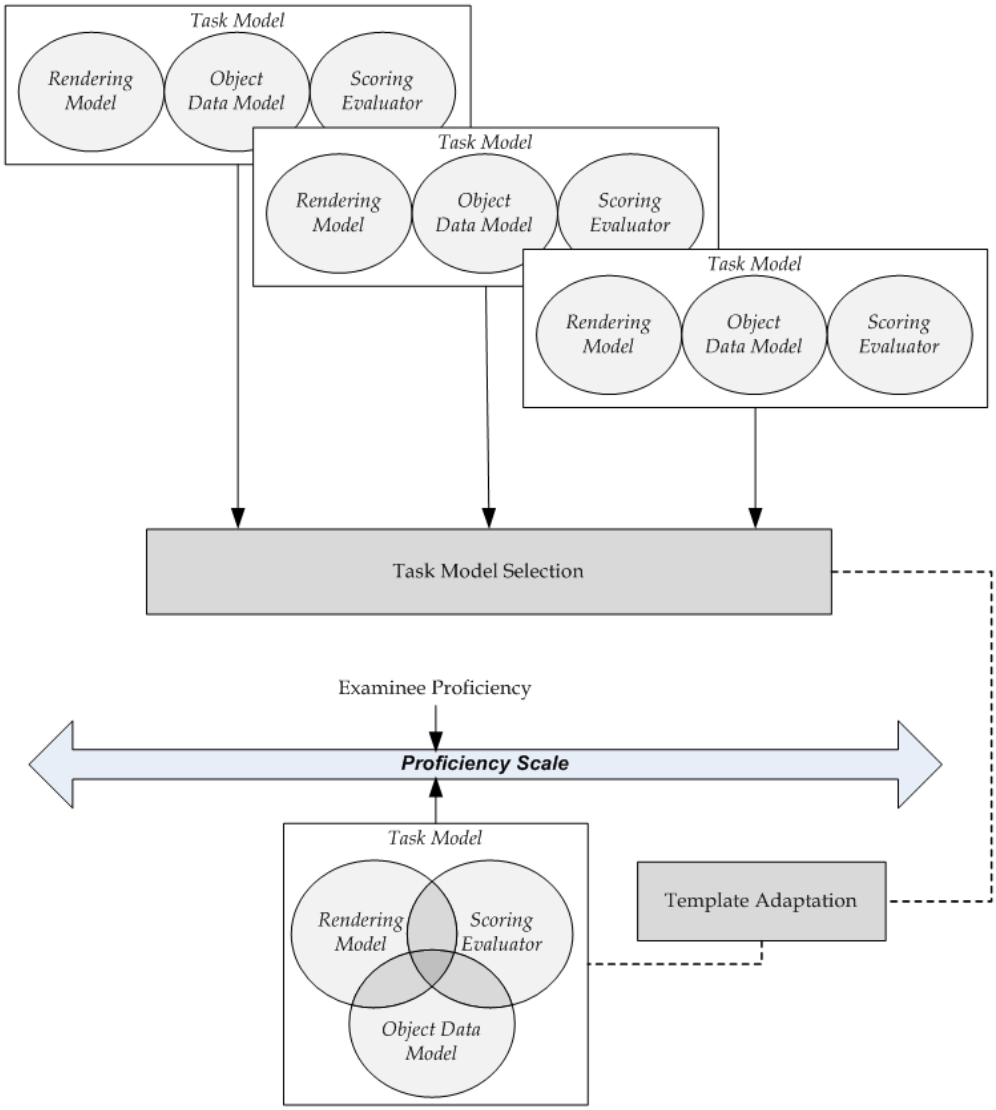
The second subsystem of a template consists of one or more *scoring evaluators*. A scoring evaluator produces item- or measurement-opportunity-level scores from a performance or response provided by the examinee (Luecht, 2001, 2005, 2006). Most scoring evaluators have three essential parts. The first part is a raw-response representation—that is, stored data that captures the essential, scorable actions or choices made by the examinee. The second part is the answer set or criterion. These can range from discrete answers such as multiple-choice answer keys to more elaborate criteria such as sets of feature-based matrices used in computing a score from parsed elements making up the raw-response representation. The third part of a scoring evaluator is the algorithm that applies the criterion to the responses and generates (usually) a numeric score. Algorithms can range from simple pattern-match evaluations to complex mathematical computations. A *scoring evaluator* embodies the performance expectations for each task which can range from simple pattern-match evaluators (e.g., a single-best answer multiple-choice evaluator that matches the stored answer to the examinee's response) or very elaborate, "intelligent" evaluators using neural nets, latent semantic analysis, or hidden Markov models.

The third template subsystem is the *object data model* provides a formal representation task features and scorable information associated with the rendering model, the scoring evaluator, or the content of the assessment task itself that can be used/activated by a real-time feature-selection software agent to manipulate the CAPT to an appropriate level of difficulty. The object data model also represents surface-level features that can be used to provide multiple item instances for security purposes. Essentially, every reusable component of the assessment task which is not locked down in the rendering model or coded as part of the scoring evaluator must be represented as variables—that is, "data"—in the object data model. This object-oriented data framework[3] provides a great deal of flexibility to systematically *modify complexity* by altering the rendering model and/or scoring evaluator(s). While it might appear that so many "moving parts" makes the problem of engineering each CAPT and subsequently calibrating the modifications to the assessment tasks intractable from a psychometric perspective, this is precisely what we want—enough moving parts to provide reasonable adaptation of the task and sufficient variations to provide security against memorization of specific "item shells" or otherwise highly constrained tasks that merely change some very trivial surface-level item features such as names or other identifying information. CAPTs have the potential to be far more complex than simple item models—for example, more complex than algorithmically generated mathematics items (e.g., Meisner, Luecht & Reckase 1993).

[3] Object-oriented data (OOD) structures are assumed as a natural way to implement CAPTs; however, other data structuring schemas are possible.

As shown in Figure 3, a CAPT has two possible levels of adaption. One level involves the task-model selection—that is, locating the appropriate task model. This level of adaptation is depicted in the upper part of Figure 3 where one of the three task models is selected. The second level involves modification of the CAPT template components to alter the difficulty of the task model features via the template (i.e., change the rendering model components, scoring evaluator, and object data model components) to locate the task near the examinee's apparent proficiency.

Figure 3. Two Levels of Adaptation for CAPTs



A remaining challenge for implementing CAPTs is to link changes in complexity, as depicted in Figure 3, to measurement precision—that is, to create a psychometric criterion for making changes in task complexity, information density, context complexity, and auxiliary tools. We can use a Rasch model for polytomous data to demonstrate how this criterion can be implemented in modifying a CAPT. In practice, there are many other, more complex

psychometric scaling/task calibration models that could be used.

Under the Rasch model, the conditional information for a polytomously scored item can be expressed as

$$I_i(\theta) = \sum_{k=0}^{m} P_{ik}(\theta)\left[k - \sum_{k=0}^{m} kP_{ik}(\theta)\right]^2 \quad (1)$$

where $P_{ik}(\theta)$ is a scoring category response probability for $k = 1, \ldots, m$ categories associated with item i (Linacre, 2005). The response probability is influenced by the difference between the examinee's proficiency, θ, and the conjunctive location, $b_{ik}^{*} = b_i + d_{ik}$, where b_i is the item location and d_{ik} is the threshold location for a particular category.

In the context of a CAPT, we can introduce different "*nodes*" associated with the rendering template, scoring evaluator, and/or data properties and content in the object data model. A node is a specific instance of a rendering template, a scoring evaluator, and relevant fixed elements of the object data model (i.e., selected fields, content, and properties) that can be associated with a location on the proficiency scale. The location can be characterized by a threshold parameter under a polytomous IRT model or by an item difficulty parameter under an IRT model for dichotomous data[4]. For example, it should be obvious that we can systematically *change* information by altering a particular threshold location parameter, d_{ik} in Equation 1. The same effect would be achieved by altering item difficulty parameters under a dichotomous IRT model. Each node must, therefore, be calibrated and be assigned a unique item difficulty parameter (under a dichotomous IRT model) or a threshold parameter (under a polytomous IRT model). The adaptation of the CAPT amounts to selecting different nodes in that affect task complexity, information density, context complexity, or facility and utility of auxiliary tools.

Figure 4 shows the node configuration for two assessment tasks that have the same location parameters, $b_1 = b_2$, but six different thresholds, d_{ik}, $k = 1, \ldots, 6$. The heterogeneous nodes displayed in the upper image might correspond to a generic configuration for a CAPT when the task is first selected. The more homogeneous nodes for the second task (depicted in the lower image in Figure 4) are actually selected by the CAPT software engine in real-time to adjust the complexity of the assessment task.

[4] Note: If response dependencies among the nodes can be minimized, a dichotomous IRT model might be wholly satisfactory for calibration purposes (Goodman & Luecht, 2009).

Figure 4. Node Configurations for Two Tasks: Same Difficulty, Different Thresholds

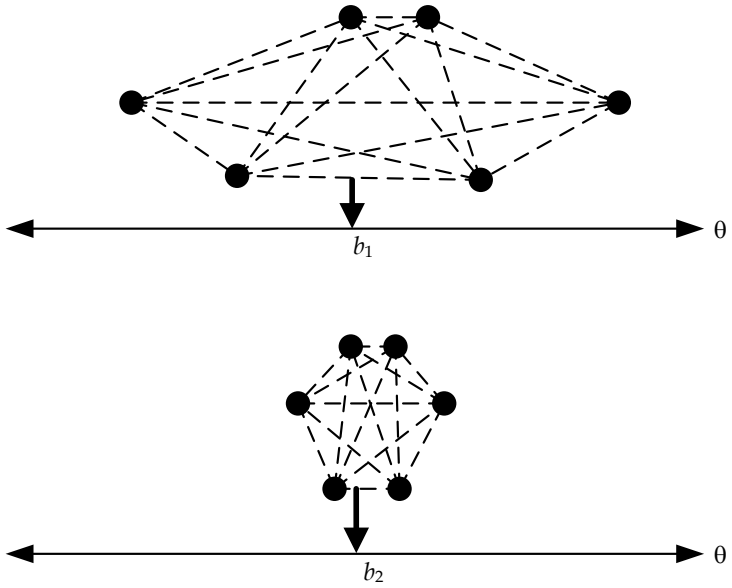
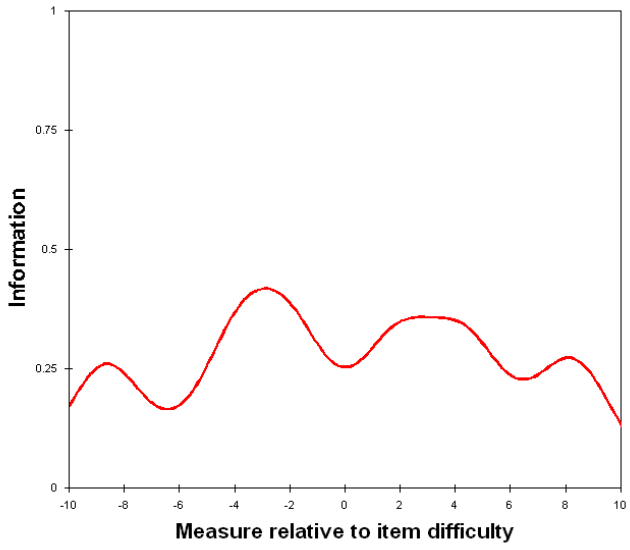


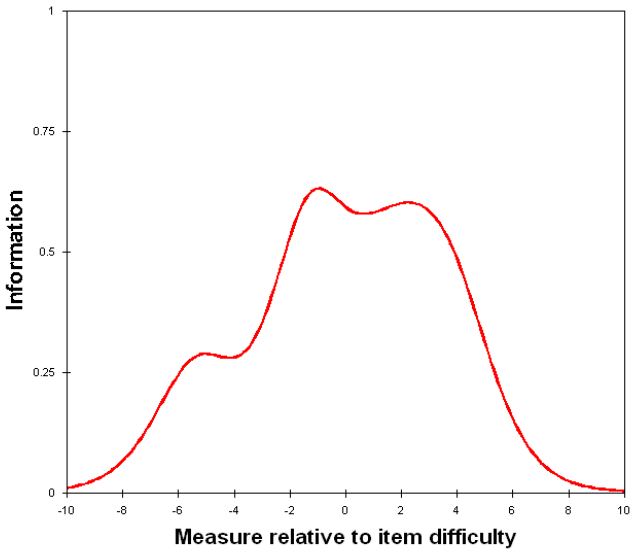
Figure 5 presents the corresponding measurement information curves for these two assessment tasks. It should be apparent that the information is higher and more centralized for the second task. This is the type of selective targeting of the nodes that would be achieved by the CAPT selection engine.

Figure 5. Measurement Information for Two Assessment Tasks

a. Heterogeneous Nodes



b. Homogeneous Nodes



An Algorithm and Implementation Strategy for Adaptation

The node configuration introduced above is a convenience that lends itself nicely to a two-stage adaption and selection algorithm. Although it might seem tempting to propose first selecting the CAPT task model/template and then modifying the template components to maximize score precision, a reverse optimization strategy is actually more effective to ensure a maximally informative CAPT. The first stage, therefore, adaptively selects (configures) the nodes for each CAPT to generate a pool of maximally informative tasks at the current, provisional proficiency score estimate. The second stage selects the particular CAPT with maximal information, given that all CAPTs have been arranged into the most informative configuration of nodes possible. The process iterates until an appropriate stopping criterion is satisfied (fixed number of CAPTs administered, minimum error variance achieved, or a prescribed amount of decision accuracy is reached).

Let each node represent a unique component related to the one of the four dimensions making up each task model: (1) task complexity, (2) information density, (3) context complexity, or (4) facility and utility of auxiliary tools and information. We can conceive of these dimensions as primary skill dimensions. Any CAPT may have no representation, one component represented, or multiple components represented on each of the dimensions. For example, a particular CAPT may have no nodes related to facility and utility of auxiliary tools and information, three nodes for task complexity, seven nodes for information density, and four nodes for context complexity. Other attributes of the nodes can also be represented—such as critical content attributes that must be represented and that have a direct or indirect effect on item difficulty. A predetermined specification as to the counts of nodes along each of the dimensions is assumed—conceptually similar to specifying the numbers of items meeting particular content requirements on a conventional test.

We can represent the selection of nodes as well as attributes for each node as binary indicators, x_{ik} 0 {0,1} for $i = 1, \ldots, I$ CAPTs in the item bank and $k = 1, \ldots, K_i$ nodes associated

with a particular CAPT. We need to add two additional indexing variables: one to represent the $j = 1$ to J dimensions[5] and a second to represent the $h = 1, ..., n$ CAPTs to be selected for a particular examinee. The basic configuration maximization algorithm can be expressed, using linear programming notation described by van der Linden (2005) as

$$\text{maximize } \sum_{k=1}^{K_i} I_{ik}\left(\hat{\theta}^{(r-1)}\right)x_{ik} \quad \text{(maximize the information)} \tag{2}$$

subject to

$$\sum_{k\in V_j} \geq n_{j0h}\,, \quad \text{(minimum count of nodes for each dimension, } j\text{)}, \tag{3}$$

$$\sum_{k\in V_j} \leq n_{j1h}\,, \quad \text{(maximum count of nodes for each dimension, } j\text{)}, \tag{4}$$

and

$$x_{ik} \in \{0,1\}\,. \tag{5}$$

For convenience, we represent the node information functions as discrete quantities, $I_{ik}(\theta)$. The category information is easily derived under most polytomous IRT models as the proportion of the information associated with each score category (node in this case); that is, $I_{jk}(\theta)=I_i(\theta)P_{ik}(\theta)$, where $P_{ik}(\theta)$ is the category response probability. This algorithm is repeated for all $I - r + 1$ unselected CAPTs.

The second stage of selects the r^{th} CAPT that is maximally informative. A second binary indicator (decision) variable is introduced, $y_i = \{0,1\}$, to represent the selections of the CAPTs under the following optimization model.

$$\text{maximize } \sum_{i=1}^{I} y_i\left[\sum_{k=1}^{K_i} I_{ik}\left(\hat{\theta}^{(r-1)}\right)x_{ik}\right] \quad \text{(maximize the CAPT information)} \tag{6}$$

subject to

$$\sum_{i\in\bar{R}_h} = h-1, \quad \text{(previously selected CAPTs)}, \tag{7}$$

$$\sum_{i=1} = n\,, \quad \text{(test length)}, \tag{8}$$

and

$$y_i \in \{0,1\}\,. \tag{9}$$

Equations 2 to 5 and 6 to 9 can be further augmented with addition constraints for content, item exposure, etc. Only the basic outline of the test assembly problem is presented above. A method of optimal test assembly for item sets with "shadow tests" (see van der Linden, 2005, Chapters 7 and 9) can also be applied here with the necessary modification to treat the nodes as items.

Hierarchical Calibration and Scoring

One of the primary advantages of using AE templates is that we can create many

[5] J = 4 in this example since no additional attributes beyond the primary cognitive task dimensions are represented.

interchangeable items from each template (see Figure 1). The same principle applies to CAPT templates and nodes. By manipulating *incidental* content (cf., Irvine, 2002) it should be obvious that multiple instances of each node can be produced from a single CAPT template. There are many possible surface-level changes in content (e.g., settings, names, identifying information) or the rendering model that will not affect the task complexity, information density, context complexity, or facility and utility of auxiliary tools and information.

A hierarchical Bayesian calibration and scoring framework for families of dichotomous items (i.e., shells or clones) introduced by Glas and van der Linden (2003) can be readily extended to the problem of calibrating the nodes for CAPTs. The unique instances of each node (or unique instances of a collection of nodes) are calibrated as a class that shares the same underlying distribution of item parameter estimates. The mean(s) of the posterior distribution of item parameter estimates for a particular node (or class of nodes in the polytomous case) are used for selecting the best CAPT and then adaptively modifying it to produce maximum score precision (or, alternatively, minimum expected posterior error variance). For scoring purposes, we can integrate first over the distribution of item parameter estimates to estimate the expected response function and then integrate over the posterior distribution of proficiency (Glas & van der Linden, Eq. 18).

Glas and van der Linden's hierarchical Bayesian approach offers the advantage of efficiency in pretesting assessment tasks as well as overcoming various security risks associated with testing over time—specifically overcoming the associated item exposure risk when there are relatively small item pools in high-stakes settings. That is, the number of combinations of node configurations, allowing for variation in the rendering models, scoring evaluators, and object-data model content, can produce a enormous number of potential assessment tasks. If proper quality control procedures are used to "tighten" the templates, every new instantiation of the "item" associated with a particular template does not need to be pretested and calibrated. Instead, a common set of *hyper-parameter* estimates can be used, empirically updated over time. Furthermore, particular templates that fail to converge to a acceptable degree of estimation error variance and model fit common, or item writers who stray from the task-model design specifications, can be identified and dealt with in an appropriate way.

Concluding Comments

Assessment engineering (AE) is a framework for designing ordered proficiency-based construct maps and evidence models, associated task models that isolate precision where it is needed, templates to provide a consistent type of measurement information that can be manipulated as needed along a measurement scale, and a hierarchical Bayesian approach to calibration and scoring that provides efficiencies in pretesting, as well as certain security advantages. Computerized adaptive performance tasks (CAPTs) make use of AE task models and templates, but actually manipulate various template components to correspondingly change the task complexity, information density, context complexity, and facility and utility of auxiliary tools and information. Logically determined complexity indices assigned along these latter four dimensions have been recently demonstrated to correlate strongly with item difficulty for complex performance assessments in accounting. Changes in task complexity, etc., implement via changes in a template will result in corresponding change in location (difficulty) of the CAPT and further alter the amount of measurement information (precision) provided by the task(s).

However, each AE template has multiple "moving parts", making principled manipulation an

incredibly complicated undertaking. To avoid those complications, a structured node-configuration was introduced to represent the state or instance of a particular template. The nodes provide a convenient way of linking the instantiation of a template to IRT item parameters. The nodes can, therefore, be calibrated and used adaptively to generate the most informative CAPT possible. A two-stage selection algorithm was recommended to choose the most informative configuration of nodes (i.e., the particular state of a rendering model, scoring evaluator, and content in the object data model). Finally, a hierarchical Bayesian calibration and scoring approach was recommended to allow multiple instances of the nodes under a particular template to be used interchangeability.

This presentation stops short of demonstrating an actual implementation of a CAPT. That is a limitation, but remains a viable topic for future research. What is presented here is a practical way of implementing self-adapting, complex performance assessment tasks that will work in terms from the multiple perspectives of test development, computer systems and software design, as well as psychometrics. In that regard, it is hoped that readers see a useful contribution on which to build.

References

Bradlow, E. T., Wainer, H., & Wang, X. (1999). A Bayesian random effects model for testlets. *Psychometrika, 64*, 153-168.

Embretson, S. E., & Wetzel, C. D. (1987). Component latent trait models for paragraph comprehension. Applied Psychological Measurement, 11, 175-193.

Goodman, J.; Luecht, R. (2009, April). *An Examination of the Residual Covariance Structures of Complex Performance Assessments Under Various Scaling and Scoring Methods*. Paper presented at the Annual Meeting of the National Council on Measurement in Education.

Case, S. M.; & Swanson, D. B. (1998). *Item Writing Manual, 3rd Edition*. Philadelphia, PA: National Board of Medical Examiners.

Glas, C.E.S.; & van der Linden, W. J. (2003). Computerized adaptive testing with item cloning. *Applied Psychological Measurements, 27(4)*, 247-261.

Haladyna, T.M., & Shindoll, R.R. (1989). Items shells: A method for writing effective multiplechoice test items. *Evaluation & the Health Professions, 12*, 97-104.

Irvine, S. H. (2002). The foundations of item generation for mass testing. In S. H. Irvine and P. C. Kyllonen (Eds), *Item Generation for Test Development* (pp. 3-34). Mahwah, NJ: Larwende Erlbaum Associates.

Kirsch, I. S., & Mosenthal, P. B. (1990). Exploring document literacy: Variables underlying the performance of young adults. *Research Reading Quarterly, 25*, 5-30.

LaDuca, A. (1994). Validation of professional licensure examinations: Professions theory, test design, and construct validity. *Evaluation & the Health Professions, 17*, 178-197.

Linacre, M. (2005). Dichotomous and polytomous category information. *Rasch Measurement Transactions, 19:1*, 1005-6.

Luecht, R. M. (2001, April). *Capturing, codifying and scoring complex data for innovative computer-based items*. Paper presented at the Annual Meeting of the National Council on Measurement in Education, , Seattle, WA.

Luecht, R. M. (2005, April). *Extracting Multidimensional Information from Multiple-Choice Question Distractors for Diagnostic Scoring*. Paper presented at the Annual Meeting of the National Council on Measurement in Education, Montreal, Quebec.

Luecht, R. M. (2006, May). *Engineering the Test: Principled Item Design to Automated Test Assembly.* Invited special event presentation at the Annual Meeting of the Society for Industrial and Organizational Psychology.

Luecht, R. M. (April, 2007). *Assessment Engineering in Language Testing: from Data Models and Templates to Psychometrics*. Invited paper (and symposium) presented at the Annual Meeting of the National Council on Measurement in Education, Chicago.

Luecht, R. M. (March, 2008). *The Application of Assessment Engineering to and Operational Licensure Testing Program*. Invited paper presented at the Association of Test Publishers Annual Meeting, Dallas.

Luecht, R. M. (November, 2008). *Assessment Engineering in Test Design, Development, Assembly, and Scoring*. Keynote address at the East Coast Organization of Language Testers (ECOLT), Washington, DC.

Luecht, R. M.; Gierl, M. J.; Tan, X.; & Huff, K. (April, 2006). *Scalability and the Development of Useful Diagnostic Scales*. Paper presented at the Annual Meeting of the National Council on Measurement in Education, San Francisco, CA.

Meisner, R.; Luecht, R. M.; & Reckase, M.R. (1993). *Statistical characteristics of items generated by computer algorithms, ACT Research Report Series, RR-93-9*. Iowa City, IA: ACT, Inc..

Muraki, E. (1992). A generalized partial credit model: application of an EM algorithm. *Applied Psychological Measurement, 16*, 159-176.

Samejima, F. (1969). Estimation of ability using a response pattern of graded scores. *Psychometric Monograph, No. 17*.

van der Linden, W. J. (2005). *Linear Models for Optimal Test Design*. New York: Springer.