Using Competencies to Map Performance across Multiple Activities

Robby Robson
Eduworks Corporation
Corvallis, OR
robby.robson@Eduworks.com

Jonathan Poltrack
Advanced Distributed Learning (ADL) Initiative
Johnstown, PA
jonathan.poltrack.ctr@adlnet.gov

ABSTRACT

When a single training system accumulates data on learner performance, the data are stored in a way determined by the system's designers. This enables the system to access these data and to apply them to its interactions with learners. In environments such as live-virtual-constructive federations, each component may store performance data in its own way, making it difficult for one component to access and use data produced by another. To enable cross-component sharing of performance data, it is necessary to establish shared definitions of skills and outcomes; create a common language for expressing performance data; interpret data produced at wildly differing levels of granularity; and (in some cases) satisfy a large array of security and privacy requirements.

This paper is based on work done by the US Advanced Distributed Learning (ADL) Initiative, the Credential Engine foundation, and several standards bodies. It starts by discussing the above challenges and their manifestations in use cases ranging from federations of sophisticated adaptive training and simulation systems to more traditional online learning environments. The paper then describes a potential solution for collecting and processing assertions of competency, skills, and performance from multiple sources. Each assertion is of the form "Learner X has (or has not) achieved competency Y at level Z with confidence p based on evidence E." "Competencies" are drawn from shared, machine-readable frameworks that can represent knowledge, skills, ability, and objectives. Assertions can be collected directly or generated by ingesting granular performance data and correlating it to competencies, enabling algorithms that use explicit rules and relationships to draw further inferences.

We have recently tested one such system with Army special operators (n = 79) at the JFK Special Warfare Center and School as part of the ADL's Total Learning Architecture demonstration. The paper discusses our approach to establishing trusted networks and complying with privacy and security requirements; how we used the Experience API (xAPI) and industry standards; and lessons learned from that event.

ABOUT THE AUTHORS

Dr. Robby Robson is a researcher and innovator who has contributed to numerous technologies and standards widely used today in learning management systems, digital libraries, cryptography, and other areas. He is Principal Investigator (PI) on the ADL Initiative's Competency and Skills System (CASS) project and has worked in the area of competencies and outcomes-based education and training since the late 1990's. He is former chair of the IEEE learning technology standards committee and is currently a member of the IEEE Standards Association Standards Board, the IEEE Future Directions Committee, the Learning Resource Metadata Initiative, and various related technical advisory boards. Dr, Robson received his doctorate in mathematics from Stanford University and co-founded Eduworks Corporation in 2001.

Jonathan Poltrack is a software engineer and project manager who has been involved with technology-assisted learning and the DoD's Advanced Distributed Learning (ADL) Initiative since 1999. Through 2004, Jonathan was a contributor, editor and developer of the Sharable Content Object Reference Model (SCORM), which became a de facto global e-learning specification. In 2009, Jonathan rejoined ADL with the goal of modernizing aging learning technologies by creating a new series of learning platform specifications to enable the use of emerging technologies and learning science. The first of these specifications, the Experience API (xAPI), updates the SCORM Run-Time and expands types of learning content to be inclusive of native handheld apps, simulations, virtual worlds, sensors, games and other content modalities. Currently, Jonathan is organizing many competency-based education projects at the ADL Initiative

Using Competencies to Map Performance Across Multiple Activities

Robby Robson
Eduworks Corporation
Corvallis, OR
robby.robson@Eduworks.com

Jonathan Poltrack
US Advanced Distributed Learning Initiative
Johnstown, PA
jonathan.poltrack.ctr@adlnet.gov

1. INTRODUCTION

In live-virtual-constructive (LVC) and synthetic training environments (STEs) (Johnston, et. al., 2015) learners often engage in multiple activities provided by multiple components of the environment. For example, a learner might take a pre-test delivered by a learning management system (LMS), participate in a live briefing, train on a simulator, and be assessed by performance in a complex multi-player exercise that includes LVC components. As learners engage in these various activities with these various components, evidence is produced about the *competencies* that the learners possess or do not possess. Here, and in this paper, "competency" is used as a generic term that includes individual or related sets of skills, knowledge, abilities, and attitudes (Chouhan & Srivastava, 2014). The evidence collected can include direct or indirect evidence derived from actions within a simulation, game, live exercise, or other learning activity, as well as direct assessments of learning objectives (LOs) that correspond to the attainment of competencies.

1.1 Improving Training Effectiveness

Ideally, a training system should not engage a trainee in an activity for which the trainee does not have the necessary prerequisites or in an exercise that teaches competencies the trainee has already mastered. Thus, to personalize and properly sequence training activities, it is necessary to have a picture of the relevant competencies that a trainee does or does not hold. We call this a *competency profile*. Many training systems maintain competency profiles in the form of which LOs have been achieved. For example, traditional SCORM-based eLearning content can report the status of objectives to a learning management systems (LMS), which then stores these for retrieval by the content. Intelligent tutors maintain learner models that include (and are often completely comprised) of competencies, while serious games and simulations used for training maintain information on players' levels and achievements that loosely correspond to competency profiles. These profiles, however, are scoped to a single training system. *The primary goal of the work in this paper is to enable a single competency profile to be shared among multiple systems* (Figure 1), *and to enable multiple systems to make assertions about the same individual with respect to the same competency.*

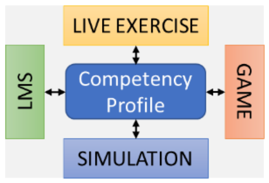


Figure 1: Shared Profile

1.2 Problems Addressed

Achieving the above goal requires (1) creating a shared understanding of the competencies addressed, and (2) collating evidence produced by multiple systems into a single shared profile. Both are necessary to create a coherent training experience in which the learner is effectively and efficiently guided from one activity to the next and in which one activity can adapt based on performance during a prior, different activity. As an example, consider a soldier whose cyber skills are evaluated in an LMS course and who then participates in a team exercise in a virtual cyber range. As things stand today, the role played by the soldier and the scenario used in the exercise are likely to be assigned without any knowledge of the soldier's skills. Using the approach described in this paper, the cyber range could adapt the training scenario and the soldier's mission role to the skill level the soldier has previously demonstrated, thereby providing training that is more germane and that is in the soldier's zone of proximal development (McLeod, 2012).

1.3 Overview of this Paper

In this paper, we present an approach to creating a shared understanding of competencies and shared competency profiles that was sponsored by the US Advanced Distributed Learning (ADL) Initiative. This approach is based on *competency frameworks*, which are explained in Section 2, and a method of processing *competency assertions* from multiple sources, which is discussed in Section 3. An open source implementation of this approach is described in Section 4, and the results of the first phase of a design-based research study with 79 participants are discussed in Section 5. Section 6 discusses lessons learned and next steps.

2. COMPETENCIES AND COMPETENCY FRAMEWORKS

Competencies can to refer to knowledge, skills, and abilities (or attitudes) (KSAs). Competencies can also refer to learning outcomes and learning objectives (terminal or enabling) in the sense that an LO is a competency that is to be obtained. Several standards exist for representing competencies in an interoperable manner (see Section 6). Common elements in these standards include:

- A unique ID for the competency
- A description of the competency
- Relationships between a given competencies and other competencies (see below)
- Levels at which the competency can be held (see below) (e.g., beginning, intermediate, advanced, or 1 – 5)
- Methods for assessing the competency (e.g., a rubric)

2.1 Relationships Among Competencies

The most common types of *relationships* express that one competency is part of, or a prerequisite for, a higher-level competency, or that two competencies are equivalent or similar. As examples:

- The ability to properly aim a weapon may be considered part of a marksmanship competency. Thus *aiming* is part of, or is a sub-competency of, or is a child of *marksmanship.*
- Knowledge of addition is a *prerequisite for* (but not part of) competency in multiplication, i.e. it is generally believed that one master addition before multiplication, and a teacher would be loathe to teach a student multiplication if the student did not know how to add.
- The LO "demonstrates familiarity with the iPad" *enables* "successfully operates a tablet" and would likely be used as an enabling learnig objective (ELO) in a course that had successful operation of a tablet as a terminal objective (TLO) although it is not a prequesite since there are other types of tablets.
- The Grade 3 mathematics standards in Virginia and California cover the same knowledge, skills, and abilities, so a student mastering one set will have mastered the other (Virginia, 2017; California, 2013. The competencies are grouped differently, e.g. fractions are under "Numbers and Number Sense" in Virginia and "Number and Operations – Fractions" in California, but the overall standards are *equivalent*.
- Competency in piloting a Boeing 737 is *similar* to competency in piloting an Airbus 320, although there are enough differences that certification in one does not automatically confer certification in the other.

Borrowing terminology from the W3C Simple Knowledge Organization System (SKOS) (W3C, 2009), if competency **A** is part of or a special case of competency **B**, we say that **A** *narrows* **B**, and **B** *broadens* **A**.

2.2 Competency Levels

Several standards for competencies include the notion of a *level*. This can be confusing because there are two different types of level. The first defines different levels of performance on the same set of skills and abilities, usually as measured through an assessment. For example, a soldier who demonstrates the ability to hit a target 90% of the time might be assigned a higher level of competence than one who hits it only 75% of the time. We call these *performance levels*. Another type of level is typified by a scale such as beginner, journeyman, and master. In most real-world cases, progressing through these levels requires acquiring new skills and abilities, as well as performing better on existing ones. As such, different levels of this type should be modeled as different competencies rather than levels of the same competency, e.g. with the beginning level narrowing the journeyman level and the journeyman level narrowing the master level. In this paper, we restrict the notion of the level of a competency to a specific, measurable, and attainable performance level.

2.3 Rollup Rules

A concept that is not included in most standards, but that is important in applications, is that of a *rollup rule.* A rollup rule defines how mastery of a competency **A** can be determined from mastery of other competencies $\mathbf{B_1},\ldots,\mathbf{B_K}$, where the $\mathbf{B_k}$ usually (but not necessarily) narrow **A**.

Example: A communication competency might be demonstrated by demonstrating either excellent verbal communication skills *or* by demonstrating excellent written communication skills and good verbal communication skills. If **C** is the communication competency and **V** and **W** are the verbal and written communication skills competencies, then "excellent" and "good" are levels of **V** and **W**, both of which narrow **C,** and there is a rollup rule that says: **(Excellent V)** or (**Excellent W** and **Good** V) implies **C.**

2.4 Frameworks

Sets of competencies that pertain to a task, job, or subject are often organized into *frameworks.* Frameworks can have no structure, i.e. just be collections of competencies, or be structured by relationships among the competencies they contain. Typically, frameworks are full or partial trees, ordered by a broadens/narrows type relationship, and frameworks are abundant in the real world. Examples include state curriculum standards, the Department of Labor's collection of skills and discrete work activities associated with specific jobs, the National Institute for Cybersecurity Education (NICE) workforce framework (Newhouse et. al., 2016), lists of skills or professional requirements developed by associations representing occupations ranging from metalworking to lawyering, and lists of skills associated with military occupational specialties (MOS). However, most of these frameworks only exist in "paper" format, i.e. as PDFs or web pages, with no means for them or the competencies they contain to be accessed via a web service call or application programming interface (API). To enable competencies to be accessed and used by the virtual or constructive elements of an STE, let alone be shared among them, it is necessary to store frameworks and competencies in a machine-readable format. This is one major function of the system described in Section 4.

2.5 The Structure of Frameworks

Many frameworks are structured with hierarchical relationships, usually a parent/child relationship that is not explicitly defined, a broadens/narrows relationship, a prerequisite relationship, or an enables relationship. Examples were given in Section 2.1, where we also mentioned the SKOS relationships "broader" and "narrower." In SKOS, "broader" means more general, and "narrower" means related but not more general (W3C, 2009, Section 8.1). Thus "vehicle" is broader than "car" and "engine" is narrower than "car," even though an engine is not a type of car. From an application perspective, the hierarchical relationships in frameworks have similar effects on instructional design, adaptation, and sequencing of activities. As a result we can usually get away with a single relationship. In notation, we write **A>B** to mean that **A** broadens, requires, or is enabled by **B**. The crucial point is that *if **A>B** and a learner has not mastered **B**, then there is an inference that they have not mastered and are not ready to acquire **A**.*

Most real-world frameworks are structured by a single hierarchical relationship, and although some have tree structure that indicates a "sub-competency" relationship plus an additional formal notion of prerequisites or enabling learning objectives, for our purposes we can consider every framework **F** to have a single relationship that, for convenience, we will call "broadens" and denote by **A>B.** Using standard definitions from graph theory, we say that **A** is a parent of **B** (and **B** is a child of **A**) if **A>B** and there is no **C** with **A>C>B** and we make **F** into a directed graph by defining the nodes to be the competencies and adding a directed edge from each competency to each of its children. In all non-contrived examples of frameworks that we have encountered, frameworks are directed acyclic graphs (DAGs), meaning there is no directed path (of length greater than 0) from any competency to itself. In most cases, real-world frameworks are collections of trees, i.e. every competency has at most one parent, but there are cases where, for example, writing skills are required for multiple tasks in a framework and are therefore children of multiple competencies in the framework's graph structure.

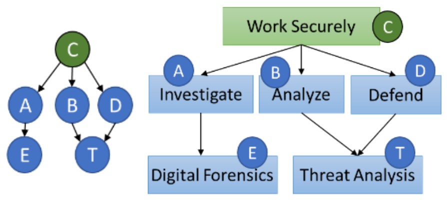


Figure 2: Portion of NICE Framework

Example: A small portion of the National Initiative for Cybersecurity Education (NICE) Cybersecurity Workforce Framework (Newhouse *et. al.*, 2016) is shown in Figure 2. The arrows show broadening relationships. In Figure 2, the framework is abstracted to the DAG shown on the left. In the DAG, **A**, **B**, and **D** are the children of **C**. **E** is not a child of **C** because there is a competency (**A**) between **C** and **E**, although **E** is a descendant of **C** because there is a directed path from **C** to **E**. Similarly, **C** is a parent of **A** and an ancestor but not a parent of **T**.

Another type of relationship is an equivalence relationship. For the sake of completeness, we point out that although these should not appear in a single framework since they would be redundant, what can happen is that different authors add equivalent competencies with different names to the same framework. In this case we can modify the framework to be a DAG by identifying the equivalent competencies with each other, and in the rare case that this creates a cycle, collapsing the entire cycle to a single competency. The justification is that holding (or not holding) any competency in the cycle implies holding (or not holding) all others.

3. ASSERTIONS

Competencies and Competency Frameworks may be used *as is* in instructional systems design, but their most common applications involve determining which competencies are held by one or more individuals. In this regard, it is naïve to believe that we can know with certainty whether an individual actually holds a given competency. Even if a learner passes an exam or demonstrates a physical ability, it is possible that the learner was lucky, or the assessment was not valid, or that the learner's current knowledge or ability is different than it was when tested. Moreover, we often determine competency based on certifications, transcripts, and other credentials whose validity and reliability unknown. For these reasons, we view evaluations of competency, whether made by assessment, observation, or inference, as *assertions* about an individual's competency rather than a determination of their competency.

To properly interpret assertions, it may be necessary to evaluate:

- Who (or what) made the assertion? How reliable is the source?
- What evidence is there to back up the assertion?
- How confidently is the assertion made? For example, it is a leap of faith to conclude that a pilot can make good decisions under stressful circumstances based on one observation.
- How long ago was the assertion made, and how does that affect the validity of the assertion? People tend to forget knowledge they don't use and lose skills they don't practice.

With this in mind, the general form of a competency assertion is:

> *An* ***agent*** *asserts at a specified* ***date (and time)*** *that an* ***entity*** *has (or does not have) a* ***competency*** *at a specified* ***level*** *with a specified* ***confidence (or estimated probability)*** *based on specified* ***evidence*** *and with the assertion expiring at specified* ***date (and time)*** *(or, more generally, with a specified* ***decay function*** *that defines the rate at which confidence in the assertion decays over time).*

As an example, suppose a pilot successfully completes a mission in an F-18 simulator during an LVC exercise and the software reports that the pilot has demonstrated the ability to fly an F-18. This could be translated into the assertion:

> "Training System TF18-R88-90 asserts on 30 November 2017 that LT CMDR Maria Rodriguez can fly an F-18. This assertion is made with 40% confidence based on her completion of a simulated mission (with details and results available at https://training.navy.mil/LVCmissions/F60A9E14/) and expires on 29 November, 2019."

If six months later CAPT Jones sees that LT CMDR Rodriquez has been certified to fly an F-18, the Captain may infer that she also knows how to operate the aircraft's navigation system, which in our terminology is a narrower competency than flying the aircraft. This can be translated into an assertion of competency made by CAPT Jones with the certification as evidence. A pilot certification is relatively strong evidence of the ability to carry out all of the routine tasks required to fly the aircraft, so we might be inclined to trust this assertion regardless of whether CAPT Jones specified a confidence level.

At the current time, human evaluators are more likely than software-based systems to make judgments about confidence and to record notes about performance, but as the algorithms used in STEs become more sophisticated, we expect that all of these data will become more readily available through software-based reporting mechanisms. Similarly, at the present time a human instructor is more likely to make use of competency assertions to tailor instruction, but as algorithms improve, STEs will also make use of competency assertions to personalize instruction without reliance on human input.

3.1 Competency Profiles

In its simplest form, a competency profile is a list of competencies that are held, with the implicit assumption that if a competency is not on the list, its status is unknown. In its more complex form, a competency profile includes for each competency in one or more frameworks an estimate of whether the competency is possessed or not possessed, or an indication that the profile contains no information on that competency. In its most complex form, a competency profile may also contain links to assertions about each competency so that a system consuming the profile may re-interpret the assertions to draw its own conclusions.

As stated in Section 1.1, it is a goal of the work in this paper to enable multiple systems to share competency profiles, including the ability for each system to make assertions about the same competencies. This raises the issue of how these various assertions are collated and used to form a single profile, which we discuss next.

3.2 **Binary Assertion Processing**

Assertion processing is the term we use for examining a set of assertions about competencies in a framework **F** and using them to create or update a competency profile for all the competencies in **F**. In this paper, we present several ways to do this. Before doing so, we point out that although the general form of assertions includes confidence estimates, real-world assertions are usually binary, i.e. either state that person has or does not have a competency, and within most training systems, all such assertions are considered to be valid and equal in weight. Thus, for most practical purposes, we can assume that assertions take the simplified form of "**agent** asserts at a given **time** that **entity has** (**or does not have**) competency **C**." We call this binary assertion processing, which is where we will start.

The first and simplest method of binary assertion processing is to consider each competency **C** in **F** in isolation. We then allow each **C** to be held, not held, or unknown, and to apply the rules:

- If there are no assertions about **C,** then its status is unknown.
- If all assertions about **C** say that **C** is held, then it is held.
- If all assertions about **C** say it is not held, then it is not held.
- If there is contradictory evidence, then a *conflict resolution rule* must be applied. Justifiable conflict resolution rules include: (1) If there is a conflict the status is indeterminate. (2) The status is the determined by the most recent assertion. (3) **C** is held if *any* (non-expired) assertion says **C** is held.

Other conflict resolution rules are possible, e.g. one could declare **C** to be held if at least 60% of the assertions about **C** said it was held, but the ones given above seem to be the obvious ones that correspond to different approaches to determining competency. In (1), any conflicting evidence causes doubt. In (2) we consider the latest assessment or evaluation to be the most accurate. This assumes that assertions have time stamps In (3) we generously assume it is enough to demonstrate competency once that we stop examining assertions once that happens.

3.3 Using Relationships in Assertion Processing

The above considers each competency in isolation, but in many frameworks inferences are available based on relationships, e.g. if a person cannot operate the navigation system, they can't fly the aircraft, and if **A** is equivalent to **B**, they should either both be held or both not be held. As discussed in Section 2.4., we only consider the broadening relationship within a framework **F** and can assume that **F** is a DAG with respect to this relationship. With this structure, we can assign a status to a competency **C** by considering the DAG consisting of all descendants and ancestors of **C** and working from the furthest away towards , i.e. traversing the graph depth first.

- Given a competency **C**, consider all competencies **D** with a path from **C** to **D** and select those whose path length is maximal among all such **D.** (If the descendants of **C** form a tree, these **D** are the leaves of the tree.) Similarly, consider all the competencies **A** with a path from **A** to **C** and select those that have maximal distance, i.e. the furthest ancestors of **C.**
- Collect all assertions about each **A** and **D** and use them to determine the status of each **A** and **D**. If there are conflicts, use whatever method is preferred.
- If **B** is a parent of a descendent **D** of **C** (i.e. is between **C** and **D)**, and the status of **D** is "not held," then consider this state to be an assertion that **B** is not held based on the evidence that **D** is not held. If timestamps are involved, the timestamp for this assertion is the time of the latest assertion made about **D.**
- If **B** is a descendent of an ancestor **A** of **C** (i.e. is between **A** and **C)**, and the status of **A** is "held," then consider this state to be an assertion that **B** is held based on the evidence that **A** is held.
- Remove all the furthest ancestors and descendants from **F** and repeat until **C** has no ancestors or descendants, at which point the status of **C** will have been determined.

Example: As an example, in the DAG shown in Figure 2 and repeated to the right in Figure 3, the process described starts with gathering assertions about **E** and **T**. If the only assertion about **E** is that **E** is not held, then **E** acquires this status, and the "fact" that **E** is not held is considered evidence that **A** is not held. If a different assertion said that **A** was held, then the conflict would have to be resolved using a method from Section 3.2. In any case, once all assertions about **E** and **T** were considered, and once the status of **E** and **T** were turned into assertions about **A**, **B**, and **D**, then **E** and **T** would be removed from the graph and assertions about **A**, **B**, and **D** would be considered. Once these were processed, all the assertions about **C** would be considered and the status of **C** would be set. Although there are no parents of **C** in the graph, if there were, they would be processed top down until **C** was reached.

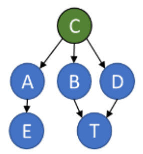


Figure 3

The algorithm described above is deterministic, logical, and easily implemented. If one resolves conflicts by using the latest assertions only, then no competency will have an unknown status as long as at least one positive assertion is made about it or one of its ancestors or one negative assertion is made about one of its descendants. From a computational perspective, however, the status of each competency in a framework must be updated separately. This means that when a new assertion is made about a competency **A** in **F**, the status of every competency connected to **A** must be re-computed. Moreover, it has the drawback that it treats all assertions as equally valid, including inferred assertions derived from the structure of **F**. This latter is questionable, since in practice the fact that **A>B** may be relatively weak evidence that if **B** is not held, then **A** is not held. For example, **A** may be a TLO with multiple ELOs, not all of which are required for **A.**

One way of addressing the last issue above is to use rollup rules instead of inferring assertions from relationships, and if such inferences are to be included for some pairs of competencies, to convert them into explicit rollup rules. In this approach, each **C** generates a directed connected DAG with **C** at its root consisting of all competencies referenced in rollup rules for **C** (the descendants of **C**), all similarly defined descendants of those descendants, and so on. This DAG is then processed depth first (from the furthest descendants upwards) using rollup rules only.

Rollup rules are more flexible and can model real-world relationships where it is possible that a competency is conferred by mastering a set of narrower competencies. For example, being a competency software developer might require proficiency in at least three programming languages. However, it may not be necessary to know any specific language, nor would proficiency in just one language be sufficient for demonstrating the overall software developer competency. Rollup rules an handle this type of complexity.

3.4 A Bayesian View

Regardless of whether binary processing uses relationships or rollup rules, it has the annoying property that the number of computations required for updates is quadratic in the number of nodes in a connected framework, i.e. if there are *n* nodes, a single new assertion requires updating the status of all nodes, and updating the status of any one node requires computing the status of all nodes based on the relationship or rollup rule structure. A bigger issue, though, is that it does not take into account differences in the strengths and validity of different assertions about the same competency and, more generally, does not model the way in which competency profiles are actually determined by live instructors and tutors. For example, at the start of a class or training session, the instructor likely has a preconceived notion of what the students know and can do, based on assumptions about their preparation and other contextual information. As the instructor interacts, observes, and evaluates each student, these beliefs are updated based on the student's actions and evaluation results. AI-based systems behave similarly, so a more realistic model is a Bayesian model in which the competency profile of an individual or team with respect to a framework is considered to be a set of estimates of the likelihood that each competency is held. The relationships within the framework create influences, rather than black and white inferences, and the process of assertion processing is that of updating the estimates based on new evidence. As of the writing of this paper, we have made good progress implementing this approach, but it was not used in the work we report in the next sections.

3.5 Relation to Human Performance Markup Language

The problem we wish to address is that of sharing competencies and outcomes/performance data among multiple systems in a training environment. The approach we have described is based on a common understanding of competencies and the ability of each system to make its own competency assertions. A different and parallel approach is to assume that each system is producing a data stream and to relate this data stream as (or after) it is reported directly to performance. This can be done using Human Performance Markup Language (HPML), which is in the process of being standardized by the Simulation Interoperability Standards Organization (SISO, 2016). We see HPML and the approaches in this paper as being synergistic and easily combined, although we have not done so to date. In our vision, HPML codes how data streams from simulations are translated into performance while xAPI is used to report less granular activities and results, and both are used as evidence of competency in competency assertions.

4. THE COMPETENCIES AND SKILLS SYSTEM (CASS)

We now switch to describing CASS, which is an open source software package whose development is being supported by the ADL Initiative and that provides infrastructure for storing, managing, and sharing both competency frameworks and competency profiles. Code, documentation, and examples are available from the CASS web site and CASS GitHub site (CASS, 2017). The services that CASS provides include:

- A repository of competency frameworks in which each competency and each framework has a persistent unique identifier that can be referenced by any training system;
- APIs for performing create, retrieve, update, and delete (CRUD) operations on competencies and competency frameworks, including on the most commonly occurring relationships;
- APIs for importing and exporting competency frameworks expressed in common standardized formats;
- A method for aligning resources to competencies and for building competency assertions by combining these alignments with xAPI data;
- A repository for storing competency assertions and competency profiles, with care taken to enforce privacy and security policies; and
- APIs for enabling "assertion providers" to perform authorized CRUD operations on assertions and for enabling systems to retrieve authorized portions of an individual or other entity's competency profile;
- Algorithms for assertion processing, used to generate competency profiles.

This section provides some more details about CASS, with further details available from the referenced web site.

4.1 CASS Architecture

CASS is written in a combination of Java for server-side processing and JavaScript for client-side modules. It is based on linked data principles (Berners-Lee, 2009), e.g. every object in CASS is referenceable by a Unique Resource Identifier (URI) and can be retrieved using a Unique Record Locator (URL). Accessing the URL provides JavaScript Object Notation – Linked Data (JSON-LD) representations of objects that refer to each other by their URLs. Wherever possible, data is represented using existing standardized schemas, in particular, those published by or proposed to Schema.org (Schema, 2017). Competencies, frameworks, Schema.org "alignment objects," competency assertions, relations, and all other objects are stored in searchable databases powered by Elastic Search (Elastic, 2017). CRUD operations are implemented over HTTP(S) using standard actions (GET, POST, DELETE, etc.). CASS implements robust encryption and security to ensure the protection of Personally Identifiable Information (PII), to enforce permissions, and to allow a wide variety of privacy policies to be implemented, ranging from "open data" to highly permissioned and controlled access.

4.2 Frameworks and Competencies CASS

CASS stores frameworks, competencies, and assertions in its own extensible representation, but was designed to import and export frameworks and competencies using common standardized formats. As a result, CASS frameworks and competencies support the properties that are common across multiple (although not necessarily all) standards. When CASS imports frameworks from external source, they become shareable and referenceable, and a "canonical URL" is included as a link to the original source. CASS competencies also have a "scope" property that is the equivalent of the condition in Mager's three-part definition of an LO (Mager, 1997). "Scope" may be used to tie a competency to a particular geography, jurisdiction, etc., but its original intent was to specify the circumstances under which a competency applied, e.g. *underwater* or *when facing enemy fire*. More information on the CASS object model can be found in CASS documentation (CASS, 2017).

The goal of creating shared understanding of competencies among collaborating systems is supported in two ways: through CRUD APIs that can be used to access and reference frameworks and competencies and through the ability to import and export frameworks in standardized formats. As of this writing, CASS supports import/export of Achievement Standards Network "standards documents" (ASN, 2017), MedBiquitous competency frameworks (in XML format) (MedBiquitous, 2017), and in spreadsheet format, where the user must map the columns of the spreadsheet to appropriate properties. Other standards used to define CASS properties and with which CASS is compatible include Reusable Definitions for Competencies and Educational Objectives (IMS Global, 2010), Common Education Data Standards (CEDS, 2017), and InLOC (CETIS, 2017) and, more recently, the IMS Global Learning Consortium's Competencies & Academic Standards Exchange (CASE) formats (IMS Global, 2017).

4.3 Assertions in CASS and the Profile API

Assertions in CASS have the properties defined in Section 3 (page 5). Evidence is a general field that can be anything from text to a URL or an object serialized as is done in email attachments. Agents and entities are represented internally by opaque IDs generated using Public Key Infrastructure (PKI) so that CASS data cannot be used to directly determine the identity of the subject or agent in a competency assertion. CASS relies on external identity servers or identity management systems to map these to "real world" identities.

Assertions can be entered into CASS in several ways. Subject to configuration, CASS supports the ability for an authorized user to make assertions about his or her own competencies. Assertions can also be directly written into CASS by a trusted and authorized external system through a CRUD API. Finally, assertions can be entered in CASS by correlating resource alignments with xAPI statements, as is described in the next section.

Once in CASS, assertions are used to build competency profiles. A *profile API* can be used by authorized systems to retrieve the competency profile of an entity. The profile API can provide the status of a single competency or all competencies in a designated framework, where the status is determined by an assertion processing algorithm, see Sections 3.2 – 3.4. The current version of CASS supports binary assertion processing only, with Bayesian processing planned for future releases.

4.4 CASS and xAPI

A frequently asked question about xAPI and CASS, both of which are part of the larger ADL's Total Learning Architecture (TLA) (ADL, 2017), is *how do learning activities report competencies using xAPI?* In this regard, it is important to understand that xAPI is designed to report about learning *experiences* and *activities*, i.e. in what activities a learner participated, what actions they took, and what scores they achieved. These are all factual data determined by the activity doing the reporting. The mastery of a competency, on the other hand, is an *assertion* that is an interpretation of the facts, and interpretation is beyond the scope of xAPI. In many real-world situations, different agents may well interpret the same results differently in different contexts, e.g. an excellent 1500-meter time for a decathlete may not even be competitive for a middle-distance runner. As a result, there are no official verbs in standard xAPI vocabularies for reporting competencies, and there should not be.

To determine competency based on learner actions reported to a Learner Record Store (LRS) via xAPI, CASS needs additional data. This additional data is the relation between the resource with which the learner engages and competencies in CASS, and it is stored in the form of an alignment object that says that a resource teaches or assesses an identified competency. The calculus is that an *alignment object + a result = a competency assertion.* For example, suppose that a virtual exercise assesses navigational skills. This is expressed as an "assesses alignment" between the exercise and a navigation competency. Alignment of this sort can be stored in CASS or retrieved by CASS from external sources. Suppose further that a learner completes the exercise and the exercise reports this to an LRS with the verb "passed." When properly configured, CASS will retrieve this statement from the LRS, put it together with the alignment, and generate an assertion to the effect that the exercise (the agent) asserts that the learner (the entity) has the navigation competency (the competency).

This approach to xAPI-based assertion generation has many advantages. The use of the alignment object enables organizations to express alignments between resources they do not own and cannot edit and competencies they do not own and cannot edit. An instructional systems designer (ISD), for example, could determine that success on an LVC mission demonstrates the ability to fly an F-18 without requiring the ISD to touch the LVC system and without requiring the ISD to have any control over the competency framework in which the F-18 piloting competency lives. The architecture, wherein CASS retrieves xAPI statements from an LRS, facilitates STEs in which multiple learning and training systems generate xAPI statements with the same LRS as a target and also enables CASS to poll the LRS and update learner profiles when it chooses. In Section 5 we discuss a trial run where this approach was implemented.

5. A TRAIL RUN

To demonstrate the potential of sharing competencies and competency profiles across multiple activities, and to test a set of systems working together using proposed standard APIs, we participated in a trial run at Fort Bragg, NC April 18th through 21st 2017. The test and demonstration included 79 learners with varying backgrounds in cybersecurity divided into three groups. The development team selected the topic area of cybersecurity because it was relevant to the career trajectories of the subjects and because there was a ready-made NICE framework (Newhouse, et. al., 2016). The portion of the NICE framework used addressed six TLOs (highest level nodes) with a total of 59 competencies. However, the actual content only addressed two of the TLOs, "Social Engineering," and "Cyber Apprentice."

The trial run implemented an early version of the TLA. The TLA is a set of Internet and software specifications (e.g. learning data model standards) under development by the ADL Initiative that is intended to enable personalized, data-driven, lifelong technology-enabled learning. Alpha versions of TLA specifications were used, including CASS APIs.

5.1 Test and Demonstration

The technologies used at Fort Bragg roughly fit into two categories: Back-end systems that implement TLA services, and cybersecurity activity providers that provided learning experiences to end users. Supporting systems included a CASS installation as described in Section 4, an LRS, and an activity index with metadata describing learning activities and their alignment to competencies. Subjects could interact with ten different applications representing a range of content types (LMS course, eBook, online videos, games) displayed on three different devices (desktop computer, tablet, mobile device) and with a *recommender engine* and dashboard used to launch activities and provide status indicators on learner progress.

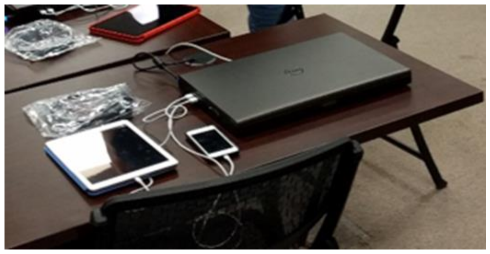
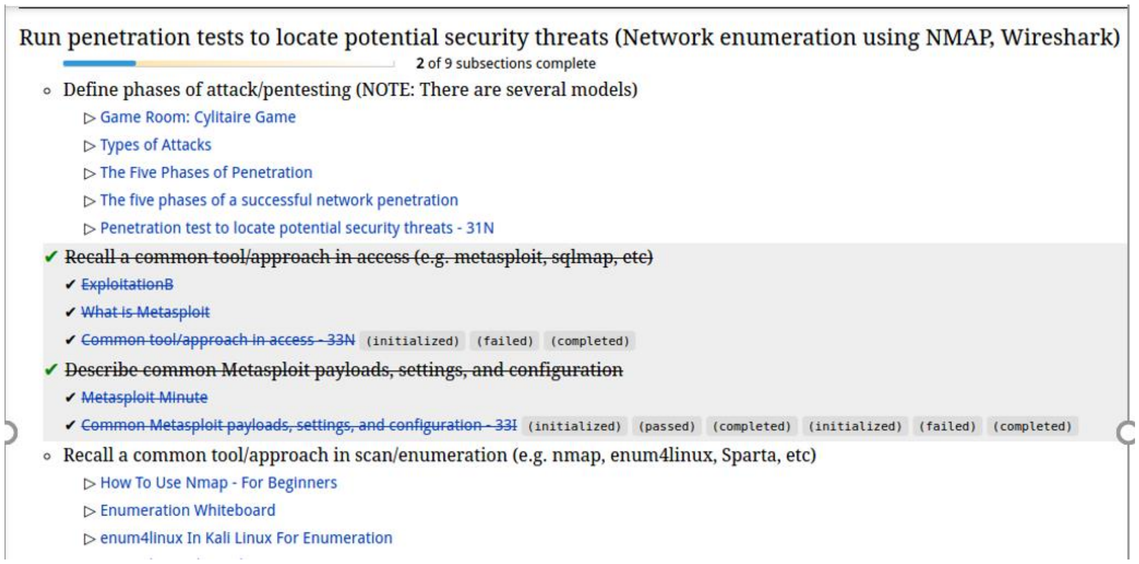


Figure 4: Devices used at Fort Bragg

In the experiment, subjects were first asked to choose whether they wanted to be a social engineer or a cyber apprentice. They were then given general instructions, which included operation of an ADL-funded recommender system that ran on an iPod (see Figure 4). The recommender system displayed "cards" with instructions as to how to engage in an activity. They could also use a dashboard that used data from CASS and an LRS to show competencies and associate modules (Figure 5) to launch activities.


Run penetration tests to locate potential security threats (Network enumeration using NMAP, Wireshark)

2 of 9 subsections complete

- Define phases of attack/pentesting (NOTE: There are several models)
  - ▷ Game Room: Cylitaire Game
  - ▷ Types of Attacks
  - ▷ The Five Phases of Penetration
  - ▷ The five phases of a successful network penetration
  - ▷ Penetration test to locate potential security threats - 31N
- ✓ ~~Recall a common tool/approach in access (e.g. metasploit, sqlmap, etc)~~
  - ✓ ~~ExploitationB~~
  - ✓ ~~What is Metasploit~~
  - ✓ ~~Common tool/approach in access - 33N~~ (initialized) (failed) (completed)
- ✓ ~~Describe common Metasploit payloads, settings, and configuration~~
  - ✓ ~~Metasploit Minute~~
  - ✓ ~~Common Metasploit payloads, settings, and configuration - 33I~~ (initialized) (passed) (completed) (initialized) (failed) (completed)
- Recall a common tool/approach in scan/enumeration (e.g. nmap, enum4linux, Sparta, etc)
  - ▷ How To Use Nmap - For Beginners
  - ▷ Enumeration Whiteboard
  - ▷ enum4linux In Kali Linux For Enumeration

Figure 5: Dashboard showing competencies and associated activities

Most of the activities were delivered through a desktop, although social engineering students also interacted with an eBook that used data from CASS to adapt the content it displayed and that included the ability to post comments about sections of the eBook. Assessments came from multiple sources, including from interactions with a cyber range provided by Sandia National Laboratories, but the most prevalent assessments used were based on concept maps. These assessments required subjects to complete or label concept maps. The maps were developed using the NICE framework as a starting point, and the assessment application reported results to the LRS using xAPI.

5.2 Data Flow

The following process illustrates how an activity provider received the status of a competency for a specific learner during the test and demonstration:

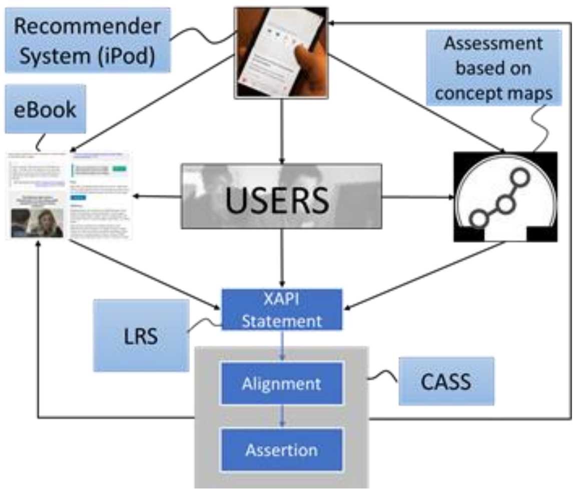


Figure 6: Setup for Social Engineering

1. Activity **X** sends xAPI statements about a learner's interactions to an LRS, where they are stored.
2. Later, Activity **Y** makes a request to CASS inquiring, "Does the learner hold competency **C**?"
3. CASS makes a request to the activity index for activities aligned with **C** and retrieves a list of such activities.
4. CASS queries the LRS for xAPI statements about the aligned activities for the specific learner.
5. After receiving evidence from the LRS, CASS creates an assertion as described in detail in Section 3.
6. CASS returns the assertion result and supporting data to Activity **Y**.

This process allows activities to provide experiential data to an LRS and other activities or applications to access the resulting assertions of competence. In this case, CASS used binary assertion processing described in Section 3.2 and a simple profile of xAPI data that served as evidence (e.g. completions, passes, fails). The assertion processing algorithm for the trial run used the rule that *if an xAPI statement indicating the user completed an activity was found, then competencies associated with that activity were considered satisfied*.

Example: As an example, assume the concept map assessment tool (Activity X above) assessed a learner and generated indicating a gap in a *penetration testing* cyber competency. In this case, the learner did not complete several *penetration testing* concept map assessments so they had incomplete xAPI records in the LRS. As the learner used the iPod recommender application to guide them through content, the recommender application (Activity Y above) made an inquiry (detailed in the process above) resulting in an assertion from CASS that they did not hold the *penetration testing* competency. As a result, the recommender application recommends an e-book on *penetration testing* to fill the gap asserted by CASS.

5.3 CASS Performance

Over a four-day period, the learners interacted with the activity providers and generated over a million xAPI statements. The subjects were given a pretest, with results being recorded in CASS as competency assertions prior to the start of the training sessions. On the average, subjects obtained three positive competency assertions via the pretest, and about four more as a result of the actual training. CASS responded to an average of 30 requests for competency status per minute, with a peak load of 65 requests per minute. Prior to running the trial, we had agreed that the time for CASS to respond to a request for a user profile had to be under10 seconds. We did not want users to wait longer than that for a recommendation, and we knew that additional latency would be introduced by waiting for API responses by the time required for the recommendation engine to process the profile. As configured, CASS computed the value of all 59 competencies each time a profile was requested. This took an average of 1.14 seconds, with 75% being completed in under 1.36 seconds and 99% in under 4.12 seconds, and the longest was, in fact, 10 seconds.

5.4 Overall System Performance

In deployments such as the one described here, the user experience can be affected by all system components and all of the TLA APIs, many of which used representational state transfer (REST) (Fielding, 2017). RESTful web services require a service *consumer* to make a request over a network and a service *provider* to respond to the request, i.e. the *consumer* pulls data from the *provider*. Although REST implementations are common at scale and are generally thought of as easy to scale horizontally, some components in the test and demonstration required near-real-time data. In the data flow described in Section 5.1, four RESTful requests were made to respond to a single request. This alone adds considerable latency, which we estimate to be about 2 seconds (0.5 seconds for each call and response).

Another factor that affected system performance was polling. To obtain near-real-time data, some components chose to poll systems such as the LRS for data at regular intervals. Each day, we observed a near 30% increase in CPU usage from the previous day, which turned out to be related to the increasing quantity of data stored each day. As more data was available, the processing required to poll the data increased. In our limited test, we neared 95% CPU usage the last days on an Amazon Web Services (AWS) t2.xlarge instance with 16Gb of memory (Amazon, 2017).

5.5 Ensuring the Quality of Data

In the trial run, xAPI statements served as the granular evidence used for competency assertions. The xAPI includes a highly flexible data model that is meant be *profiled* by constraining the data elements, verbs, and ways in which the data elements are applied for each specific environment, domain, or use case. In our case, a simple xAPI profile was used to collect *completions*, *passes*, and *fails*. In some cases, activity providers tracked additional data via xAPI, but this was largely activity provider-specific or not supported consistently by applications. Using these as part of the assertion construction mechanism resulted in inconsistent, incomplete, and (in some cases) inaccurate data, which we overcame by restricting ourselves to statements conforming to the stated profile. Another issue we discovered was that activity providers identified users using slight variations on their names. Usernames were anonymized strings containing a sequential number (e.g. user1). However, applications did not represent these consistently: Some capitalized names; some put spaces before the number, and some had no spaces. This caused problems for our analytics dashboard and when analyzing data using software such as map-reduce (Dean and Ghemawat, 2004) and had to be addressed to avoid issues with competency assertions.

6. CONCLUSIONS AND LESSONS LEARNED

The CASS team has, over the past year, worked increasingly closely with organizations such as the Credential Engine, the Learning Resource Metadata Initiative, the eXtension Foundation, CEDS, the IMS Global Learning Consortium, and others who are supporting or implementing various forms of competency-based education and training. In several instances, CASS is being used to store and manage competency frameworks. The trial run reported here is the first time the assertion processing system and profile APIs in CASS were tested. In this test, we demonstrated that it is feasible to create shared competency profiles and use them to support complex, multi-component training systems, however several things that are worth noting:

- Modeling the NICE framework and, even more so, of aligning resources to competencies took much more time than anticipated. This was done manually, in spreadsheets by the test and demonstration team. There is a clear need for automated tools to assist the processes of translating paper-based frameworks into machine-readable formats and aligning resources with competencies.
- We optimized binary assertion processing to perform in an acceptable amount of time, but we cannot expect this to scale to frameworks with hundreds of competencies because of the re-computation involved. The Bayesian approach discussed in 3.4 is the most likely way forward, and although optimization will still be required, there are techniques that allow Bayesian networks to be updated efficiently.
- We encountered inconsistent xAPI statements. When sharing outcomes or performance across systems in an STE, a receiving system (e.g. a recommender engine, a training activity, or an analytics dashboard) will assume that the assertions being made are correct and will not examine the underlying data. To avoid compounding errors, it necessary that the underlying data be clean and consistent, which emphasizes the need for standardizing xAPI verbs and statement formats.

Finally, the "million-dollar question" we wish to answer is how the ability to share competencies and profiles leads to improved learning outcomes and faster time to performance. In future years, we plan to gather data that addressees this question, but the data collected at Fort Bragg was intended to inform the design of the TLAs and test system interoperability and not to study learning effects. As such, we do not feel it is appropriate to draw any conclusions about training effectiveness from it.

7. ACKNOWLEDGEMENTS

We are grateful to the US Advanced Distributed Learning initiative for the support it has provided and to the many persons who have contributed code and perspectives to the CASS project described in this paper. Special thanks go to Fritz Ray, Damon Regan, Shane Gallagher, Stuart Sutton, and to our "bird dog" Paula Durlach, whose comments greatly improved this paper.

8. REFERENCES

ADL. (2017). Total Learning Architecture. Retrieved June 1, 2107, from https://www.adlnet.gov/tla

Amazon. (2017). Amazon EC2 Instance Types. Retrieved June 3, 2017 from https://aws.amazon.com/ec2/instance-types.

ASN. (2017). Achievement Standards Network Technical Documentation. Retrieved June 1, 2017 from http://www.achievementstandards.org/content/technical-documentation.

Berners-Lee, T. (2009). Linked Data. Retrieved June 1, 2017 from https://www.w3.org/DesignIssues/LinkedData.html.

California. (2013). California Common Core State Standards. Mathematics. Electronic Edition. Retrieved June 1, 2017 from http://www.cde.ca.gov/be/st/ss/documents/ccssmathstandardaug2013.PDF.

CASS. (2017). CASS project web site, code repository, documentation, and schema definitions. Retrieved June 1, 2017 from www.cassproject.org, https://github.com/cassproject/CASS, http://docs.cassproject.org, and http://schema.cassproject.org/

CEDS. (2017). Common Education Data Standards. Retrieved June 1, 2017, from https://ceds.ed.gov/

CETIS. (2017). InLOC (Integrating Learning Outcomes and Competencies). Retrieved June 1, 2017, from http://www.cetis.org.uk/inloc/Home

Chouhan, V. S., & Srivastava, S. (2014). Understanding competencies and competency modeling—A literature survey. *IOSR Journal of Business and Management (IOSR-JBM)*, *16*(1), 14-22.

Dean, J., Ghemawat, S. (2004). *MapReduce: Simplified Data Processing on Large Clusters*. Retrieved June 3 ,2017, from http://static.googleusercontent.com/media/research.google.com/es/us/archive/mapreduce-osdi04.pdf

Elastic. (2017). Elastic Search GitHub Repository. Retrieved June 1, 2017, from https://github.com/elastic/elasticsearch

Fielding, Roy. (2000). *Architectural Styles and the Design of Network-Based Software Architectures*. Retrieved June 3, 2017, from http://jpkc.fudan.edu.cn/picture/article/216/35/4b/22598d594e3d93239700ce79bce1/7ed3ec2a-03c2-49cb-8bf8-5a90ea42f523.pdf

IMS Global. (2017). Competencies & Academic Standards Exchangen. Retrieved June 1, 2017, from https://www.imsglobal.org/case.

IMS Global. (2010). IMS Reusable Definition of Competency or Educational Objective Specification. Retrieved June 1, 2017, from https://www.imsglobal.org/competencies/index.html.

Johnston, J. H., Goodwin, G., Moss, J., Sottilare, R., Ososky, S., Cruz, D., & Graesser, A. (2015). *Effectiveness Evaluation Tools and Methods for Adaptive Training and Education in Support of the US Army Learning Model: Research Outline* (No. ARL-SR-0333). US Army Research Laboratory.

McLeod, S. (2012). Zone of Proximal Development. In *Simple Psychology*. Retrieved June 12, 2017, from https://www.simplypsychology.org/Zone-of-Proximal-Development.html.

Mager, R. F. (1997). *Preparing instructional objectives*. 3rd Edition. CEP Press, Atlanta, GA.

Medbiquitous. (2017). MedBiquitous XML Schemas. Retrieved June 1, 2017, from http://ns.medbiq.org/#competencies

Newhouse, W., Keith, S., Scribner, B., Witte, G. (2016). NICE Cybersecurity Workforce Framework (NCWF). *Draft NIST Special Publication 800-181*. National Initiative for Cybersecurity Education. Retrieved June 1, 2017 from http://csrc.nist.gov/publications/drafts/800-181/sp800_181_draft.pdf.

OPM. Performance Management. Retrieved June 12, 2017 from https://www.opm.gov/policy-data-oversight/performance-management/performance-management-cycle/planning/developing-performance-standards/

Schema. (2017). Schema.org web site. Retrieved June 1, 2017, from http://schema.org/.

SISO. (2016). Nomination for Human Performance Markup Language Product Version 1.0. Retrieved June 1, 2017, via link available at https://www.sisostds.org/StandardsActivities/DevelopmentGroups/HPMLPDG-HumanPerformanceMarkupLanguage.aspx

Virginia. (2016). Mathematics Standards of Learning for Virginia Public Schools. Retrieved June 1, 2017, from http://www.doe.virginia.gov/testing/sol/standards_docs/mathematics/2016/stds/stds-grade3.docx.

W3C (2009). SKOS Simple Knowledge Organization System Reference. Retrieved June 1, 2017, from https://www.w3.org/TR/skos-reference/.