# Knowledge Management of Worked-out Examples – Exploring the Educational Benefits

*Rachel Or-Bach*
*The Max Stern Academic College of Emek Yezreel, Israel*

**orbach@yvc.ac.il**

## Abstract

Worked-out examples serve an important role in the educational process. In order to take advantage of such carefully designed examples, students have to "make sense" of the examples and retrieve them whenever it is appropriate. This calls for a knowledge management process to be conducted by the students; a process we implemented as a learning task during a Knowledge-Based Systems course. Students were required to incrementally define and refine a metadata scheme for representing and managing a growing set of programming examples that were introduced during the course. Our findings show that students' iterative processes of metadata creation and modification, which were conducted both individually and collaboratively, supported reflection activities and brought students to revisit, rethink and reinterpret the various code examples and the relations between them. We discuss the finding and the implications of having students deal with issues of reusability, abstraction, metadata; and with the respective tradeoffs and dilemmas.

**Keywords**: Knowledge management, metadata creation, metadata scheme, worked-out examples, community of practice.

## Introduction

Worked-out examples serve an important role in the educational process. Learning from worked-out examples is an important source of learning (VanLehn, 1996), and it is a learning mode preferred by novices (e.g. Anderson, Farrell, & Sauers, 1984; Lefevre & Dixon, 1986). Furthermore, research has shown that learning from worked-out examples is typically very effective (e.g. Sweller & Cooper, 1985; Ward & Sweller, 1990). Such examples are chosen and designed to present not just a solution for a problem, but also a reasoning process along with concepts and issues that are of special importance for the respective subject matter. In order to take the full advantage of the careful design of such examples, students have to "make sense" of the examples, retrieve them, and relate to them whenever it is appropriate. This calls for a knowledge management process to be conducted by the students; and this is what we tried to implement during a course on Knowledge-Based Systems. We expected the students during this knowledge management process of worked-out examples to think about issues of reusability, abstraction, knowledge representation; and respective tradeoffs and dilemmas. The learning tasks we designed consisted on incrementally defining and refining a metadata scheme for a set of given examples. As more and more examples were presented during

the course, students were expected to redefine the metadata scheme to accommodate the variety of examples and relations among them and to express the differences for efficient retrieval. Metadata created by a student for any learning material is a way of representing the student's current understanding of the material. The metadata scheme definition is expected to be a dynamic process that captures the growing understanding and also fosters the dynamic and subjective nature of knowledge and information (Knox, 2007). We encouraged collaboration among students in order to bring up explicitly the dilemmas in defining an appropriate metadata scheme that can be helpful for retrieval and reuse of previous examples while trying to solve a given problem. While managing, retrieving and re-examining worked-out examples is a powerful learning activity for many topics, it is of special importance for students of information technology (and related subjects). Enabling efficient retrieval brings up issues of modeling, of abstraction and of reusability that are of major importance for information technology specialists. An effort to define a metadata scheme for the worked-out examples elicits the current dilemmas of the semantic web regarding standard versus community-based ontologies (Allert, 2004).

The following sections of this paper describe the instructional strategies and the learning tasks of the course and the accompanying study. We conclude with a summary and a discussion regarding the potential benefits of the suggested learning activities.

# The Course -
# Instructional Strategies and Learning Tasks

The course "Introduction to Expert Systems" is an elective course for students in our Computer Science and Information Systems department. The course deals with knowledge-based systems emphasizing symbol processing, while other paradigms are just briefly mentioned. During this course students are introduced to issues of knowledge acquisition, knowledge representation and expert systems design along with actual implementation. The implementation part of the course involves programming with CLIPS, which is an expert system shell (http://www.ghgcorp.com/clips/CLIPS.html). This rule-based system with a declarative programming paradigm is very different from programming paradigms that the students are familiar with. The course includes a hands-on laboratory, where we employ many programming examples. Some of the examples are solutions to exercises that involve various modifications of a given program, each aims at achieving a different goal. Program examples play an important role in teaching programming and the use of a library of annotated examples or templates is highly advocated by computer science educators (Linn & Clancey, 1992; Schank, Linn, & Clancy, 1993). All the learning materials for our course are available to the students through the web. We have a mechanism that enables the teacher to hide solution files from the students while they are working on the laboratory tasks and expose them when it is appropriate. Students find themselves with a multitude of files of potentially useful code examples, which they have difficulties to retrieve when needed. This is an example of a case where students feel the need for a knowledge management mechanism and the creation of metadata to aid the retrieval does not seem artificial.

Students were asked to use the CLIPS formalism to define a metadata scheme for the examples they encounter along with instantiations and examples of retrieval rules. The following is an example of one student submitted scheme presented as a template in CLIPS with slots for the various characteristics of the code examples. Some elements like type declarations were taken out for reading convenience. The comments in parentheses are the author's comments for explaining the formalism, while the other comments (the ones not in parentheses) are the student's comments for explaining mainly the roles of the slots and their characteristics. The student's comments were translated from Hebrew.

```
(deftemplate unit              ; (A CLIPS construct for defining a general template)
   (multislot demonstrationOf )   ;a list of CLIPS commands, constructs etc.
                                     demonstraed by the code
   (slot fileUse                   ;the file was used as a demo in class or is a problem
                                     solution
      (allowed-symbols demo solution)  ; (Possible/allowed values for this slot)
      (default demo))
   (slot name)                        ;name of the file
   (slot lessonNumber      ;the lesson when it was introduced
      (range 1 13))
   (slot resource
      (allowed-symbols teacher me book students other))
   (slot isOrderedFacts    ;whether the program employs ordered facts or not
      (allowed-symbols yes no)
      (default no))
   (multislot relatedFiles) ;a list of related files
   (slot usingControlFact ;whether the program employs a control fact or not
      (allowed-symbols yes no)
      (default no))
   (slot usingDefFunction;whether the program includes a user-defined function
      (allowed-symbols yes no)
      (default no))
   (multislot input         ; input mechanisms
      (allowed-symbols interactive givenFact)))
```

The following code examples present several CLIPS programs that deal with the calculation of the area of a given square. The programs differ with regard to the use of user-defined functions, the input mechanism, output mechanism and the respective CLIPS constructs. Students were asked to consider these examples for checking how well their suggested template (metadata scheme) can differentiate between programs for useful further retrieval. These examples and a set of additional examples were used for team discussions in class, discussions dealing with scheme refinement for differentiating between examples that are similar in the general goal but different in the implementation, generality, input/output facilities etc.

```
(defrule area

(side ?s ?x)

=> (printout t "The area of the square " ?s " is " (* ?x ?x) crlf))


(deffacts data

(side A 6)(side B 2)(side C 10))
```

```
deffunction square
(?a) (* ?a ?a))


(defrule area
(side ?s ?x)
=> (printout t "The area of the square " ?s " is " (square ?x) crlf))


(deffacts data
(side A 6)(side B 2)(side C 10))
```

```
(deffunction square
(?a)(* ?a ?a))


(defrule length
(initial-fact)
=> (printout t "What is the side length? " crlf)
   (assert (side (read))))


(defrule area
(side ?x)
=> (printout t "The area of the square is " (square ?x) crlf))
```

The potential advantages for a learner defining his/her own metadata scheme might be: a feeling of "ownership", better chance of objects being revisited and reinterpreted, deeper understanding through explicit refinement of concepts and relations; and an external representation of learning that supports awareness and reflection of the learning process. Comparisons between learners' metadata schemes can provide an anchor for meaningful discussions as demonstrated in studies of use of conflicts in collaborative learning (Constantino-Gonzalez & Suthers, 2000; Or-Bach & van Joolingen, 2001). The potential for stimulating negotiation of meaning can support the development of a community of practice (Wenger, 1998). The idea that learning involves a deepening process of participation in a community of practice has gained significant ground in recent years and the collaborative refinement of a metadata scheme fosters this idea.

# The Study – Research Tools and Findings

## *Methodology and Research Tools*

Our Study is an exploratory case study with the aim of exploring the educational benefits of having students deal explicitly, through specifically designed tasks, with knowledge management of worked-out examples. The expected benefits relate to both content and to more general cognitive and metacognitive skills. The content facet relates to the specific course as well as to the students

intended profession (like issues of reusability). The other facet relates to cognitive and metacognitive skills such as categorization, abstraction, reflection and more. These issues were explored in the context of individual work and in the context of collaborative work.

The research tools we employed for investigating the evolving learning processes were: homework assignments that involved metadata definition utilizing the CLIPS formalism (definition of a general template with respective slots); observations that were conducted during laboratory/class assignments; and written reflection reports that students were required to submit along with the homework assignments. The written reflection reports consisted of explanations about the choices, the alternatives and the respective considerations that were employed while defining the metadata scheme (the slots' names, the list of possible values or the range of values, default values for various slots etc.).

## *Findings*

The submitted homework provided evidence about individual learning products and processes, while observations during team discussions provided evidence about both individual and collaborative learning processes. Very soon students found that a simple tag or indication, such as in which lesson the specific code example was given, is not informative enough for further retrieval.

In the first assignment most of the templates were like the following:

```
(deftemplate program
        (slot name)
        (slot homework
                (allowed-symbols yes no))
        (slot class_work
                (allowed-symbols yes no) (default yes))
        (slot use_bind
                (allowed-symbols yes no) (default no))
        (slot use_deffunction
                (allowed-symbols yes no) (default yes)))
```

The submitted retrieval rules dealt usually with one criterion. As the course progressed and students had to retrieve examples by several criteria, and more refined criteria they realized that they need more general characteristics (represented as multislot) and then replaced and added multislots like the following:

```
        (slot main_topic
        (allowed-symbols use_slots use_of_read math_function ))
    (multislot more_topics)
        (allowed-symbols ……. ))
    (slot similar_program))
        (allowed-symbols ….. ))
```

An important issue that was discussed by students was the gradual updating of the list of allowed symbols and the need to keep consistency of terms/vocabulary for meaningful retrieval. This required reinterpretation of the worked-out examples.

Students gradually found that code examples, and hence useful slots (metadata tags), might relate to syntax issues, to the use of various constructs in CLIPS, to different ways to solve a given problem, to ways to extend a solution etc. They found that examples can be linked by a CLIPS related issue, by a problem statement, by being a part of a bigger problem, by being one of several possible solutions, by relating to a specific pre-requisite etc. Findings show a shift from more objective and general metadata towards more subjective and specific metadata: from issues like date and lesson number towards issues like the program topic and the CLIPS constructs that are involved; and then towards more subjective characteristics such as expected contexts of use, comparisons with other programs, associations etc. Findings show also a shift towards more consideration of retrieval during tags' definition instead of just considering the storage. With regard to the vocabulary we found terms that related to the following categories which are not mutually disjunctive: CLIPS syntax, CLIPS knowledge constructs, the respective "problem story", the learning sequence, the origin (e.g. class example), expected reuse, "part of" relation, input/output mechanisms and value assignment procedures. We observed processes of abstraction, when students modified their scheme by replacing a slot (tag) or more that has "exists"/"not exists" values by a new slot with new respective values. For example, the last slot in the student's deftemplate presented before is an abstraction of a previously defined slot "using the read function" (for user input) with values "yes" and "no". Observations showed that negotiation of meaning had a major impact on learning. It also showed the evolution of a community of practice; giving students the opportunity to experience processes similar to what they may encounter in future work situations.

# Summary and Discussion

Our findings show that the learning task of metadata creation can be seen as a "mindtool". The term "Mindtools" (Jonassen, 1999) is used to describe information technology formalisms that are used to engage students in critical thinking. The activities of metadata creation brought students to rethink and reinterpret the various code examples and the various relations between them. As many of the examples were worked-out examples, the required rethinking about these examples helped students to take advantage of these worked-out examples within the specific course instructional/learning goals. The collaborative metadata creation provided an additional opportunity to foster critical thinking as well as to practice interpersonal communication and team skills. As students tend to use superficial clues for retrieval of examples (Lithner, 2003), a systematic approach for knowledge management can be beneficial for them.

We were especially interested in educational benefits that are more oriented towards students of information systems and related areas. The main relevant educational benefits relate to modeling and abstraction; to understanding the nature of knowledge and information; and to reusability related issues. Abstraction activities received a central role during students' metadata definition processes, reflecting the major role of abstraction in any computing education and in modeling. The on-going process of metadata definition individually and collaboratively made students aware of the dynamic and subjective nature of knowledge. While it is common practice to see information and knowledge as different stages in s spectrum (Bellinger, Castro, & Mills, 2004), the appreciation of the central role of the human interpreter (Knox, 2007) is important for both concepts. This conceptualization is necessary for anybody interested in information, knowledge, and the way they can be acquired, modeled, reused, or repurposed. This also brings up the issue of reusability and how metadata definition can support efficient reusability processes. Reusability is an important issue within software engineering and one impediment to software reusability is the difficulty in classifying software resources in such a way that they can be retrieved by other

people. Much of the students activities and considerations echo the ones performed in E-learning with regard to learning objects and metadata to enable the retrieval of appropriate learning objects by human teachers as well as virtual tutors (Carey, Swallow, & Oldfield, 2003, Friesen, 2002; Recker & Wiley, 2001). .

Besides the educational benefits for a student and for a community of students (or a student within a community), the suggested learning activities entail also potential benefits for tutoring. A student's series of metadata schema representing the gradual process of learning and understanding can serve as input for relevant student modeling processes. The resulted student model can in turn be utilized by a human or machine tutor for adapted tutoring interventions. This external representation of the learning process can serve also as an open student model to support students' reflection processes (Dimitrova, 2003; Kay, 2001).

Our future plans include the employment of more structured observations, interviews and surveys in order to refine the analysis and hence the description of the individual learning processes as well as the community learning. We are currently redesigning the respective homework to capture the ongoing learning process by refining the assignments and adding more specific requirements regarding reflection on the respective processes. Future plans include also the use of the new refined findings for designing adaptive interventions based on the evolution stages of the metadata scheme. The planned adaptive interventions should help students in modifying and restructuring the metadata scheme and should support productive collaboration.

# References

Allert, H. (2004). Coherent social systems for learning - An approach for contextualized and community-centred metadata. In T. Anderson, & D. Whitelock (Eds.), The educational semantic web: Visioning and practicing the future of education, *Journal of Interactive Media in Education*, *1*, Special Issue on the Educational Semantic Web.

Anderson, J. R., Farrell, R., & Sauers, R. (1984). Learning to program in LISP. *Cognitive Science*, *8*, 87-129.

Bellinger, G., Castro, D., & Mills, A. (2004). *Data, information, knowledge, and wisdom*. Retrieved November 2007 from  http://www.systems-thinking.org/dikw/dikw.htm

Carey, T., Swallow, J., & Oldfield, W. (2002). Educational rational metadata for learning objects. *Canadian Journal of Learning and Technology*, *28*, 3.

Dimitrova, V. (2003). StyLE-OLM: Interactive open Learner modeling. *International Journal of Artificial Intelligence in Education*, *13*(1), 35-78.

Friesen, N. (2003). Three objections to learning objects. In R. McGreal (Ed.), *Online education using learning objects.* London: Taylor & Francis Books.

Constantino-Gonzalez, M. A., & Suthers, D. D. (2000). Using meta-cognitive conflicts to support group problem solving. In G. Gauthier, C. Frassson, & K. Van-Lehn (Eds.), *Intelligent Tutoring Systems: 5th International Conference* (pp. 232–242). Berlin: Springer.

Jonassen, D. H. (1999). *Computers as mindtools for schools: Engaging critical thinking* (2nd ed.). Prentice Hall.

Kay, J. (2001). Learner control. *User Modeling and User-Adapted Interaction,* 11*(1/2)*, 111-127.

Knox, K. T. (2007). The various and conflicting notions of information. *Issues in Informing Science and Informing Technology*, 4, 675-689. Retrieved from http://proceedings.informingscience.org/InSITE2007/IISITv4p675-689Knox452.pdf

LeFevre, J. A., & Dixon, P. (1986). Do written instructions need examples? *Cognition and Instruction*, *3,* 1-30.

Linn, M. C., & Clancy, M. J. (1992). The case for case studies of programming problems. *Communications of the ACM, 35*(3), 121-132.

Lithner, J. (2003). Students' mathematical reasoning in university textbook exercises. *Educational Studies in Mathematics, 52*, 29–55.

Or-Bach, R., & van Joolingen, W. R. (2001) Contradictions as an anchor for providing help during collaborative learning. *Proceedings of the Workshop on Help Provision and Help Seeking in Interactive Learning Environments, The 10th International Conference on Artificial Intelligence in Education*, Texas.

Recker, M., & Wiley, D. (2001). A non-authoritative educational metadata ontology for filtering and recommending learning objects. *Interactive Learning Environments, 1*, 1–17.

Schank, P. K., Linn, M. C., & Clancy, M. J. (1993). Supporting Pascal programming with an on-line template library and case studies. *International Journal of Man-Machine Studies*, *38*(6), 1031-1048.

Sweller, J., & Cooper, G. A. (1985). The use of worked examples as a substitute for problem solving in learning Algebra. *Cognition and Instruction*, *2*, 59-89.

VanLehn, K. (1996). Cognitive skill acquisition. In J. Spence, J. Darly, & D. J. Foss (Eds.), *Annual review of psychology*. Palo Alto, CA: Annual Reviews.

Ward, M., & Sweller, J. (1990). Structuring effective worked examples. *Cognition and Instruction*, *7*, 1-39.

Wenger, E. (1998). *Communities of practice – Learning, meaning and identity*. Cambridge: Cambridge University Press.

# Biography

**Rachel Or-Bach** is a senior lecturer at The Max Stern Academic College of Emek Yezreel in Israel, in the Computer Science and Information Systems area. She earned her doctorate from the Technion – Israel Insitute of Technology. Her main research interest is Intelligent learning environments. She is a member of the Artificial Intelligence and Education society and a reviewer of the society journal.