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 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 different learning environments 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.

This paper ends with a description of a system that implements the suggested solution and its application in the context of live trials with 73 subjects run as part of a design-based research effort.

ABOUT THE AUTHORS

Dr. Robby Robson is a researcher and innovator who has contributed to numerous technologies and standards widely used 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 Runtime 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 other multi-component training environments (Johnston, et. al., 2015), learners often engage with 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 in a complex multi-player exercise that includes LVC components. As learners engage with these various components, evidence is produced about the competencies that the learners possess or do not possess, where "competencies" are skills, knowledge, abilities, or attitudes related to a task or job (Chouhan & Srivastava, 2014). The evidence can range from direct assertions of competence, e.g., made by an instructor, to test results that relate to specific competencies or associated learning objectives (LOs) to raw data that can be interpreted as evidence of mastery.

1.1 The Goal of this Work

Ideally, a training system should not engage a trainee in an activity for which the trainee does not have the necessary prerequisites or that teaches competencies the trainee has already mastered. To properly sequence training activities, it is thus necessary to track the relevant competencies that a trainee does or does not hold. A list of such competencies is called a *competency profile*.

Many training systems maintain some form of competency profile: A SCORM-based learning management system (LMS) can store objectives reported by a course, intelligent tutors maintain learner models that include competencies, and serious games and simulations maintain information on players' levels and achievements. These, however, are scoped to a single 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.*

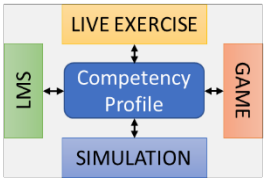

Figure 1: Shared Profile

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 and the scenario used 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. This approach was sponsored by the US Advanced Distributed Learning (ADL) Initiative and is based on *competency frameworks*, explained in Section 2, and a method of processing *competency assertions* from multiple sources, discussed in Section 3. An open source implementation is described in Section 4, and a design-based research study with 73 participants is discussed in Section 5. Section 6 discusses lessons learned and next steps.

2 COMPETENCIES AND COMPETENCY FRAMEWORKS

Competencies can 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

It is often the case that two competencies are related, e.g., one competency is part of or a prerequisite for another, or 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.
- 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.

For our purposes, the most important aspect of relationships among competencies is their effect on the determination of mastery. Whether one competency requires or is "contains" another competency, if a learner has not mastered the second there is an inference that the learner has not mastered the first**.** Borrowing from the W3C Simple Knowledge Organization System (SKOS) (W3C, 2009) and the Medbiquitous standard (Medbqituitous, 2017), we introduce a single pair of inverse relationships, "broader" and "narrower." "Broader" means more general, and "narrower" means related but not more general (W3C, 2009, Section 8.1). We consider relationships such as *requires*, *contains*, and *is enabled by* to be cases of the "broadens."

2.2 Competency Levels

Several competency standards include the notion of a *level*. In this paper, we restrict the notion of level to measurable performance levels, i.e., performance levels that can be assessed by the results of some task or evaluation.

2.3 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.

2.4 The Structure of Frameworks

Real-world frameworks often have hierarchical structures, sometimes with well-defined relationships and sometimes with vaguely defined parent/child relations. As explained in Section 2.1, we assume that the relationships defined for any given frameworks can be used identify broader and narrower competencies and that no other relationship is

needed. We can then make each framework into a directed graph by defining the nodes to be the competencies and adding a directed edge from each competency to each of its narrower competencies. All non-contrived frameworks that we have encountered turn out to be 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 some that are not. For example, writing skills may be required for multiple tasks in a framework and are therefore children of multiple competencies.

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. In the DAG, **A**, **B**, and **D** are the children of **C**. **E** is a descendant of **C** but not a child of **C** and **E** and **C** is a parent of **A** and an ancestor but not a parent of **T**.

Figure 2: Portion of the NICE Framework

Another type of relationship is an equivalence relationship. Equivalence occurs naturally across different frameworks (e.g. across state science standards), and can occur within a single framework when multiple authors add the same competency with different names. In this latter 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 any competency in the cycle is held if and only if all others are held.

2.5 Rollup Rules

A concept not included in most standards, but important in applications, is that of a *rollup rule.* A rollup rule defines how mastery of a target competency may be determined from mastery of other competencies. For example, a communication competency might be demonstrated by demonstrating either excellent verbal communication skills *or* excellent written communication skills and good verbal communication skills. Here, "excellent" and "good" are levels of verbal and written communication skills, both of which narrow a target communication competency**,** and a rollup rule says that "Excellent Verbal" or "Excellent Written and Good Verbal" implies the target competency.

3 ASSERTIONS

Competencies and frameworks are useful for designing learning experiences, 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 may be 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; the reliability of the source; the evidence presented; your own confidence in the assertion; and how time has affected the validity of the assertion. 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

Assertions entail a single competency, but applications usually want to know about all of the competencies that a learner possesses. This is called a *competency profile*. A competency profile can be just a list of competencies that are held or can be a set of estimates of the degree to which each competency in one or more frameworks is possessed. In either case, a fundamental question is how this information is determined. In other words, how does a system use a set of assertions about an individual to determine or estimate whether a competency is held? What happens when there are conflicting assertions, or when there are assertions about related competencies that have implications about mastery of a target competency? The process by which a set of assertions is collated into a profile is called *assertion processing* and is discussed next.

3.2 Binary Assertion Processing

In this paper, we present several approaches to assertion processing, but before doing so, we point out real-world assertions are usually binary, i.e. they 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 start.

The first and simplest method of binary assertion processing is to consider each competency in isolation. We then allow each competency 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 a competency to be held if at least 60% of the assertions about it said it was held, but the ones given above seem to be the ones that arise naturally. 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 two competencies are equivalent, either both or neither should be held. As discussed earlier, we can assume that each framework is a DAG with respect to a single broader/narrower relationship. With this structure, we can assign a status to a competency by considering all descendants and ancestors of a competency and working from the furthest away towards the target competency, i.e. traversing the graph depth first. For example, consider the DAG shown in Figure 2 and repeated in Figure 3. To determine

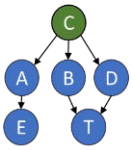

Figure 3

whether a learner holds competency **C**, we would start by 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 since **A** is broader then **E**. This would be expressed as an assertion about A, and If a different assertion said that **A** was held, then the conflict would have to be resolved using a method from Section 3.2. 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. **C** has no parents in Figure 3, but if it did, the ancestors of **C** would be processed top downwards in an analogous fashion until **C** was reached.

The algorithm illustrated above is deterministic 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, this algorithm requires that the status of each competency in a framework must be updated separately. This means that when a new assertion is made about a competency, the status of every competency connected to it must be re-computed. Moreover, it has the drawback that it treats all assertions as equally valid, including inferred assertions derived from relationships among competencies. This latter is questionable, since in practice the fact that a narrower competency (e.g. an ELO) is not held may be relatively weak evidence that a target competency (e.g. a TLO) is not held, e.g., if the TLO has multiple non-required ELOs.

One way of addressing this is to use rollup rules instead of inferring assertions from relationships. Rollup rules are more flexible and better model real-world relationships where, for example, being a competent software developer might require proficiency in at least three programming languages but not in any specific language, and proficiency in just one language would not be suffice.

3.4 A Bayesian Future

Regardless of whether binary processing uses relationships or rollup rules, it has the undesirable property that the number of computations required for updates is quadratic in the number of nodes in a connected framework. 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 does not model the way in which we believe competency profiles are determined by live instructors and tutors. We see instructors as taking a more Bayesian approach in which they start with beliefs about what the students know and can do based on past performance, general information about the students or the class, and on other contextual information. As the instructor interacts, observes, and evaluates each student, these beliefs are updated. As of the writing of this paper, we are implementing a machine learning approach that models this. In this approach, assertions are estimates of the likelihood that a competency is held and relationships create influences, rather than black and white inferences. This better reflects our picture of real-world assertion processing and overcomes many computational limitations.

4 THE COMPETENCIES AND SKILLS SYSTEM (CASS)

We now switch to describing CASS, 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.

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 designed to 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 (top of 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 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 control and competencies they do not control. 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 TRIAL 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 18 – April 21, 2017. The test and demonstration included 73 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

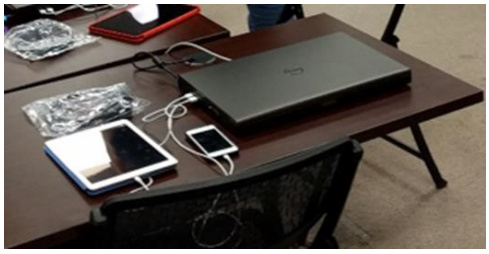

Figure 4: Devices used at Fort Bragg

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.

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.

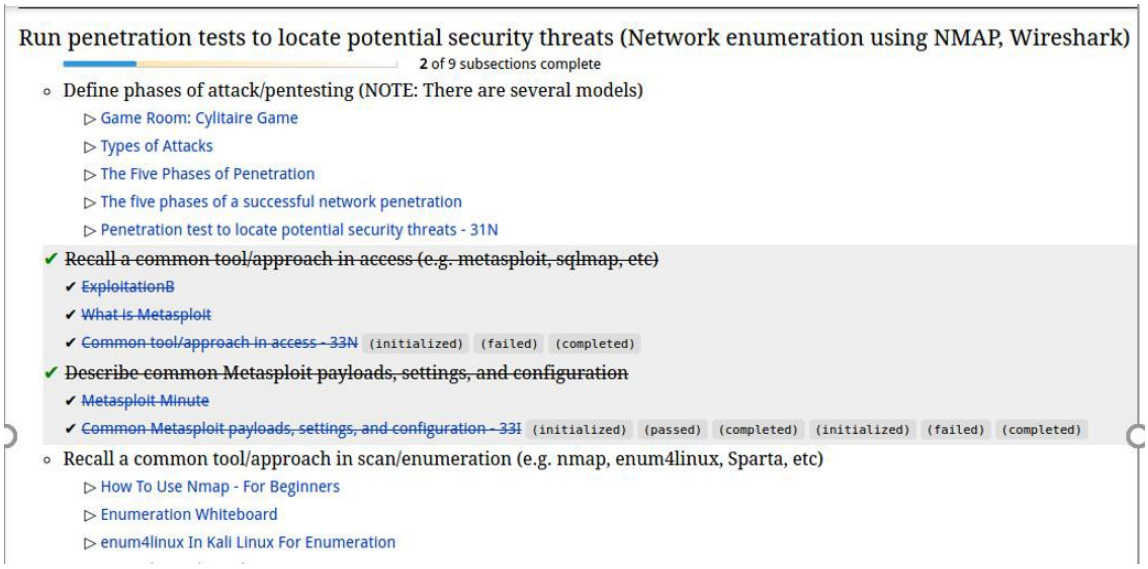

Figure 5: Dashboard showing competencies and associated activities

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.

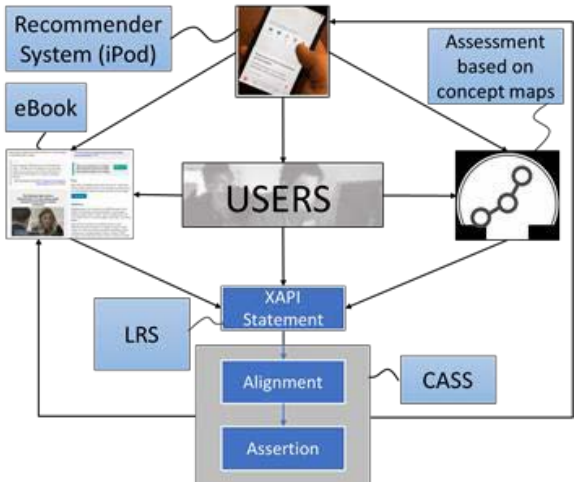

Figure 6: Setup for Social Engineering

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:

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: Suppose the concept map assessment tool (Activity X above) assessed a learner on a *penetration testing* cyber competency but the learner did not successfully complete the assessment. This generated xAPI records in the

LRS showing that the activity was not completed. As the learner used the iPod recommender application to guide them through content, the recommender application (Activity Y above) polled CASS and CASS responded with an assertion that the learner did not hold the *penetration testing* competency. The recommender application would then recommend 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 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.

Another point worth noting is the relation between the approach in this paper and Human Performance Markup Language (HPML) (SISO, 2016). HPML can be used to directly code how data produced by simulations and other learning activities are translated into performance, while xAPI reports less granular activities and results. HMPL and xAPI data both provide evidence of competency that can be encoded into competency assertions. Contemporary work describing HPML and its combination with xAPI reporting can be found in Bruno's paper (Bruno, 2017).

Finally, the "million-dollar question" is how the ability to share competencies and profiles leads to improved learning outcomes and faster time to performance. This is addressed in part in (Gallagher, 2017), but from our perspective, the data collected at Fort Bragg was collected to inform the design of the TLA and to test system interoperability, and not to study learning effects.

7 ACKNOWLEDGEMENTS

We are grateful to the US Advanced Distributed Learning Initiative for its support and to the many persons who have contributed code and perspectives to the CASS project. 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

Bruno, E. (2017). Interoperable assessments using HPML: A novice conning skills acquisition use case. I/ITSEC 2017.

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

Gallagher, S., Folsom-Kovarik, J.T., Schatz, S., Barr, A. & Turkaly, S. (2017). Total Learning Architecture Development: A Design-Based Research Approach. I/ITSEC 2017.

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 fromhttp://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/.