

# Embedding Experiences in Micro-Didactical Arrangements

Eric Ras, Stephan Weibelzahl

Fraunhofer Institute for Experimental Software Engineering (IESE)  
Sauerwiesen 6, 67661 Kaiserslautern, Germany  
{eric.ras, stephan.weibelzahl}@iese.fraunhofer.de

**Abstract.** Experience-based Information Systems (EbIS) enable organizations to capture, store and reuse knowledge and experiences for continuous competence development. However, there are several shortcomings that seem to limit the usage of the stored knowledge. Focusing on technical issues, the stored experiences consist mainly of contextual knowledge provided by domain experts, while declarative and procedural knowledge is required in addition to facilitate learning for novices. Moreover, these systems do not support learning in an optimal way because they do not activate learning processes. We present an approach that enriches retrieved experiences with additional learning elements in so-called micro-didactical learning arrangements, created by a pedagogical agent based on cognitive learning goals and an instructional design model. The advantages of this approach are twofold: first, the applicability of experience packages increases by adding learning elements to the package; second, the application of the experience and the newly gained knowledge in practice deepens the learning effect.

## 1 Introduction

Continuous competence development is essential to keep track with the requirements of today's work environments. This trend can be observed especially in the Information and Communication Technologies sector with its increasing flood of information, rapid deterioration and hence ageing of knowledge, as well as the continuously changing requirements for problem understanding and solving. As a result, these facts require lifelong learning to remain competitive in the information society.

Today, learning is less a reaction to 'being learned' but more the reaction to varied requirements of learning situations and learning environments. The short innovation cycles in Software Engineering lead to many learning situations where new knowledge is required to solve new challenges during daily work.

In the future, learning within an organization will balance out structured, directed learning and unstructured, autonomic learning. Autonomic learning consists of learning without direct teaching. Learners define their own learning goals according to given situations and select the learning steps as well as their sequence to reach the goals. Autonomic learning is more a way of explorative learning than learning

This is a preprint of the final version which is copyrighted by Springer Verlag 2004: Ras, E., & Weibelzahl, S. (2004). Embedding Experiences in Micro-Didactical Arrangements. In G. Melnik & H. Holz (Eds.). *Advances in Learning Software Organizations. Sixth International Workshop on Learning Software Organizations, LSO 2004*, pp. 55-66, Berlin: Springer.

based on given procedures and rules. Directed learning will be launched by the organization to communicate and change their strategy, culture, products and services, which involves individuals, teams or the entire organization. Autonomic learning originates within the organization, initiated by individuals and communities of practice.

The increasing number of Knowledge Management Systems (KMS) led to research and developments focusing mainly on capturing, structuring, and packaging knowledge for reuse. One of the domains where KMS's were profitably implemented is Software Engineering [22], a quickly changing, knowledge-intensive business involving many people working in different phases and activities. However, organizations frequently encounter problems identifying the content, location and use of knowledge. As a result 50 to 60 percent of KM deployments failed because organizations did not have a good KM deployment methodology or process, if any at all [17]. Regarding the deployment process, learning is considered to be a fundamental part of Knowledge Management (KM) since employees must internalize (learn) shared knowledge before they can use it to perform specific tasks [22]. It was assumed that KMS's could solve the problem of continuous competence development by providing intelligent retrieval mechanisms and innovative presentations techniques. KMS's focuses mainly on the knowledge, (i.e., the product of learning processes), and less on learning processes itself and the needs of individuals. A recent study stated that 'next generation' KMS developments should focus on designing KM technologies for people and not make people to adapt to KM technologies [17]. Designing KM technologies for people means supporting people in their learning processes to ensure that the provided knowledge can be transferred back to the work process. Enhancing learning means more than sequencing 'chunks' of knowledge. It requires an understanding of learning goals and processes, and the different types of learners and their competence levels.

An interview-based study showed that perceived connections between KM and e-learning are not operationalized, i.e., the integration ideas are rarely implemented in practice. KM addresses learning mostly as part of knowledge sharing processes and focuses on specific forms of informal learning (e.g., learning in a community of practice) or to providing access to learning resources or experts [9]. In addition to these interviews, an outcome of a follow-up workshop was that future KM initiatives should shift their focus from knowledge sharing to support actual learning from others and applying the experiences of these people [8].

In this paper, we present an approach that enriches retrieved experiences (i.e., applied knowledge) with additional learning elements in micro-didactical learning arrangements in order to enable learning from others' captured knowledge. These arrangements are created by a pedagogical agent based on learning goals and associated educational learning patterns.

## 2 Experience-Based Information Systems (EbIS)

While knowledge is frequently seen as the range of learned information or understanding of a human or intelligent information system, experience is considered to be “knowledge or practical wisdom gained through human senses, from directly observing, encountering, or undergoing things during the participation in events or in a particular activity” [25, p.24]. Experience Management (EM) can be seen as a sub-field of KM that aims at supporting the management and transfer of relevant experiences [6, 25]. The software system used for managing, storing, retrieving and disseminating these experiences is called an Experience-based Information System (EbIS) [15] that is based on the *Experience Factories* concept [5]. Another type of systems that are not based on the EF concept is *Lessons Learned Systems* (LLS) [27]. Amongst the definitions for lessons learned the most complete definition as stated by Weber et al. [27, p.3] is: “A lesson learned is knowledge or understanding gained by experience. The experience may be positive, as in a successful test or mission, or negative, as in a mishap or failure. Successes are also considered sources of lessons learned. A lesson must be significant in that it has a real or assumed impact on operations; valid in that it is factually and technically correct; and applicable in that it identifies a specific design, process, or decision that reduces or eliminates the potential for failures and mishaps, or reinforces a positive result.” [23]. In this paper we use the term *experience package* (EP) which includes both experiences embedded within EbIS’s and lessons learned within LLS’s.

We argue that experience packages that are retrieved by EbIS’s are often inadequate for learning and competence development for several reasons. First, the experience base might be incomplete, outdated or might just contain wrong information. This problem has been addressed by several knowledge elicitation and maintenance approaches (see [6] for an overview). Second, users might refuse to apply an experience package, because of a lack of confidence. There is a chance that the quality of packaged experiences is also influenced by contradictory interests (e.g., commercial interests). Thus, it might be unclear whether applying the packaged experience might involve a certain risk. A good and transparent quality assurance process might alleviate this problem. Finally, even if the EB has a high coverage and precision, and even if the users have enough confidence in the quality of the information, the packages might still be inappropriate for learning due to the fact that the learning issues are not considered by the EbIS. Often, users need additional information about the subject domain, because experts provided the experience without giving extensive explanations of the background and because the users lack knowledge of domain concepts. Our approach aims at closing this gap by enriching the experience packages with learning elements based on didactical considerations.

### 3 Learning Based on Packaged Experiences

One reason why Software Engineering knowledge is usually captured from experts is that their knowledge is assumed to be concise, correct and complete. Further, by finding out what kind of knowledge is considered essential by the experts, we hope to get information about what should be learned by novices. A problem that occurs when expert knowledge is used for teaching novices is that there is not only a quantitative difference between expert and novice knowledge bases, but also a qualitative difference, e.g., the way in which knowledge is organized [11].

The following sections provide an overview of the different knowledge types and related knowledge stages and how we can learn from expert knowledge. Finally, we derive requirements for micro-didactical learning arrangements.

#### 3.1 Knowledge Types

Anderson developed a model of the architecture of human knowledge. He classifies knowledge not according to its content but according to its state in the person's long-term memory. Two types of knowledge were defined [4,13]:

- Declarative knowledge consists of 'knowing about' – e.g., facts, impressions, lists, objects and procedures,, and 'knowing that' certain principles hold. Declarative knowledge is based on concepts that are connected by a set of relations forming a network that models the memory of a person. This leads to the conceptual and theoretical understandings that remain long after many facts are forgotten. For instance, declarative knowledge items in the domain of Software Engineering might be: a definition of 'test case', a listing of defect types, a detailed explanation of key testing principles.
- Procedural knowledge consists of 'knowing how' to do something, i.e., skills to construct, connect and use declarative knowledge. Learners are doing tasks, such as understanding and processing relationships between items (e.g., facts or objects) and creating new connections between them. Procedural knowledge contains the discrete steps or actions to be taken, and the available alternatives to perform a given task. Procedural knowledge consists also of 'if-then' rules that describe conditions when to perform certain actions in a specific situation. These rules are abstract, modular (i.e., they can be combined), goal-oriented, and operate on the basis of declarative knowledge. With sufficient practice, applying the rules of procedural knowledge may become an automatic process, thus allowing the person to perform a task without conscious awareness. For instance, procedural knowledge items in the domain of Software Engineering might be: a method for deriving test cases from requirements, a method for classifying defects choosing the right reading technique to perform an inspection.

Both declarative and procedural knowledge can be abstract or concrete. The knowledge can be connected to more or less concrete information that can be described technically, e.g., by semantic networks. Nevertheless, knowledge about experienced situations or evaluating facts or determining circumstances in given situa-

tions, cannot be classified as declarative or procedural knowledge. Therefore, a third form of knowledge, conditional or contextual knowledge describing 'when, where and why', has extended the spectrum of knowledge in cognitive science [10]. In the context of didactical design, Tennyson and Rasch [26] defined contextual knowledge as another type of knowledge:

- Contextual knowledge consists of 'knowing when, where and why' to use or apply declarative or procedural knowledge. Contextual knowledge is created by reflecting on the usage of declarative and procedural knowledge in practice in different contexts. Contextual knowledge enables the individual to be aware of commonalities between situations, and of the appropriateness or applicability of principles or procedures in a new context.

In summary, three types of knowledge have been presented. Packaged experiences, i.e., documented experiences, consist mainly of contextual knowledge. They originate in most cases from expert memories and they lack declarative basis background knowledge and detailed procedural knowledge, resulting in them being ineligible for learning purposes. The next section details the problem mentioned above and focuses especially on barriers that exist when individuals are learning from expert knowledge.

### **3.2 Learning from Expert Knowledge**

The standard method for transferring knowledge from experts to novices is the 'copy-model': expert knowledge is considered as learning material and they are transferred directly to the learner by using an appropriate medium. Even if this model is commonly applied in many different educational contexts and KMS's, it does not comply with the structures and processes of human information processing [3]. What makes the transfer of expert knowledge or experiences difficult? Firstly, from a cognitive science point of view, new knowledge is always related to existing knowledge in human memories. This means that novices might have problems in relating new expert knowledge to their existing 'basics'. Secondly, learning is a special case of information acquisition and information storage. The learning process is dependent first on the quality of the information to be learned and second on the cognitive activities of learning. If those activities do not take place because of the problem stated above, the efficiency of information acquisition and storage is decreased. Thirdly, there is not only a quantitative difference between expert and novice knowledge bases, but also a qualitative difference, i.e., the organization of knowledge [11]. Cognitive schemata from experts cannot be transferred to the memories of novices. This fact results from a compilation process that is performed when new knowledge is learned: updating or forgetting of old knowledge, creating new relations or rules between knowledge items and aggregating knowledge items etc. Fourthly, asking experts about their knowledge results often in an enumeration of many facts, methods and principles explained in a complex manner. Experts forgot about how they learned those knowledge chunks, and they are unable to explain why they choose certain activities to perform them in a certain manner. The applied

knowledge is somehow 'routine' [11]. Finally, transferring past experiences made by others requires more than only contextual knowledge, in particular problem-solving strategies for a specific context and knowing 'when, where, and why' knowledge should be used. It requires a strong anchoring with declarative and procedural knowledge.

The goal of effective knowledge transfer is to guide a novice through so-called knowledge stages to become an expert. Fitts and Posner defined three different stages [12]: The goal during the first *cognitive stage* is to build up basic conceptual knowledge and to integrate the knowledge into a semantic network. By solving problems in this stage, the person accesses simple concepts and content specific rules that are represented as declarative knowledge. In the second *associative stage*, domain specific rules are created. These 'if-then' rules are based on associations between conditions and specific operations, and they broaden procedural knowledge in the current context of the problem to be solved. If the 'if'-part of the rule is fulfilled, the 'then'-part is automatically used for problem solving. The third *autonomous (automatic) stage* is reached after many years of practice. The rules have a high degree of association because of many applications. This results in a replacement of many rules by high-level and simple rules. These simple rules have often a reduced 'if' and 'then' part. A characteristic of these rules is that they cannot be verbalized, i.e., the rules cannot be explained to others and they determine autonomous activity and behavior in general. This evolution process makes the transfer of knowledge and connected rules between novices and experts difficult.

The next sections propose an approach how to bridge the gap between novice and expert knowledge levels, resulting from many compiling cycles of knowledge evolution in experts' memories.

### **3.3 Enabling Learning from Packages Experiences**

The main purpose of an EbIS is to provide the right experience package to solve a problem in a certain situation. Often, the packaged experiences originate from applied expert knowledge, documented by the experts themselves. As described in the previous sections, novices might have problems to apply these experiences in practice.

The central goal of our approach is to embed experts' experience packages related to the Software Engineering domain into *micro-didactical learning arrangements* that enable cognitive learning processes and that allow less experienced persons to acquire the necessary skills to apply the experience packages in practice.

Such a learning arrangement consists of a learning offer that allows learning in context. The experience is enriched with additional so-called learning elements. A learning element is an atomic chunk of knowledge, either declarative, procedural or contextual, that represents content about facts, processes, rules and principles and augments specific cognitive learning activities by reading, searching, orientating, summarizing and reflecting. In the following, the requirements of such an arrangement are listed:

- A micro-didactical arrangement contains learning elements to all three types of knowledge and thus ensures that contextual knowledge is anchored with declarative and procedural knowledge.
- An arrangement follows the constructivist and the pedagogical principle of autonomous learning, i.e., explorative learning. Explorative learning is less a didactical design principle, but it ensures that the learners deal explicitly with the problem to be solved; the learners choose their own learning goal; the learners are collecting new knowledge by themselves, construct their own cognitive schemata and strengthen the associations between different knowledge types; the learners perform learning by-doing to get new insights in complex circumstances and principles and acquires necessary skills for applying the embedded experience.
- Explorative learning does not take place as a linear process from one topic or one difficulty level to another but more in a cyclical manner. Knowledge on different topics is acquired in parallel. Changing between topics could be useful for a better understanding of the topics directly related to the problem. Such a cyclical procedure is typical observed when complex problems have to be solved. Explorative learning is essential to understand the relationships between different subdomains, to enhance the orientation within the domain by building up individual cognitive schemata.
- An arrangement is created automatically by a so-called pedagogical agent. A pedagogical agent uses a set of micro-didactical patterns based on an instructional model, a Software Engineering ontology and a semantic network for learning elements (see next section).
- An arrangement is conform to current e-learning standards and this ensures the compatibility to other platforms, like Learning Content Management Systems (LCMS).

The following section describes the design of our approach: a model for competence levels with associated knowledge stages is defined, a set of predefined learning goals, a taxonomy for learning elements that are used for enriching packages experiences is provided, and the creating process for micro-didactical arrangements is elaborated.

## 4 Enabling Learning in EbIS

By referring to the knowledge stages described above, we define a *novice* as a person in the cognitive knowledge stage who starts to build up basic declarative knowledge. A *practitioner* is somebody who is currently in the associative stage, i.e., creating more associations between declarative knowledge and identifying ‘if-then’ rules to broaden the procedural knowledge in the current context of the problem to be solved. An *expert* is a person whose knowledge has reached an autonomous stage. Experts are able to create or adapt their rules automatically by informal learning on the job and applying current knowledge in practice. The following sections describe how we can enrich packaged experiences by learning elements and how we can address the

requirements listed in Section 3 to create micro-didactical arrangements. Our approach focuses especially on knowledge acquisition for novices and practitioners.

#### 4.1 Educational Goals

Educational goals differ widely in dependence of the target audience and the knowledge of the learners. For instance, novices should usually learn the basic facts and definitions of a topic, before bothering them with details and inconsistencies. Practitioners are interested in getting hints and instructions on how to perform a given task in the first place. Thus, in order to provide suitable learning material, an adequate learning goal has to be specified. In our approach, we refer to Bloom's taxonomy of educational goals [5,15], which is widely accepted and applied in various topic areas including Software Engineering [1].

Bloom defines and describes six performance levels that differ in terms of the complexity from a cognitive point of view. The lowest level is called *knowledge* and aims at the acquisition of facts and definitions. The learner should be able to recall information such as dates, events and places. The next level, *comprehension*, goes beyond knowledge, as the learner is required to understand the meaning of the information. Learners should be able to interpret and compare facts and summarize ideas. The *application* level aims at using the acquired information, methods and concepts to solve problems. Theories are applied in new situations. The *analysis* level comprises the ability to identify underlying patterns and components. On the *synthesis* level learners are able to use old ideas to create new one, to generalize from given facts and to draw conclusions. Finally, learners that reached the *evaluation* level are able to assess the value of theories, make choices based on reasoned argument and recognize subjectivity.

Our approach addresses only the first three levels of this taxonomy, because these are important for reaching the upper levels and can be taught directly, while the fourth to sixth level require a longer term and deeper insight into a subject matter.

#### 4.2 Taxonomy of learning elements

In order to select the most suitable learning elements from the repository, we assign each element to a learning element type derived from another taxonomy introduced by Meder [20]. According to this taxonomy, learning elements belong to one of four types: First, they might provide an *Explanation* (e.g., definition, description, example). Second, they might serve as *Orientation* (e.g., overview, summary, history). Third, a learning element might be *Action* knowledge that contains information on how to perform a task (e.g., procedure, checklist, rule, principle). Finally, it might be a *Reference* to a source (e.g., document reference, annex reference, glossary reference). These learning element types can be categorized in terms of Bloom's educational learning goals. *Knowledge* and *Comprehension* are goals related to the cognitive knowledge stage (i.e., declarative knowledge). The *Application* goal is part of the associative knowledge stage (i.e., procedural knowledge).

Each learning goal requires a different learning activity that focuses on different aspects of cognitive schemata. Table 1 provides an overview of the assignment of learning elements to goals. Meder claims that the list of elements has been applied successfully but might require extensions for new domains.

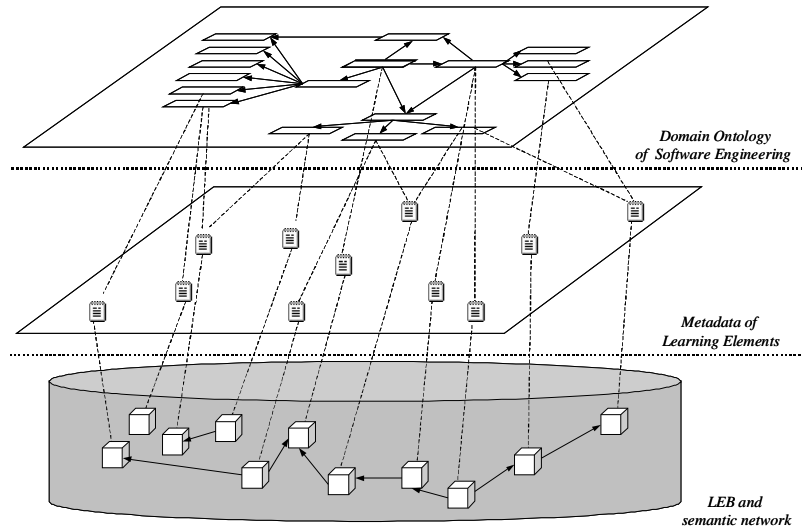
**Table 1.** Overview of educational goals and associated learning elements

Knowledge	Comprehension	Application
Definition	example	procedure
description	counterexample	administrative direction
theorem	summary	instruction
reference	history	social direction
	overview	principle
	scenario	strategy
		checklist
		law
		rule

Based on this categorization every atomic learning element is now related to a high level educational goal. The next section describes the relations between the learning elements that are used by the pedagogical agent to create the learning arrangements.

### 4.3 Learning Elements and Their Relations

Learning elements (LE's) are stored in the learning element base (LEB). The LE's that are used in our system have been created during recent projects related to Software Engineering learning content. Each learning element is described by means of a metadata set (see Fig. 1). The metadata description is based on the Learning Object Metadata standard (LOM) [14]. LOM specifies the syntax and semantics of Learning Object Metadata, defined as the attributes required to fully and adequately describe a learning object. Within the metadata description relations to the Software Engineering (SE) domain ontology are coded. We distinguish between two types of references: references describing the learned concept(s) after using the LE and the references listing the prerequisite concept(s) for using this LE, i.e., knowledge that should be known before using the LE. The ontology consists of key SE-concepts (i.e., keywords) that are connected by relations such as: *is-a*, *part-of*, *consumes*, *produces*, etc.

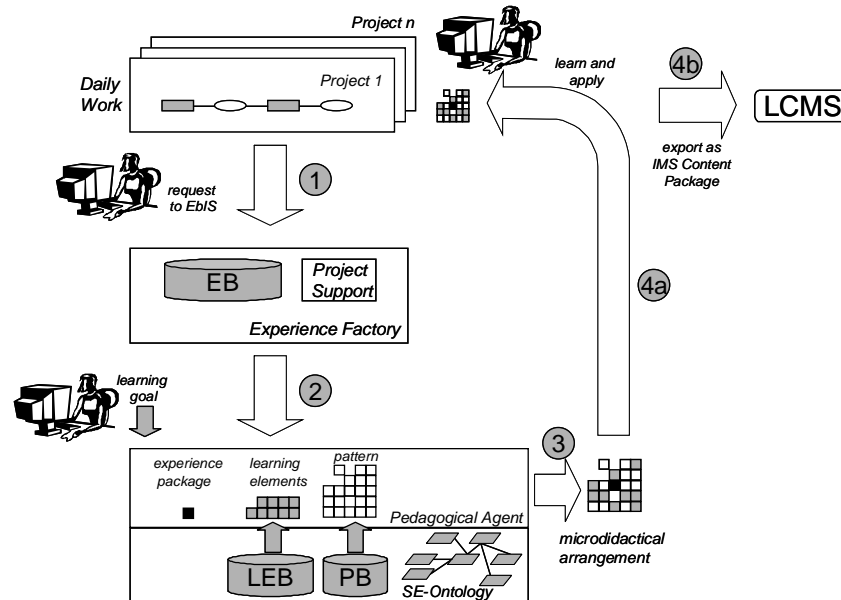


**Fig 1.** Learning elements and their relations

#### 4.4 Creation Process of Micro-Didactical Arrangements

By providing a set of educational goals, a learning element taxonomy, and their classification to knowledge types, we are now able to describe the process of creating a micro-didactical arrangement.

After the occurrence of a problem during daily work (see step 1 in Fig. 2), the user searches for suitable experiences in the repository to solve the problem. The retrieved and selected experience is forwarded to a pedagogical agent (see step 2). The user selects a learning goal by taking the decision on which level she or he wants to acquire knowledge. The pedagogical agent selects an appropriate pattern from the pattern base (PB) according to the type of experience, i.e., technology, process or product package, and the selected learning goal. Patterns are instances from an instructional model that is based on the 4C/ID approach introduced by Merriënboer et al [21]. The focus of this model is on learning task-specific skills. We apply the model for enabling the construction of individual cognitive schemata and to ensure the applicability of experience related knowledge into practice. According to the selected learning goal and the experience type, learning tasks are selected to learn necessary skills for applying the experience.



**Fig 2.** Creation and use of micro-didactical arrangements

As described before, each experience package is annotated by concepts of the SE domain ontology. The agent is able to retrieve LE's that refer to the same concepts as the experience package and to infer prerequisite LE's by using the metadata description of the LE's. Then, the agent inserts the experience and the LE's into the pattern (see step 3). The agent uses the rhetoric-didactical relations, selects and sequences the LEs according to the selected learning tasks. The produced micro-didactical arrangement is provided to the user (see step 4a). The arrangement supports explorative learning in a cyclical way and the construction of cognitive schemata. The package experience is the central element of the arrangement and the trigger for learning. Depending on the selected goal, the user can decide which learning task to take first, i.e., whether he or she wants to acquire background knowledge (basics) first, or to get an orientation in the SE domain by browsing a graphical representation of the ontology, or to read examples and to try parts of them in practice. The learner can decide whether the appropriate knowledge level is reached that is prerequisite for understanding or applying the experience package. An option is, to make the micro-didactical arrangement conform to the *Sharable Content Object Reference Model (SCORM)* and to export it as IMS Content Package in order to make it available to conventional Learning Content Management Systems (LCMS) (see step 4b). SCORM is a reference model for learning content and provides the framework and detailed implementation reference that enables content, technology, and systems using SCORM to 'talk' to each other, thus ensuring interoperability, reusability and manageability [2].

## 5 Summary and Future Work

In this paper we stated the barriers when people are learning from expert knowledge from a cognitive and didactical point of view. We propose an approach that enables novices and practitioners to learn from packaged expert knowledge, by providing a set of learning goals, a taxonomy for learning elements, and an agent who combines learning elements with experiences to micro-didactical learning arrangements according to an instructional model. The advantages of this approach are twofold: first, the applicability of experience packages increases by embedding the experience into a learning arrangements; second, the application of the experience and the new gained skills deepens the learning effect and strengthens the cognitive schemata.

Future work will focus on further developing and evaluating educational patterns in practice and using the results for adapting the instructional model that is used by the agent for creating patterns. Skill taxonomies and learning tasks will be developed that enforce the construction of cognitive schemata for the SE domain. Another issue will be to observe current development of e-learning standards and specifications to ensure the interoperability of micro-didactical arrangements with LCMS.

## References

1. Abran A., Bourque P., Dupuis R., MooreDonald J. W.: Guide to the Software Engineering Body of Knowledge (SWEBOK). IEEE Press, Piscataway (NJ). Retrieved on 2004/02/28, <http://www.swebok.org/>
2. ADL: Sharable Content Object Reference Model (SCORM) Version 2004, (2004), Retrieved 2004/02/20, <http://www.adlnet.org>
3. Anderson, J. R.: Kognitive Psychologie. Spektrum der Wissenschaft, Heidelberg, (1988)
4. Anderson, J. R.; Rules of the mind. Hillsdale, Erlbaum, New York (1993)
5. Basili, V.R., Caldiera, G., Rombach, D.: Experience Factory. In Marciniak, J.J. (ed.), Encyclopedia of Software Engineering, John Wiley & Sons, vol 1, (1994), 469–476
6. Bergmann R.: Experience Management: Foundations, Development Methodology, and Internet-Based Applications. Springer, ( 2002)
7. Bloom B. S., Engelhart, M. D., Furst, E. J., Hill, W. H., Krathwohl, D. R.: Taxonomy of educational objectives: The classification of educational goals: Handbook 1 cognitive domain. Longmans, Green and Company, New York (1956)
8. Efimova, L., Swaak, J., Converging Knowledge Management, Training and e-learning: Scenarios to make it work. Journal of Universal Computer Science, vol. 9, no. 6 (2003), 571-578
9. Efimova, L., Swaak, J.: KM and (e)-learning: towards an integral approach? Proc. KMSS02, EKMF, Sophia Antipolis (2002), 63-69
10. Enns, C. Z.: Integrating Separate and Connected Knowing: The Experiential Learning Model. Teaching of Psychology 20(1), (1993), 7-13
11. Ericsson, K. A., Krampe, R. T., Tesch-Römer, C.: The role of deliberate practice in the acquisition of expert performance. Psychological review 100, (1993), 363-406
12. Fitts, P. M., Posner, M. I.: Human performance. Brooks Cole, Belmont, CA (1967)
13. Gagne, R. M., Briggs, L. J., Wager, W. W.: Principles of instructional design. (3rd ed.). Holt, Rinehart and Winston, Incorporated, New York (1988)

14. IEEE Learning Technology Standards Committee: Learning Object Metadata Standard. Retrieved on 2004/02/26, <http://ltsc.ieee.org/wg12/>
15. Jedlitschka A., Nick M.: Software Engineering Knowledge Repositories. In *Empirical Methods and Studies in Software Engineering: Experiences from ESERNET*, (2003), 55-80
16. Krathwohl, D. R., Bloom, B. S., Masia, B. B.: Taxonomy of educational objectives: The classification of educational goals. Handbook ii, Affective domain, David McKay Company, Incorporated, New York (1964)
17. Lawton G.: Knowledge Management: Ready for Prime Time? *Computer*, vol. 34, no. 2, (2001), 12-14
18. Madche, A.: -VISION- A roadmap Toward Next Generation Knowledge Management. Presentation, Prague, VISION EU Project, IST-2002-38513, (2002), Retrieved on 2004/02/20, <http://km.aifb.uni-karlsruhe.de/fzi/presentations/docs/1035178906.ppt>
19. Mann W. C., Thomson S. A.: Rhetorical Structure Theory: A Theory of Text Organization. Technical Report RS-87-190, Information Science Institute, USC ISI, USA (1987)
20. Meder, N.: Didaktische Ontologien, Retrieved on 2004/02/21, <http://www.l-3.de/de/literatur/download/did.pdf>
21. Merrienboer van, J. J. G., Clark, R. E., & de Croock, M. B.: Blueprints for complex learning: The 4C/ID\* model. *Educational Technology, Research and Development*, 50(2), (2002)
22. Rus, I., Lindvall, M.: Knowledge Management in Software Engineering. *IEEE Software*, May/June (2002), 26-38
23. Secchi, P., Ciaschi, R., Spence, D.: A Concept for an ESA lessons learned system. In P. Secchi (Eds.), *Proceedings of Alerts and LL: An Effective way to prevent failures and problems*, Tech. Rep. WPP-167, Noordwijk, The Netherlands: ESTEC, (1999), pp. 57-61
24. Steinacker A., Seeberg C., Fischer S., Steinmetz R.: MultiBook: Meta-data for Webbased Learning Systems. *Proceedings of the 2nd International Conference on New Learning Technologies*, (1999)
25. Tautz C., Customizing Software Engineering Experience Management Systems to Organizational Needs. PhD thesis, University of Kaiserslautern, Germany, 2000. Fraunhofer IRB Verlag, (2001)
26. Tennyson, R. D., Rasch, M.: Linking Cognitive learning theory to instructional prescriptions. *Instructional Science*, 17, (1988), 369-385
27. Weber, R., Aha, D.W., Becerra-Fernandez, I.: Intelligent lessons learned systems. *International Journal of Expert Systems Research & Applications*, Vol. 20, No. 1, (2001), 17-34