International Journal of Engineering Education Vol. 27, No. 1, pp. 41–51, 2011
Printed in Great Britain

0949-149X/91 $3.00+0.00

An Environment for Project-Based Collaborative Learning of Software Design Patterns*

ZORAN JEREMIC,[1] JELENA JOVANOVIC[1] and DRAGAN GASEVIC[2]
[1] FON-School of Business Administration, University of Belgrade, Serbia
[2] School of Computing and Information Systems, Athabasca University, Canada.
E-mail: jeremycod@yahoo.com, jeljov@gmail.com, dgasevic@acm.org

Software engineering education faces increasing pressure to provide students with those skills required to solve different kinds of software problems both, alone or as a member of a development team. Consequently, one of the main goals of software engineering curriculum is to teach students how to model, design and implement software, as well as how to exploit previous successful experiences and knowledge of others in solving similar problems. These are inherently practical skills and rely on functioning knowledge. To facilitate a learning environment in which students can acquire a necessary level of understanding, it is necessary to apply an active learning paradigm, which recognizes that student activity is critical to the learning process. In this paper, we propose a project-based collaborative learning environment for learning software design patterns that integrates several existing educational systems and tools based on the common ontological foundation. The learning process in the suggested environment is further facilitated and augmented by several context-aware educational services. Our first evaluation demonstrated some promising results for effective teaching and learning of design patterns.

Keywords: semantic web; ontologies; collaborative learning; project-based learning; software patterns; context-awareness

1. Introduction

Software engineering education, like the education in other engineering disciplines, faces increasing pressure to provide students with the skills necessary to integrate theory and practice, and thus enable them to succeed in their professional jobs. Software engineering students should learn how to solve different kinds of software problems both on their own and as members of a development team. The learning process should provide them with the ability to model, design and implement software. These are inherently practical skills and rely on functioning knowledge [1]. Therefore, the learning process needs to move beyond the traditional lecture format. As stated by many researchers [2], problem solving in software engineering is best learned through practice, and taught through examples. Having a teacher show a solution on the screen can go part of the way, but is never sufficient. Students therefore need to be provided with a significant number of assignments, work on them collaboratively and thus prepare for the work in software development teams. To allow for such a comprehensive learning, many courses include, or are complemented by, a project component in which students, usually in small teams, develop a medium-sized software application.

One of the main goals of a software engineering curriculum is to teach students how to exploit previous successful experiences and knowledge of other people in solving similar problems. This knowledge about successful solutions to recurring problems in software design is also known as software design patterns (DPs) [3]. Although software DPs are present in software engineering for over a decade, they are becoming increasingly important with the vision that diverse communities of experienced software practitioners, communicating mostly via the Internet, can share and collectively develop a set of design repertoires in the form of patterns.

However, the ability to find and use patterns decreases in proportion to the number of patterns documented in several different pattern forms [4] and stored in many online repositories, i.e. Yahoo! Design Pattern Library [5]), Portland Pattern Repository [6], Hillside.net Pattern Catalog [7]. In addition, frequently improper naming of software patterns (i.e., patterns should be named to explain what they do) makes it hard for students to find the requested patterns by using conventional search engines. Even more, students are often not aware that the solution to a specific software problem they are dealing with already exists and is described by a software pattern. Finally, apart from learning individual DPs and the principles behind them, students should learn how to understand and apply patterns they have not seen before, how to integrate different DPs, and how to use this knowledge in real-life situations.

This indicates a rising need for the social constructivist approach in software engineering educa-

* Accepted 2 September 2010.

tion, as well as intelligent educational services that provide students with right in time advices about learning resources and possible collaboration partners. In particular, an active learning paradigm is needed which recognizes that student activity is critical to the learning process. The basic philosophy of this paradigm [8] is to foster a deep understanding of a subject matter by engaging students in learning activities, not letting them be passive recipients of knowledge. Moreover, students should actively be involved in social interactions aimed at the knowledge construction and sharing in a given learning context.

Following this paradigm, and using active learning techniques, project-based learning and collaborative learning, we have developed an integrated learning environment for software DPs called DEPTHS (Design Patterns Teaching Help System) [9]. DEPTHS integrates an existing Learning Management System (LMS), a software modeling tool, diverse collaboration tools and relevant online repositories of software DPs. The LMS enables students to learn at the pace and in a place that best suits them providing them at the same time with a variety of learning activities and resources. The software modeling tool enables students to experience patterns-based software development in the context of real-world problems. Online repositories of software DPs provide students with a plenty of important resources on DPs containing both valuable examples of DPs and instructions how they should be used. Collaboration tools support different kinds of collaborative activities, such as discussions, collaborative tagging, and commenting. The integration of these different learning systems and tools into DEPTHS learning environment is achieved by using the Semantic Web technologies. Specifically, these technologies enabled us to formally represent and merge data about students interactions with the systems and tools integrated in DEPTHS. On top of that data, we have built context-aware educational services that are available throughout the DEPTHS environment. These services enrich and foster learning processes in DEPTHS in two main ways:

- recommendation of appropriate learning content (i.e., Web page(s), lessons or discussion forum threads describing software DP). We can recommend fine-grained learning resources, and make the recommendations aware of the learning needs recognized by our proposed learning environment.
- fostering informal learning activities by bringing together students and experts that are dealing with the same software problem or have experience in solving similar problems.

In this paper, we describe pedagogical background that this comprehensive learning environment is based on—project-based learning and collaborative learning, as well as context-aware educational support provided in the form of educational services available within each system and tool integrated into DEPTHS.

2. Background

Effective learning of software DPs requires a constructive approach to be applied in the teaching process. It is very important that students experience software development and use of DPs on real-world examples, in order to develop a deep understanding of basic principles behind them and to learn how to apply them in different situations. Having this in mind, we have explored a number of theories and research fields in the area of project-based and computer supported collaborative learning. We have identified the following three as the most important for teaching/learning software DPs: Learning through Design, Project-based learning (PBL) and Engagement theory. In addition, we have explored the literature related to the notion of context in the (e) learning domain and context awareness in online learning environments. In what follows, we first provide a brief overview of the three above mentioned pedagogical approaches and subsequently introduce the notion of context in (e) learning settings.

2.1 Relevant learning theories

In *learning through design*, students develop deep understanding of academic content by creating meaningful products that reflect their knowledge of the subject domain. These projects require that students not only learn the subject matter well enough to represent it in a final design, but also that they master the particular means of production used to create the product itself. Production means can include physical model-making, scale-model drawing, or the creation of software via programming and/or multimedia authoring. Acquiring these concrete design skills is therefore a crucial part of the learning gains that students make in such projects [10].

Project-based learning (PBL) is a teaching and learning model that organizes learning around projects. Projects comprise complex tasks and activities that involve students in a constructive investigation that results in knowledge building. Furthermore, learning activities should be long-term, interdisciplinary, and student-centered and must reflect a real world issues and practices. Engaging students in problems that are trivial for them or can be solved with the already acquired knowledge could not be

considered as PBL as they do not lead to the acquisition of new knowledge [11].

The engagement theory is based upon the idea of creating successful collaborative teams that work on tasks that are meaningful to someone outside the classroom [12]. Its core principles are summarized as 'Relate', which emphasizes characteristics such as communication and social skills that are involved in team effort; 'Create', which regards learning as a creative, purposeful activity; and 'Donate', which encourages learners to position their learning in terms of wider community involvement. Later research inspired by this approach, suggests a generic framework called *Genex framework* [13] that describes four phases a creative process will most likely pass through: 'Collect', which regards searching and browsing digital libraries, visualizing data and processes; as well as 'Relate', 'Create' and 'Donate' as explained in the engagement theory. These four phases do not form a linear path. Creative work may require returning to earlier phases and numerous iterations. Schneiderman [13] has identified eight activities that support creativity: 1) searching and browsing digital libraries, 2) consulting with peers and mentors, 3) visualizing data and processes, 4) thinking by free associations, 5) exploring solutions, 6) composing artifacts and performances, 7) reviewing and replaying session histories and 8) disseminating results.

Based on the guidelines for teaching software engineering to students [2, 14] and our own teaching experience, we believe that the presented theories provide a solid base for effective learning of software DPs. Accordingly; we based the DEPTHS framework on them and in the next section, we present how they are actually applied in DEPTHS.

2.2 Notion of learning context

During the last couple of years *context-aware learning* has gained a constantly increasing attention of the e-learning research community. However, the majority of traditional e-learning approaches ignores learning context and provides de-contextualized forms of learning.

The notion of context has been the subject of debates among researches in different domain areas. In the domain of learning, like in other domains, different authors have different interpretations of the term 'context'. Despite of this diversity, researchers seem to agree that a learning context is about the environment, tools, resources, people (in terms of social networking), and learning activities. Being more specific, context in learning systems is mostly characterized by the learners, learning resources and a set of learning activities that are performed in the light of a specific pedagogical approach [15].

Context-aware systems attempt to capture aspects of the user's current situation (including current personal state and current environment state) in order to improve the efficiency of interaction with the system. This could include improving precision of search results, proactive recommendations, and mediation of communication, among other things [16]. In learning settings, context-aware services could relate learning resources and learning activities to the student's needs, interests and competences, ensuring that these are useful and enjoyable. That way, engagement could be increased since the system provides appropriate information at the right moment. This further leads to a more compelling learning experience.

3. Project-based learning in DEPTHS

It is possible to develop many scenarios for learning software patterns in DEPTHS environment. However, a typical one is based on a project-based learning approach with collaborative learning support (Fig. 1). In this scenario, a teacher defines a specific software design problem that has to be solved in a workshop-like manner by performing several predefined tasks: brainstorming, creating and submitting solutions, evaluating solutions etc. These activities enable and even foster active learning that has strong foundation in the engagement theory and Genex's framework (see Section 2).

Brainstorming has foundation in two Genex's phases, *collect* and *relate*. First, a student is asked to read and analyze the problem provided by the teacher (Fig. 1, step 1). Afterward he presents his ideas about possible ways for solving the problem under study and to discuss and rate his peers' ideas. In order to get enough information to perform this task, he needs to search online repositories about software DPs and other related course content. DEPTHS makes this search more effective by providing semantically-enabled context-aware learning services for finding related online (Fig. 1B) and internally produced resources. Moreover, to get some initial directions on the performing task, the student uses a semantically-enabled service for finding peers (Fig. 1A) to discover people who have shared interests and are or have been engaged in similar problems. As we explain in the following section, both kinds of services are enabled by leveraging formally represented semantics of the learning context and learning resources (both online resources and those internally produced). Afterwards, the student has to find associations between the gained knowledge and the problem that has to be solved and to propose potential solution strategies. Later, consultations are directed at confirming the idea and refining it to accommodate criticisms.

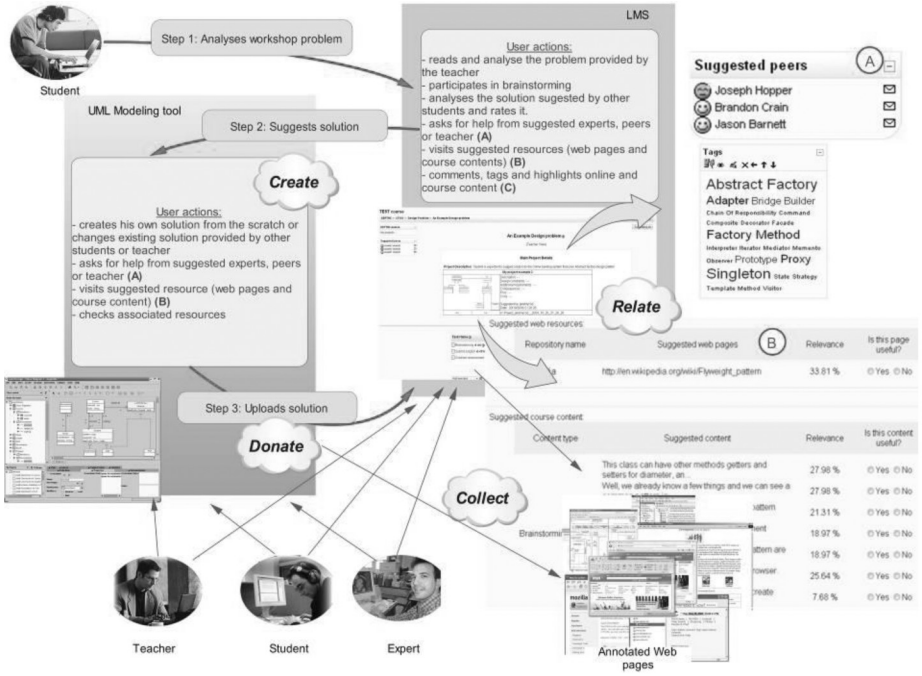

Fig. 1. An example learning scenario with DEPTHS: problem-based learning with collaborative learning support (titles in clouds indicate Genex's framework phases).

Genex's phase *create* is found in several DEPTHS activities, namely exploring earlier works (projects, discussions or brainstorming) on similar problems, creating design artifacts using software modeling tool or evaluating peers' solutions (Fig. 1-step 2).

Previous works on similar problems could be useful for students as they give them opportunities to learn from positive examples; and provide them with new facts, information, and an idea how to apply the same approach (design patterns) in a similar situation. Moreover, exploring previous works provokes critical thinking as it helps the student think about alternatives along with their advantages and disadvantages. DEPTHS context-aware learning services for discovery of relevant learning resources (both external and internal) greatly facilitate this task of exploring relevant previous work. These services are powered by semantic annotations of learning resources: ontologies enable capturing and formal representation of the semantics of the content of those resources, as well as the context of their creation and usage (see Section 4).

Having acquired the required knowledge, students should complete the deliverable using the software modeling tool. This kind of learning activity requires students to externalize their knowledge, to analyze possible solutions and to provide a design rationale.

After completing the project, students are asked to evaluate their own project, as well as to perform evaluation of each other's work. Students reflect critically on their own and others' contributions, and acquire knowledge about other possible solutions. This helps them recognize possible improvements in their own solutions. DEPTHS uses ontologies to capture the semantic of the students' evaluations, so that they can be used for recommendations as well as feedback provisioning.

Genex's *donate* component in DEPTHS stresses the benefits of having authentic deliverables that will be meaningful and useful to someone else (Fig. 1—step 3). All students' projects are published and publicly available; they are stored together with contextual semantic-rich metadata which facilitates their discovery and reuse. Students may be anxious that their work will be so visible, but it does seem to push them along to polish their projects. Moreover, students can learn from each other as portions of their projects became available before the final due date.

4. DEPTHS Architecture

Figure 2 illustrates the architecture of DEPTHS. It integrates existing, proven learning systems, tools and services in order to provide an effective collaborative environment for teaching and learning software DPs. In particular, DEPTHS currently integrates an LMS (Fig. 2A), a software modeling

tool (Fig. 2B), a feedback provisioning tool for teachers (Fig. 2C), a collaborative annotation tool (Fig. 2D), and online repositories of software patterns (Fig. 2E). This integration is achieved through a flexible underlying ontology-based framework called LOCO (Fig. 2F).

LOCO (Learning Object Context Ontologies) [17] is a generic framework capable of formally representing all particularities of the given learning context: the learning activity, the learning content that was used or produced, and the student(s) involved. Accordingly, the framework integrates a number of learning-related ontologies, such as learning context ontology, a user model ontology, and domain ontologies. These ontologies allows one to formally represent all the details of any given learning context, thus preserving its semantics in machine interpretable format and allowing for development of context-aware learning services. The LOCO ontologies are developed by following the Linked Data principles and best practices [18]. In particular, linkages were established with well-known Web ontologies such as the Dublin Core vocabulary, FOAF (Friend-Of-A-Friend, [19]), and SIOC (Semantically Interlinked Online Communities, [20]).

DEPTHS currently makes use of two ontologies of this framework: a domain ontology is used for representing the domain of software patterns, whereas the learning context ontology was extended to allow for an unambiguous representation of learning contexts specific to the systems, tools and services that DEPTHS integrates (the ontologies are available at the project's website: [21]).

The LOCO ontologies are used as the basis for storage and exchange of data among DEPTHS components. In particular, these ontologies underlie two DEPTHS repositories (Fig. 2J):

- *Repository of interaction data* stores data about students' interaction with learning content as well as their mutual interactions in the learning environment. The interaction data are stored in the form of RDF triples compliant with the learning context ontology of the LOCO framework (e.g., {loco: ContentItem} loco: hasUserEvaluation {loco: UserNote}) (loco—denotes the namespace of the Learning Context ontology of the LOCO framework).
- *Repository of LO metadata* stores semantic metadata about all kinds of learning objects (LO) used in the courses under study. This metadata formally defines the semantics of the learning content the metadata is attached to. These data are stored as RDF triples compliant with the LOCO's learning context ontology and the DEPTHS domain ontology of software DPs (e.g., {loco: ContentItem} loco: hasDomainTopic {dp: DesignPattern}) (dp—denotes the namespace of the domain ontology (i.e., ontology of software DP)).

DEPTHS also includes the *Repository of design artifacts* which uses open standard formats to store software artifacts created by students. The students' artifacts are stored in two formats: XML Metadata Interchange (XMI) and Scalable Vector Graphic (SVG). The former facilitates storing UML diagrams in the format suitable for later reuse in any

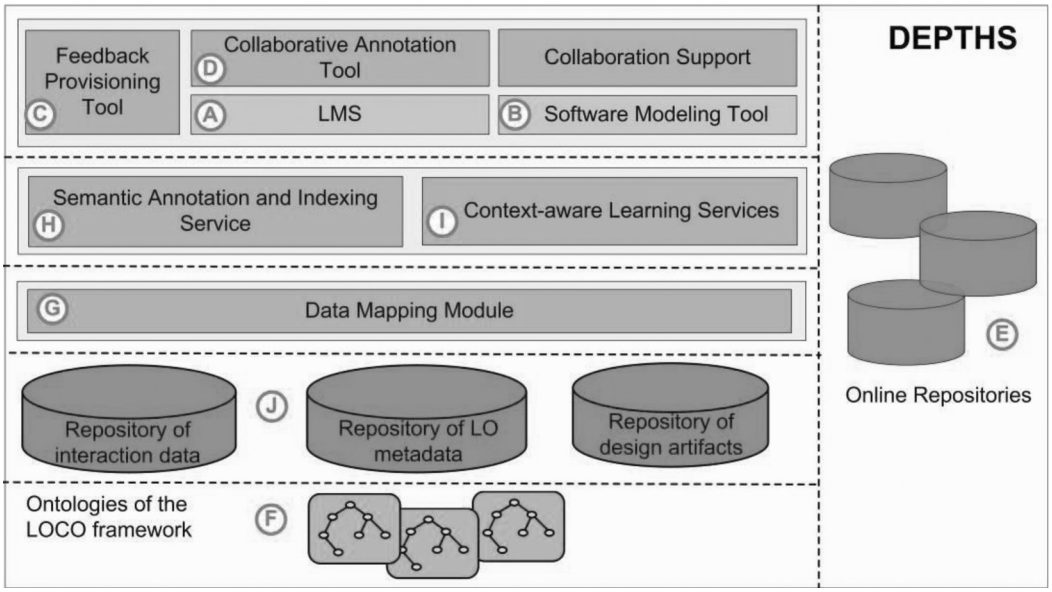

Fig. 2. DEPTHS architecture.

software modeling tool, whereas the latter is suitable for content presentation in a Web browser.

Since different learning systems, tools and services use different formats for representing and storing interaction data, DEPTHS integrates *Data Mapping Module* (Fig. 2G) which performs the mapping of those native data formats (e.g., the exchanged chat messages stored within the LMS's database, using a proprietary database schema) into RDF triples compliant with the LOCO's learning context ontology (e.g., {loco: ChatMessage} loco: sentBy {loco: User}). The resulting (RDF) data is stored in the *Repository of interaction data*. Data mapping is performed throughout each learning session in order to keep the semantic repository updated (with data about the interactions occurring during that session).

4.1 Educational services in DEPTHS

The next layer in the DEPTHS architecture consists of learning support services, namely Semantic Annotation and Indexing Service and Context-aware Learning Services.

Semantic Annotation and Indexing Service (Fig. 2H) is used for semantic annotation and indexing of online resources in publicly accessible repositories of DPs, as well as internally produced content (created by students) stored in the *Repository of design artifacts*. This module analyses text of each document in an online repository of DPs, recognizes specific topics defined in the domain ontology (i.e., software DP's name), finds what the document is about and how relevant it is for a specific DP. Here, we just briefly summarize how these computations are done.

First, DEPTHS's web crawler traverses through the structure of an online repository in order to collect URLs of all documents that can be annotated. Then, the Semantic Annotation Service is invoked to annotate semantically the collected documents using the concepts of the domain ontology (i.e., ontology of software DPs). Having tested many of the available tools for semantic annotation, we decided to use the KIM framework [22] that provides APIs for automatic semantic annotation of documents. However, the annotations produced by KIM APIs were not sufficient as we needed to know for each document what it is (primarily) about. To find the dominant software DP for each document (i.e., what the document is about), DEPTHS does indexing of the document. This assumes the use of proven statistical measures, namely term frequency-inverse document frequency (TF-IDF) [23] and cosine similarity [23]. TF-IDF is used to evaluate how important a word is to a document in a collection. This technique finds the most important DP in a specific document not only by taking into account its instances within the document, but also by considering its instances in other documents in the corpus. Thus, the document will be more important for a DP if that DP occurs many times within a small number of documents, whereas it would be considered less important if the DP occurs fewer times in that document, occurs in many documents or occurs in virtually all documents. This way, a collection of relevant documents is created for each DP. Afterwards, cosine similarity compares each document with an imaginary document (whose relevance is ideal), and sorts the documents in the collection based on their similarity, i.e. their relevance for specific pattern.

Semantic annotation of the internally produced content (e.g., chat messages, discussion forum messages and ideas) is performed in the similar way, immediately after a user creates a content unit (i.e., following the event of submitting a new content unit to the system). Additionally, each recognized term (DP label) in the content is changed to the hyperlink, used for launching web-resource and internal content finding services.

Context-aware learning services (Fig. 2I) are accessible to all systems and tools integrated in the DEPTHS framework and are exposed to end users (students) as context-aware learning features. They are based on Semantic web technologies, and include (but are not limited to):

- *Web resource finding*. Based on the student's current learning context, this service generates a list of recommended Web resources from publicly accessible repositories of software DPs. To do this, it computes the relevance of each resource (i.e., Web page) available from these repositories for the student's current learning context and selects the most relevant pages for the student. The computation of relevancy of a Web resource is based on two kinds of semantic metadata: 1) the semantic metadata assigned to the resource by the Semantic Annotation and Indexing Service (i.e. {dp:WebPage} dp:isRelevantFor {dp:DesignPattern}) and 2) the formal representation of the student's current learning context represented in accordance with the LOCO' learning ontology (e.g., {loco:LearningContext} loco:userRef {loco:User}; {loco:LearningContext} loco:contentRef {dp:DesignProblem}). When evaluating the relevancy of already crawled and annotated Web resources, DEPTHS also makes use of students estimation of the resource's relevance for the current learning context (each time a student visits a suggested Web resource he is asked to rate its relevance for the given context). Students' positive and negative ratings affect the resource's overall rating according to the influence factor

(value between 0 and 1) defined by the teacher or the system administrator.

- *Discovery of relevant internally produced resources.* This service suggests internally created resources (e.g., discussion threads, brainstorming notes, and project description) that could be useful for a student to solve a problem at hand in the given learning context. The computation of relevance is done in a similar manner to the one applied for external, Web resources.
- *Experts, teachers and peers discovery.* Based on the current learning context, this service suggests other students or experts as possible collaborators. Collaborators are selected and sorted using an algorithm which considers their competences on three different levels (Fig. 3): same content (i.e. current software problem), similar or related learning content (i.e. similar software problem) and broader content (i.e. software problem in the same course). Estimation of the peer's competence on each level is performed through assessing three types of competence indicators:
  - Participation in learning activities (i.e. brainstorming, submitting or assessing peers' works). Each activity has different impact factor on student competences that is defined in the system itself but could be changed by the teacher.
  - Knowledge level estimated by the teacher and other peers' evaluations, including projects evaluations and ideas ratings. However, not do all ratings have the same influence on knowledge level estimation. For example, a high mark given by a student with high competences on the given topic has more impact on final knowledge level appraisal than a high mark given by a student with average or low competences.
  - Social connections with the peer asking for help—the stronger social connection with a specific person, the more suitable that person is for help provision. We believe that an already appointed social connection could be much more successful and effective than new connections with people one does not know.

  All this data is collected through the queries performed on both DEPTHS's semantic repositories. Each type of competence indicator has different influence on the overall competence. This influence depends on the influence factors assigned to them (default systems' values could be changed by the teacher).

  Potential collaborators can be from the DEPTHS environment, but can also be located through the problem solving community portals and relations established via peers.
- *Context-based semantic relatedness.* This service is used by all other services, as it allows for: i) computing context-based semantic relatedness between tags that students define and/or use in the given learning contexts; ii) connecting students' tags with appropriate concepts of the domain ontology (i.e. disambiguation of the tags with the domain concepts) [23]; iii) resolution

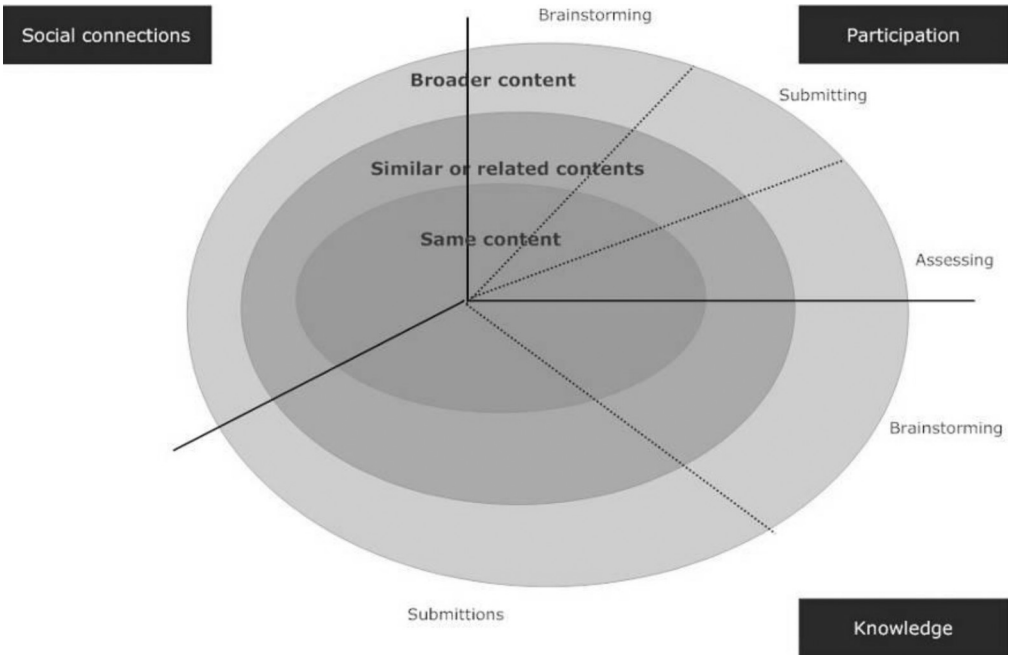

Fig. 3. Factors affecting estimation of potential collaborators' competences in DEPTHS.

of students' queries containing both, tags and domain concepts relevant for the given learning context. This service connects tags with the concepts of the domain ontology as well as resources annotated with these concepts and stored in the DEPTHS semantic repositories (repository of LO metadata and repository of interaction data). That way the system selects only appropriate tags to show in the given moment based on the current learning context, and connects these tags with appropriate domain concepts related to them.

4.2 DEPTHS implementation

We have implemented DEPTHS by leveraging open-source solutions and extending them with Semantic web technologies. Specifically, we have integrated Moodle LMS [25], ArgoUML [26] software modeling tool, OATS (Open Annotation and Tagging System) tool for collaborative tagging and highlighting [27] and LOCO-Analyst tool to provide teachers with feedback regarding students' activities [17]. Moreover, we use semantic annotation services of the KIM framework [27] and Sesame server [28] for semantic repositories. In order to provide students with context-aware educational services of the DEPTHS framework, we have extended both Moodle and ArgoUML so that they can make use of these services. Moreover, we have developed a Moodle module that supports project-based collaborative learning, that is, it supports and integrates in DEPTHS several kinds of collaborative activities such as brainstorming, submitting and assessing projects. Coupled with ArgoUML and educational services in DEPTHS it provides effective learning of software DPs, as described in Section 3 (screenshots used in Fig 1 are taken from this implementation).

5. Evaluation

The evaluation of DEPTHS was conducted in February 2009, in the context of a course that the first author of this paper taught at the Department of Computer Science of the Military Academy in Belgrade, Serbia. DEPTHS was evaluated with a group of 13 students of the fifth year of the computer science program who took part in our course on software development. The students already had some elementary knowledge in the domain of software DPs, but they were not familiar with the particular software DPs used in this experiment (Facade, Adapter, Strategy, Composite, Visitor and Decorator).

As the number of participants was limited by the number of students, we decided to use 'blocked designed' method for creating groups for evaluation [29]. We categorized students based on the teacher's subjective opinion about their knowledge in the domain of software development, previous results in the courses related to software engineering, leader's capabilities and communicative skills. Afterward, we formed 4 approximately equivalent groups (3 groups with 3 students and 1 group with 4 students). The size of the groups is based on our belief and teaching experience that work in small size groups (3 or 4 students) is a necessity for effective education of software engineers.

The aim of the evaluation was to determine how effective DEPTHS is for learning DPs. Specifically, we wanted to evaluate the perceived usefulness of the engagement theory, implemented in DEPTHS, in software design pattern education. Moreover, we wanted to check if active student's involvement in real world problems and the employment of context-aware educational services could ensure more effective way of imparting knowledge in the domain of software development.

Before the experiment started, a demonstration of DEPTHS functionalities along with a training using a task similar to the one used in the experiment, were performed with students. Each group was assigned a different task (i.e., a software design problem). Students were asked to suggest solutions and evaluate each others' solutions within one week period of time. Actually, project organization used in the experiment was based on the learning workflow described in Section 3.

We used an interview to collect data about the students' satisfaction with and attitudes towards learning with the DEPTHS system. The interview was also supposed to reveal the students' perceptions regarding the effectiveness of learning with DEPTHS. The questions were divided into three sections based on the type of information we were interested in. The first section (14 questions) gathered data regarding the students' previous experience with computer-assisted learning. The questions of the second section (15 questions) were related to the DEPTHS system and the third section (11 questions) was aimed at evaluating the learning program on software DPs offered by DEPTHS system. Most of the questions (33) were multiple-choice questions (based on a Likert-like scale) with 5 possible answers, ranging from 1 (most negative) to 5 (most positive). There were 6 open-ended questions and 1 combined (multiple choice and open-ended).

We used three methodologies to analyze the results gained in this experiment. First, we analyzed the results of the overall corpus of the students using standard descriptive statistic instruments such as frequency, mean, median, and average. The second kind of analysis consisted of comparing the groups

of students that were derived by splitting the results data based on the students' answers on the questions from the first section of the interview. Finally, we used Pearson's chi-square test to find if there is significant association between different variables. We used SPSS tool [30] to process data and analyze the results.

Having analyzed the results, we found that the majority of students (84.62%) have experience in using Internet to find relevant information, collaborate with colleagues on solving common tasks (53.85%) and use tools for message exchange and discussion (84.62%). However, they have far less experience with online learning tools (only 23.07% are familiar with e-learning tools) and using the Internet to find peers for solving problems (only 38.46% answered positively).

The DEPTHS system received high marks from the students. Majority of them (53.85%) reported that they have learned as effectively as in traditional way, and 30.77% reported that they have learned more than in traditional way. The students reported it was intuitive and very easy to use (76.92%), but they also have reported some technical issues. These issues were caused by the software bug that caused problems in uploading UML diagrams to the repository, and have been resolved one day after the beginning of the evaluation. We believe that this issue could have affected students' confidence in the system. The students felt educational services provided in DEPTHS are very helpful: Web resource recommending service—92,30%; course content recommending service—84,61%; and peers recommending service—76,92%. They also thought that the activities provided within the tasks considerably contribute to the learning process (brainstorming — 76.92%, and evaluating each other's works—100%).

Having analyzed the trends of the different groups of students based on their answers on the first group of questions, we found that all students that have taken a course offered through an LMS consider educational services provided in DEPTHS as very useful. Students that have used computers to find relevant collaborator for domain that they are currently working on, have much more positive attitude for brainstorming tool in DEPTHS and learning from other students ideas. There is no significant difference between those students who are familiar with e-learning and those who are not with respect to their experience of learning in DEPTHS environment. However, it is interesting to notice that all students who are not familiar with e-learning gave the highest mark for educational service for Web pages recommending.

We have identified 27 variables' pairs that have significant association through the use of Pearson's chi-square (we used standard level of significance $p<0.05$). For example, there is a significant association between the students' answers to the question if they used tools for sending messages and discussions, and the question if they think that the other students' ideas were useful. Many useful conclusions could be drawn from these associations about how close various variable impacts might be on student achievement. For example, we found that students' satisfaction with the educational services for recommending relevant Web pages and course content affected their latter satisfaction with their learning results using this program. We believe that this association is strongly related with the students' level of adoption of these services as very important and useful part of a learning system. However, the deeper analysis of these results is out of scope of this paper.

6. Related work

The framework proposed in this paper is closely related to two research fields: collaborative learning in the domain of software engineering and context-aware learning. Even though extensive work has been done in both research fields, to the best of our knowledge there were very few attempts in developing collaborative learning environments that support knowledge creation and sharing through the collaborative learning process based on the active learning principles.

The approach proposed in [31] presents an intelligent tutoring system, called COLLECT-UML, the goal of which is to support the acquisition of both problem-solving skills and collaboration skills. In this environment, students construct UML class diagrams that satisfy a given set of requirements. COLLECT-UML supports collaborative learning and provides feedback on both collaboration issues and task-oriented issues. Our framework uses a similar approach to the learning process, that is, students learn through the practical problem-based examples in collaboration with other students. However, our framework offers higher learning potential as it provides access to the relevant learning resources, sharing and commenting of produced learning artifacts, and facilitates context-aware learning (i.e., context-aware retrieval of formal and informal learning content, and recommendation of peers). Another work presented in [32] describes OMT-Editor, a collaborative learning environment for object-oriented design problems using Object Modeling Technique (OMT), a precursor of UML. However, communication interface requires students to begin each contribution with a suggestive phrase, or sentence opener, such as, 'I think', 'Please show me', etc., in order to provide the system information about their intent. This restric-

tion significantly confines the collaboration among participants to the set of integrated phrases.

In [33], the authors suggested an approach similar to the one presented in this work. They have developed MICE—a learner-centered platform for regulating learners' programming styles when studying a programming language using an integrated development environment. It also integrates an LMS and a set of tools for communication and collaboration among users. Even though MICE follows a similar approach to integration of existing tools, it still lacks access to online resources that is available in our framework. Besides our framework promises additional support for collaborative learning as it offers social tagging support.

One of the main objectives of the EU project APOSDLE is to develop a system that would be able to provide knowledge workers with learning resources relevant for their present work context [34]. In particular, based on the immediate work context of a user, the system should identify his/her missing competencies and learning needs and suggest appropriate learning resources. These learning resources are created on-the-fly from a variety of resources (documents, videos, expert profiles, and so on) already stored in the workplace and may be in the form of learning material or suggestions to contact experts and/or colleagues. The system's functionality is based primarily on its knowledge base that stores an integrated representation of various kinds of knowledge (e.g., domain, task, and instructional). Knowledge integration and advanced search and retrieval capabilities (associative information retrieval) are enabled by the Semantic Web technologies. Obviously, this approach has a lot of commonalities with the one we suggested in this paper. Nonetheless, having grounded our approach in pedagogical theories and best practices of collaborative learning, we can expect to provide students with better learning experience.

An e-learning framework proposed in [35] supports recommendation of peers based on a student's context. The student's context is defined as a result of the interaction of three key elements: the knowledge potential, the social proximity and the technical access. Comparing to the DEPTHS approach to recommendation of peers, the approach presented in [35] is advantageous as it considers technical context that includes factors that may influence e-learning such as technical media or time proximity. However, unlike the approach proposed in DEPTHS, that approach does not consider the influence of student's participation in the learning activities on his competences to help other students.

7. Conclusions

Collaborative learning through project-based work helps students reflect on their learning experiences in ways that promote abstraction from experience, explanation of results, and understanding of conditions of DPs applicability in real world situations; it also provides the experience of working in software development teams. Following this paradigm, we have developed a learning environment for software DPs which leverages semantic technologies to integrate several existing learning systems and tools, and provide context-aware educational services that together allow for effective learning of software DPs. Our present implementation and first evaluation results convince us that this environment could significantly contribute to effective teaching and learning of DPs. Semantic Web technologies facilitate development of beneficial educational services that makes search for relevant resources and possible peers fast and effective.

We are encouraged with the results of the initial evaluation study that show very positive students' attitude toward learning in DEPTHS learning environment. Students' perception of system's usefulness is valuable and encouraging for our further research. However, the results we got still do not have a statistical power, as the participants' sample was too small. Further research is required that would include sufficient participants to ensure the general applicability of the findings. In addition, in our future work we intend to do a more precise evaluation of each specific educational service as well as a quantitative evaluation of the students' learning effectiveness.

References

1. J. B. Biggs, Teaching for quality learning at university, *Buckingham: Open University Press/Society for Research into Higher Education*. (Second edition), 2003.
2. M. Jazayeri, The Education of a Software Engineer, In: *Proceedings of the 19th IEEE International Conference on Automated Software Engineering*, 2004, pp. xviii–xxvii,
3. E. Gamma, R. Helm, R. Johnson and J. Vlissides, *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley, Reading, MA, 1995.
4. L. Rising, *The Patterns Handbook: Techniques, Strategies, and Applications*, Cambridge University Press, New York, 1998.
5. Yahoo! Design Pattern Library, http://developer.yahoo.com/ypatterns/, Accessed March 31, 2010.
6. Portland Pattern Repository, http://c2.com/ppr/, Accessed March 31, 2010.
7. Hillside.net Pattern Catalog, http://www.hillside.net/patterns/, Accessed March 31, 2010.
8. L. Warren, Migrating to a Teaching Style that Facilitates Active Learning. *CiLTHE Stage 1 Dissertation*, Lancaster University, 2002.
9. Z. Jeremic, J. Jovanovic and D. Gasevic, Towards a Semantic-rich Collaborative Environment for Learning Software Patterns. In: *Proceedings of the 3rd European Conference on*

Technology Enhanced Learning, Maastricht, The Netherlands, September 16–19 2008, pp. 155–166.
10. J. L. Kolodner, D. Crismond, J. Gray, J. Holbrook and S. Puntembakar, Learning by Design from Theory to Practice, In: *Proceedings of International Conference of the Learning Sciences*, Georgia Tech, Atlanta, USA, December 16–19 1998, pp. 16–22.
11. A. Bongio, et al. Towards a Collaborative Open Environment of Project-Centred Learning, Nejdl, W., Tochtermann, K. (eds.) In: *EC-TEL 2006. LNCS,* **4227**, Springer, Heidelberg, 2006, pp. 561–566.
12. G. Kearsley and B. Schneiderman, Engagement theory: A framework for technology-based learning and teaching, Originally at http://home.sprynet.com/~gkearsley/engage.htm, Accessed March 27, 2010.
13. B. Shneiderman, Creating Creativity: User Interfaces for Supporting Innovation, *ACM Transaction on Computer-Human Interaction*, **7**(1), 2000, pp. 114–138.
14. D. Bagert, T. Hilbum, G. Hislop, M. Lutz, M. McCracken and S. Mengel, Guidelines for Software Engineering Education, Version 1.0. *Technical Report CMU/SEI-99-TR-032,* Software Engineering Institute, Carnegie Mellon University, Pittsburgh PA, 1999.
15. M. Siadaty, C. Torniai, D. Gasevic, J. Jovanovic, T. Eap and M. Hatala, m-LOCO: An Ontology-based Framework for Context-Aware Mobile Learning, In: *Proceedings of the 6th International Workshop on Ontologies and Semantic Web for Intelligent Educational Systems @ 9th Int'l Conf. on Intelligent Tutoring Systems*, 2008.
16. S. Braun, A. Schmidt and C. Hentschel, Semantic Desktop Systems for Context Awareness—Requirements and Architectural Implications, In: *1st Workshop on Architecture, Design, and Implementation of the SemanticDesktop (SemDesk Design), 4th European Semantic Web Conference (ESWC2007)*, Innsbruck, Austria, 2007.
17. J. Jovanovic, D. Gasevic, C. Brooks, V. Devedžic, M. Hatala, T. Eap and G. Richards, Using Semantic Web Technologies for the Analysis of Learning Content, *IEEE Internet Computing*, **11**(5), 2007, pp. 45–53.
18. How to Publish Linked Data on the Web, http://www4.wiwiss.fu-berlin.de/bizer/pub/LinkedDataTutorial/, Accessed March 31, 2010.
19. FOAF Vocabulary specification 0.97, http://xmlns.com/foaf/0.1, Accessed March 31, 2010.
20. SIOC (Semantically-Interlinked Online Communities), http://sioc-project.org/, Accessed March 31, 2010.
21. DEPTHS (Design Patterns Teaching Help System), www.learningdesignpatterns.org, Accessed March 31, 2010.
22. The KIM Platform: Knowledge & Information Management, http://www.ontotext.com/kim/index.html, Accessed March 31, 2010.
23. C. D. Manning, P. Raghavan and H. Schütze, Introduction to Information Retrieval, *Cambridge University Press*, 2008.
24. C. Torniai, J. Jovanovic, D. Gasevic, S. Bateman and M. Hatala, E-Learning Meets the Social Semantic Web, In: *Proceedings of 8th IEEE International Conference on Advanced Learning Technologies*, Santader, Spain, July 2008, pp. 389-393.
25. Moodle, http://moodle.org, Accessed March 31, 2010.
26. ArgoUML, Open Source Software Engineering Tools, http://argouml.tigris.org, Accessed March 31, 2010.
27. The Open Annotation and Tagging System (OATS), http://ihelp.usask.ca/OATS, Accessed March 31, 2010.
28. Sesame, Accessed March 31, 2010.
29. W. D. Crano and M. B. Brewer, *Principles and Methods of Social Research* (2nd edn). Mahwah, NJ: Lawrence Erlbaum Associates, 2002.
30. SPSS, www.spss.com, Accessed March 31, 2010.
31. N. Baghaei, A. Mitrovic and W. Irwin, Supporting collaborative learning and problem-solving in a constraint-based CSCL environment for UML class diagrams, *International Journal of CSCL*, 2(2–3), Springer, New York, 2007, pp. 150–190.
32. A. L. Soller, Supporting Social Interaction in an Intelligent Collaborative Learning System, *International Journal of Artificial Intelligence in Education*, **12**(1), 2001, pp. 40–62.
33. J. Jovanovic, S. Rao, D. Gasevic, V. Devedžic and M. Hatala, An Ontological Framework for Educational Feedback, In: *Proceedings of the 5th International Workshop on Ontologies and Semantic Web for Intelligent Distributed Educational Systems*, California, USA, July 9–13 2007, pp. 54–64.
34. C. Ghidini, V. Pammer, P. Scheir, L. Serafini and S. Lindstaedt, APOSDLE: Learn@work with semantic web technology, *In 'I-Know '07*, Graz, Austria, September 5–7, 2007.
35. Z. Yanlin and Y. Yoneo, A Framework of Context Awareness support for peer recommendation in the e-learning context, *British Journal of Educational Technology*, **38**(2), 2007, pp. 197–210.

Zoran Jeremic is an Assistant Professor of Software Engineering in the Department of Simulations and Distance learning at the Military Academy in Belgrade, Serbia. He received his B.S. degree from the Military academy, and his M.Sc and PhD degrees from the University of Belgrade. His research interests are in the areas of semantic technologies, Web technologies, software engineering, technology-enhanced learning, and personalized learning. He is a member of the GOOD OLD AI research network. He can be reached at http://zoranjeremic.org.

Jelena Jovanovic is an Assistant Professor of Computer Science with the Department of Software Engineering, FON—School of Business Organization, University of Belgrade, Belgrade, Serbia. She received her B.S., M.Sc. and PhD degrees in informatics and software engineering from University of Belgrade in 2003, 2005 and 2007, respectively. Her research interests are in the areas of semantic technologies, Web technologies, technology-enhanced learning, and personalized learning. She is a member of the GOOD OLD AI research network. She can be reached at http://jelenajovanovic.net.

Dragan Gasevic is a Canada Research Chair in Semantic Technologies and an Associate Professor in the School of Computing and Information Systems at Athabasca University. He is also an Adjunct Professor in the School of Interactive Arts and Technology at Simon Fraser University and an associated research member of the GOOD OLD AI Research Network at the University of Belgrade. He is a recipient of Alberta Ingenuity's 2008 New Faculty Award. His research interests include semantic technologies, software language engineering, technology-enhanced learning, and service-oriented architectures. He can be reached at http://dgasevic.athabascau.ca.