Supporting Flexible Competency Frameworks

Erica Melis[1], Arndt Faulhaber, Ahmad Salim Doost[2], and Carsten Ullrich[3]

[1] DFKI GmbH, 66123 Saarbruecken, Germany
[2] Saarland University, 66123 Saarbruecken, Germany
[3] Shanghai Jiao Tong University, 200030 Shanghai, China

Abstract. Since Bloom's initial work on competencies in 1956, various competency systems have been designed and used to assess students' competencies. Different pedagogical researchers and stakeholders prefer different systems. We have been collaborating with them. Such systems are essential for the adaptation by adaptive intelligent tutoring systems. Now, this paper presents how ActiveMath integrates several competency systems to bridge the gap between different competency systems and thereby facilitating the reuse of learning objects across system boundaries. The combination of competency-related data is achieved by mapping a new competency system to the internal one.

Keywords: Competency System, Student Model, Course Generation

1 The Need for Flexible Support of Competencies

In our long-term experience with metadata in eLearning and Intelligent Tutoring with ActiveMath [11], required competencies and difficulties of exercises turned out to be important not only for technical reasons but also in the interdisciplinary work with pedagogists.

Pedagogists (and psychologists) have been looking for and using a classification system at least since 1956 [3]. The psychology-driven approaches in [5, 1] and [2] are examples for later approaches most of which are empirically driven.

The large PISA-studies needed a common classification for students' skills or competencies in order to compare student results across schools, regions, and countries. It seems that, similar to Bloom's taxonomy, teachers (and hence authors) could understand and apply the PISA competencies and even levels.[4] This seems to be more difficult for the more complex taxonomy in [2].

Systems for Technology-Enhanced Learning (TEL) need a classification of competencies to automatically react to student actions at the micro-level (e.g. with feedback) and at the macro-level (e.g. with learning objects' (LOs) sequencing), and to build a student model by processing these reactions. Such a classification needs to be reusable, as reproducible as possible, and coherent in order to enable reuse of LOs. In concrete domains it should reflect an optimal granularity to support instructional effectiveness [16] and it should be based on a cognitive task analysis.

[4] Personal communication with Kristina Reiss

Now, our long-term experience shows that for pedagogists as well as for TEL-content it seems that different groups of designers and applicants are biased towards different classifications of competencies. On the one hand PISA competencies are not as detailed (and hence less confusing) as competencies based on rather detailed cognitive processes. On the other hand, they may insufficiently discriminate (the exercises/tests) for diagnoses.

Hence, TEL-platforms such as ActiveMath need to be able to use several popular competency systems in order to enable the re-use of content from different author groups and process them in different applications of the platform.

This paper reports work on competencies and on components of ActiveMath which handle competencies and even different systems of them with the ultimate goals of assessment and adaptation. The paper is organized as follows: a brief overview (Section 2) on well known and frequently used competency systems shows that TEL-platforms need to be able to represent different competency systems. We then describe how a TEL-platform can be extended to represent and employ several competency systems (Section 3). Section 4 describes related work and puts our work in context. We conclude with Section 5.

2 Competency Systems

Competency systems were developed based on the observation that solving some exercises require different skills than others. The competencies defined in a competency system provide a vocabulary (often hierarchical) to classify exercises according to the involved skills. The single *one* competency system satisfying all pedagogical viewpoints does not exist. Different systems were designed over time, for instance to capture domain-specific skills in mathematics. In this section we present several examples of competency systems represented in ActiveMath.

2.1 Bloom's Taxonomy

The first major noticeable work about a hierarchical system of competencies was published in the mid 50s, last century [3]. Benjamin Bloom advised a compelling hierarchy of educational goals. Nowadays often visualized as a pyramid (*knowledge* representing the basis and the most complex goal, *evaluation* forming the tip), the competencies consist of:

- *Knowledge:* remembering, memorizing, recognizing, recall identification, recall of information
- *Comprehension:* interpreting, translating from one medium to another, describing in one's own words, organization and selection of facts and ideas
- *Application:* problem solving, applying information to produce some results, and use of facts, rules and principles
- *Analysis:* subdividing something to show how it is put together, finding the underlying structure of a communication, identifying motives, and separation of a whole into component parts

- *Synthesis:* creating a unique, original product that may be in verbal form or may be a physical object, combination of ideas to form a new whole
- *Evaluation:* making value decisions about issues, resolving controversies or differences of opinion, and development of opinions, judgements or decisions

2.2 PISA-Based Competencies

In the late 90s, the OECD (Organisation for Economic Co-operation and Development) started the *PISA* studies (Programme for International Student Assessment), which "aim to measure how far students approaching the end of compulsory education have acquired some of the knowledge and skills essential for full participation in the knowledge society" [14].

PISA is based on the notion of *competency-based learning* [13]: learning mathematics should not only aim at solving a problem but also at thinking mathematically and arguing about the correctness or incorrectness of the problem solving steps and involved methods, to perform simple and complex computations, etc.

The competencies in LeActiveMath (implemented in the ActiveMath system) are the following (see also [12] and [7]): think mathematically (e. g.,, pose questions, understanding scope and limitation of a concept); argue mathematically (e. g., develop and assess mathematical arguments, know what a proof is and what not); solve problems mathematically (e. g., identify, pose and specify problems, monitor and reflect on problem solving); model mathematically (e. g., translate content into mathematical terms, knowing the difference between the situation and the model); use mathematical representations (e. g., understand and utilize different sorts of representation); deal with symbolic and formal elements of mathematics (e. g., use terms, equations and functions to model and interpret); communicate (e. g., explain solutions, use a special terminology); use tools and aids (know about existence of tools and their range and limitations). These top-level competencies have subcompetencies, e.g., *model* has the subcompetencies *decode* and *encode*.

2.3 Revised Bloom

Anderson et al. [1] extend Bloom's taxonomy and pair cognitive processes with knowledge to represent *competencies*. In order to provide a basis to foster a better understanding of curricula, they introduce two different dimensions (in contrast to the single dimension in Bloom's original work), the dimension of cognitive processes and the dimension of knowledge. Pairs of cognitive processes and knowledge elements form objectives (that constitute the basic building blocks of curricula). This was motivated by analyzing objectives listed in curricula, normally consisting of phrases such as "The student will learn to differentiate between rational numbers and irrational numbers" ([1], p. 5). They point out that such phrases are normally composed of a verb describing the intended cognitive process and one or more nouns referring to knowledge the students are supposed to acquire.

3 Competencies in Systems for Technology-Enhanced Learning: ActiveMath

The three-tier architecture that describes typical TEL systems consists of a domain model, a student model and a pedagogical model [18, 6]. The domain model contains domain concepts, the LOs and their respective metadata. The user or student model manages information about users. Based on observations of the student's interactions, it stores, infers and updates information about an individual student and estimates his proficiency concerning single competencies. The pedagogical model (or adaptation model) contains the knowledge on how to adapt the behavior of the system, e. g., how to present content from the domain model taking into consideration the information provided by the student model.

In the following we describe how the ActiveMath system supports multiple competency systems.

3.1 Preliminaries about ActiveMath

ActiveMath is a web-based ITS for studying mathematics. Its main components are organized as client-server applications with additional web-services. Its learning content (definitions, examples, exercises, etc.), are semantically encoded in an extended OMDoc [8]. This semantic representation is a basis for the interoperability of various web-services in the diagnosis process. It allows for semantic diagnosis using external Computer Algebra System (CAS) and domain reasoner services. A course generator in ActiveMath can organize learning objects into a book, which is specific to a student's abilities regarding specific competencies.

3.2 Domain Model

In order to allow the student model to infer proficiencies plausibly, exercises have to be enriched with metadata that relate them to items of the domain model (e.g. concepts) and to competencies needed to produce a proper solution.

For the support of multiple competency systems, the domain model of ActiveMath introduces special metadata elements for exercises. The `competency-system`-metadata in the metadata section of exercise encodings specifies which competency system is used. The `competency`-metadata defines which competencies are trained (cf. Fig. 1). It is possible to specify multiple competencies for an exercise, but all of these must belong to the one specified competency-system.

3.3 Student Model

A user model tries to capture various characteristics of a user required in a specific context. In the context of e-learning systems, the user model is called *student model.* Its most important task is the estimation of the students' level of proficiency with respect to the learning material.

```
<exercise id="Represent_Barbara">
  <metadata><extradata>
      <competency-system value="PISA" />
      <competency value="solve" />
      <competency value="tools" />
      <difficulty value="very_easy"/>
      <!-- ... further metadata ... -->
    </extradata></metadata>
  <!-- ... exercise content ... -->
</exercise>
```

Fig. 1. An exercise annotated with competencies from the *PISA* competency scheme.

In ActiveMath, whenever a student tries to solve an exercise, the exercise system informs other components – particularly the student model – about results and provides the metadata associated with the exercise, such as the trained competencies and the competency system they are based on. The student model keeps track of exercise results, inferring from them students' proficiencies relative to the competencies exercised.

At present, ActiveMath supports three different kinds of competency systems. BLOOM's original system, the PISA competencies and the ActiveMath-competency system (AMCS) [10] – an adaptation of Anderson et al.'s work [1]. Internally, the student model of ActiveMath calculates estimations for students' proficiencies using the AMCS, a fine-grained system with about 6 times as many competencies (or rather cognitive processes) as in BLOOM's system.

Other competency systems are mapped to the AMCS in order to facilitate the combination of information gathered from students' interactions that refer to different competency systems. A *translation-map* for every competency system defines weights for each available competency from the source competency system (the one used in the exercise) to the internal competency system (cf. Table 1). The assigned weights specify the amount of influence an evidence (i. e., an exercise result) has for the calculation of the student's proficiency.

Table 1. A part of the translation-map for translating from the *PISA-Based* competency system to the *revised Bloom* competency system.

PISA-Based Competency	Revised-Bloom Competency	Weight
`generalize`	`summarize`	1.0
`argue`	`critique`	0.5
	`prove`	0.5
	`attribute`	0.1
	`self-explain`	0.1

The idea is to translate non-AMCS competencies associated with an exercise to the internal competency system before processing the user input. All system components can access the mapping functionality, including the course generator and the exercise system, which employ the proficiency estimations for adaptation purposes. Currently, the competency mappings in ActiveMath have been defined manually. Though, in order to achieve optimal translations, the mappings will have to be adjusted by means of data-mining using real-life data.

In case the combination of data resulting from the use of different competency systems is not wished for, the student model can process and store evidences separately per competency system.

3.4 Pedagogical Model

Course generation offers a tradeoff between pre-authored *one-size-fits-all* courseware and individual look-up of learning objects [4]. It supports learners by an on-demand assembly of sequences of learning objects. The generations take information about learning objects, the learner and his/her learning goals into account. A course generator assembles the sequences according to a set of methods that implement pedagogical knowledge. The methods determine which learning objects the course generator includes in a course, the order of these objects, the structure (sections/subsections) of a course, etc.

Paigos, the course generator developed for ActiveMath, is able to generate courses according to seven different scenarios [19] (rehearse, connect, train intense, train competencies, exam simulation and guided tour). Competencies play an essential part in Paigos, especially for the exercise and example selection. In the previous course generation methods, competencies were hard-coded, since the scenarios were designed in project focusing on the PISA framework. In the following, we analyze how competencies come into play and we describe how to abstract to a flexible competency system. Competencies are used when 1) inserting a sequence of exercises or examples that cover all competencies within a given competency system in order to present new content in multiple contexts [17]; 2) selecting exercise or examples for a competency given as a parameter; 3) selecting an exercise or example for a specific competency.

Figure 2 contains an example of a method for case 1. Expressions starting with `?` stand for variables. `?c` is instantiated with a specific concept at run-time. The method encodes the knowledge that in order to train all competencies with respect to a concept `c`, the course generator should insert learning objects that practice the competency *think* with respect to `c`, etc.

This case is abstracted in the following way: first, when ActiveMath starts course generation, it passes the used competency system to the course generator, where it is stored as the fact `(competency-system (`$\texttt{c}_1$`,...,`$\texttt{c}_n$`))` (where $c_1,\ldots,c_n$ are competencies). This fact is reused by the methods shown in Fig. 3, which recursively walks over competencies $c_1,\ldots,c_n$ (the last method ends the recursion by matching the empty list).

Figure 4 contains methods that search for exercises of a parameterized competency. In these methods, the competency is not hard-coded, but instantiated

```
(:method (practiceAllCompetencies ?c)
         () ;; no "if" part, method always applicable
         ;; then
         ((practiceCompetency think ?c)
          (practiceCompetency solve ?c)
          (practiceCompetency represent ?c)
          (practiceCompetency language ?c)
          (practiceCompetency model ?c)
          (practiceCompetency argue ?c)
          (practiceCompetency tools ?c)
          (practiceCompetency communicate ?c)))
```

Fig. 2. Training all competencies, hard-coded.

```
(:method (practiceAllCompetencies ?c)
         ()
         (practiceAllCompetenciesH ?c (competencies ?comps)))
(:method ((practiceAllCompetenciesH ?c (competencies (?head . ?rest)))
         ()
         ((practiceCompetency ?head ?c)
          (practiceAllCompetenciesH ?c (competencies ?rest))))
(:method (ipracticeAllCompetenciesH ?c nil)
         ()
         ())
```

Fig. 3. Training all competencies, flexible.

by an earlier applied method (cf. Fig. 3). These methods do not need to be changed for a flexible competency system.

Several methods encode very specific pedagogical knowledge. E.g., the method shown in Fig. 5 specifies that in case the learner has a high competency *argue*, the course generator can assume he/she is able to find proofs or establish evidence on her own. Thus, the method does not insert an evidence but an exercise for the competency *argue*. Such specific pieces of knowledge depend on the used competency system and we cannot generalize them in all cases without loss of quality. In case a competency system can encode a semantically similar case, a pedagogical expert has to add a corresponding method.

4 Related Work

Adaptive learning technologies aim at providing a learning experience tailored to the individual student. However, adaptation can only be reasonable if the abilities of a student can be modelled into the system, i.e. the students' competencies have to be known to the system for instance by interpreting interactions with the system. The definition of a competency system (CS), hence, is a most basic

```
(:method (practiceCompetency! ?competency ?c)
         () ;; no "if" part, method always applicable
         ((trainWithSingleExercise ?c very_easy ?competency) ...
          (trainWithSingleExercise ?c very_difficult ?competency)))

(:method (trainWithSingleExercise! ?c ?difficulty ?competency)
         ;; if suitable exercise exists
         ((assign ?exercise (call GetResources
                              ((class Exercise) (relation isFor ?c)
                               (property hasDifficulty ?difficulty)
                               (property hasCompetency ?competency))))
         ((insert ?exercise))) ;; then part
```

Fig. 4. Selecting an exercise for a competency given as parameter.

```
(:method (prove! ?c)
         ;; if
         ((learnerProperty "hasCompetencyArgue" ?c ?argue)
          (>= ?argue 3)
          (assign ?exercise (call GetResources
                               ((class Exercise) (relation isFor ?c)
                                (property hasCompetency argue)))))
         ((insert ?exercise))) ;; then part
```

Fig. 5. Selecting an exercise of a specific competency.

prerequisite to the task of adaptation. A competency system should enable the system to infer knowledge/ability gaps concerning certain learning goals as well as subsequent steps in the actual learning/training session. In recent years, efforts in researching competencies have been multifold and often interdisciplinary, ranging from learning system/content specific skills [16] to approaches aiming at more abstract and general definitions of competencies [1, 2, 13]. Some researchers even include qualitative mappings between their competency system and other established CSs [15].

Various projects include the modelling of competency ontologies (often part of some domain ontology) in order to provide a rich, machine-interpretable representation of the targeted domain. Intergeo [9] for instance, an EU funded project, is “dedicated to the sharing of interactive geometry constructions”. It offers a “web platform which will [. . .] provide access to thousands of interactive geometry constructions and resources using them.”

Closely related to competencies, particularly (but not exclusively) in school or university education, are questions regarding educational standards and their inherent curricula. For centuries governments and other educational institutions have striven to standardize curricula (as the US does with their SAT tests) with increasingly international magnitude (notably the Bologna process within the EU). “The search platform should cross the boundaries of the curriculum standards of Europe. A topics and competency based approach to retrieval for interactive geometry [. . .] has been adopted: it requests the contributor of an interactive geometry resource to input the competencies and topics involved in a construction, and allows the searcher to find it by the input of competencies and topics close to them", to cite some intentions of the Intergeo project, these hint at the tight relation between curricula, competencies and intercultural standardization issues that currently concern many researchers in the TEL domain.

Since at present, E-Learning standards such as LOM[5] include metadata for competencies, but no conversion rules between different CSs, various learning platforms implementing different (usually incompatible) CSs can only make restricted use of one another's LOs with respect to user adaptation.

Notable among the work concerning the integration of different CSs is a paper of Paquette's [15]. He presents a top-level ontology for competencies, and a competency system pairing *generic skills* with *resources* (corresponding to cognitive processes and knowledge elements respectively in the AMCS). He combines and extends several previous approaches such as BLOOM's and includes mappings between his and preceding approaches, though it is not clear whether and how the mapping is implemented in an actual system.

5 Conclusion

This paper addressed a number of CSs used by instances of the adaptive TEL platform ActiveMath. It describes the competency support and the extension of modules to support a variety of CSs. In particular, ActiveMath's student model and course generation extensions are discussed. The student model can translate the elements of a number of major CSs into elements of its internally used pairs of concept and cognitive process. The student model's interface outputs a percentage describing its belief of the student's state of knowledge for each pair. The output can be translated to any modeled CS. The course generation is no longer restricted to a single CS. Instead, the one given as a parameter is used to select appropriate learning objects. In some cases, the encoded pedagogical knowledge searches for an object of a specific, hard-coded competency. These cases are not abstracted as they encode highly specialized knowledge.

This research provides guidance to TEL platform developers who are faced with the challenge to integrate content encoded using a different competency system — a challenge every platform that supports reuse of content will face.

Acknowledgement This work has been supported by the eContentPlus project Math-Bridge (ECP-2008-EDU-428046) and by China Postdoctoral Science Foundation (No. 20080430656). Only the authors are responsible for the content.

[5] http://ltsc.ieee.org/wg12

References

1. Anderson, L., Krathwohl, D., Airasian, P., Cruikshank, K., Mayer, R., Pintrich, P., Wittrock, M.: A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxomnomy of Edicational Objectives. Longman, New York (2001)
2. Assche, F.V.: Linking content to curricula by using competencies. First International Workshop on Learning Object Discovery and Exchange (2007)
3. Bloom, B. (ed.): Taxonomy of educational objectives: The classification of educational goals: Handbook I, cognitive domain. Longmans, Green, New York, (1956)
4. Brusilovsky, P., Vassileva, J.: Course sequencing techniques for large-scale web-based education. International Journal of Continuing Engineering Education and Lifelong Learning 13(1/2), 75–94 (2003)
5. Cen, H., Koedinger, K., Junker, B.: Learning factor analysis - a general method for cognitive model evaluation and improvement. In: M. Ikeda, K.A., Chan, T.W. (eds.) ITS-06. LNCS, vol. 4053, pp. 164–175. Springer-Verlag, (2006)
6. De Bra, P., Houben, G.J., Wu, H.: AHAM: a Dexter-based reference model for adaptive hypermedia. In: HYPERTEXT '99: Proceedings of the tenth ACM Conference on Hypertext and hypermedia. pp. 147–156. ACM Press (1999)
7. Klieme, E., Avenarius, H., Blum, W., D?brich, P., Gruber, H., Prenzel, M., Reiss, K., Riquarts, K., Rost, J., Tenorth, H., Vollmer, H.J.: The development of national educational standards - an expertise. Tech. rep., German Federal Ministry of Education and Research (2004)
8. Kohlhase, M.: OMDoc: An Open Markup Format for Mathematical Documents [version 1.2], LNCS, vol. 4180/2006. Springer Verlag Heidelberg (2006)
9. Libbrecht, P., Desmoulins, C.: A cross-curriculum representation for handling and searching dynamic geometry competencies. Semantic Web Technologies for e-Learning, vol. 4, pp. 136–155. IOS Press (2009)
10. Melis, E., Faulhaber, A., Eichelmann, A., Narciss, S.: Interoperable competencies characterizing learning objects in mathematics. In: Intelligent Tutoring Systems, ITS 2008. LNCS, vol. 5091, pp. 416–425. Springer, Berlin (2008)
11. Melis, E., Goguadze, G., Homik, M., Libbrecht, P., Ullrich, C., Winterstein, S.: Semantic-Aware Components and Services of ActiveMath. British Journal of Educational Technology 37(3), 405–423 (2006)
12. Niss, M.: Mathematical competencies and the learning of mathematics: the danish KOM project. Tech. rep., IMFUFA, Roskilde University (2002)
13. OECD (ed.): Measuring Student Knowledge and Skills – A New Framework for Assessment. OECD Publishing, Paris, France (1999)
14. OECD: PISA – the OECD programme for international student assessment. Brochure (2007)
15. Paquette, G.: An ontology and a software framework for competency modeling and management. Educational Technology & Society 10(3), 1–21 (July 2007)
16. Pardos, Z., Heffernan, N., Anderson, B., Heffernan, C.: The effect of model granularity on student performance prediction using bayesian networks. In: User Modeling. pp. 435–439. LNAI, Springer-Verlag (2007)
17. Reinmann-Rothmeier, G., Mandl, H.: Unterrichten und Lernumgebungen gestalten. Pädagogische Psychologie. Ein Lehrbuch, pp. 601–646. Beltz PVU. (2001)
18. Richardson, J.J., Soloway, E., Polson, M.C.: Foundations of intelligent tutoring systems. L. Erlbaum Associates, Hillsdale, N.J. (1988)
19. Ullrich, C.: Pedagogically Founded Courseware Generation for Web-Based Learning – An HTN-Planning-Based Approach Implemented in PAIGOS. No. 5260 in LNAI, Springer (2008).