

ZORAN JEREMIC

UNIVERSITY OF BELGRADE, SERBIA

PROJECT-BASED COLLABORATIVE LEARNING OF SOFTWARE PATTERNS

Abstract: Teaching and learning software design patterns (DPs) is not an easy task. Apart from learning individual DPs and the principle behind them, students should learn how to apply them in real-life situations. Therefore, to make the learning process of DPs effective, it is necessary to include a project component in which students, usually in small teams, develop a medium-sized software application. Following this paradigm, and using active learning techniques, project-based learning and collaborative learning, we have developed a learning environment for software DPs which leverages semantic technologies to integrate several existing learning systems and tools.

Key words: collaborative learning, project-based learning, Semantic Web, software patterns.

1 INTRODUCTION

The major concern of today's software engineering education is to provide students with the skills necessary to integrate theory and practice; to have them recognize the importance of modeling and appreciate the value of a good design; and to provide them with ability to acquire special domain knowledge beyond the computing discipline for the purposes of supporting software development in specific domain areas. Software engineering students should learn how to solve different kinds of software problems both on their own and as members of a development team. As stated by many researchers [Jazayeri, 2004], 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 must be given a significant number of assignments and work on them collaboratively and thus prepares for the work in software development teams.

In addition, it is essential that students learn 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) [Gamma et al, 1995]. 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.

Apart from learning individual DPs and the principle 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. 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.

All the above mentioned specificities of learning software DPs indicate the need for the social constructivist approach in software engineering education. In particular, an active learning paradigm is needed which recognizes that student activity is critical to the learning process. The basic philosophy of active learning paradigm [Warren, 2002] is to foster deep understanding of subject matter by engaging students in learning activities, not letting them be passive recipients of knowledge. Moreover, the students are involved in the knowledge construction and sharing through social interactions in the 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) [Jeremic et al, 2008]. DEPTHS integrates an existing Learning Management System (LMS), a domain specific tool for software modeling, diverse collaboration tools and relevant online repositories of software DPs. 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 domain specific tool enables students to experience patterns-based software development in the context of real-world problems. Online repositories of software DPs provide students with 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, commenting etc. To enable the integration of these different learning systems and tools in a comprehensive learning environment, we have used the Semantic Web technologies. In this paper, we described pedagogical background that this framework is based on project-based learning and collaborative learning.

2 LITERATURE REVIEW

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 (LTD), Project-based learning (PBL) and Engagement theory.

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 [Kolodner, 1998].

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.

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 [Kearsley et al, 1999]. 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 genetic framework called “Genex framework” [Schneiderman, 2000], that describes four phases a creative process will most likely pass through, “Collect”, which regards searching and browsing digital libraries, visualizing data and processes, “Relate”, “Create” and “Donate”. These four phases do not form a linear path. Creative work may require returning to earlier phases and numerous iterations. Schneiderman [Schneiderman, 2000] 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.

In the following section, we present how these instructional approaches are applied in DEPTHS.

3 PROJECT-BASED LEARNING IN DEPTHS

A typical scenario for learning software patterns with DEPTHS assumes a project-based learning approach with collaborative learning support (Figure 1). In

particular, 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.

Brainstorming has foundation in two Genex's phases, *collect* and *relate* (see Section 2). First, the student is asked to present his ideas about the possible ways for solving problem, discuss his peers' ideas and rate them. 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 (Figure 1B) and internally produced resources. Moreover, to get some initial directions on the performing task, the student uses semantically enabled peers finding service (Figure 1A) to find people who have shared interests and are 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). Afterward, 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.

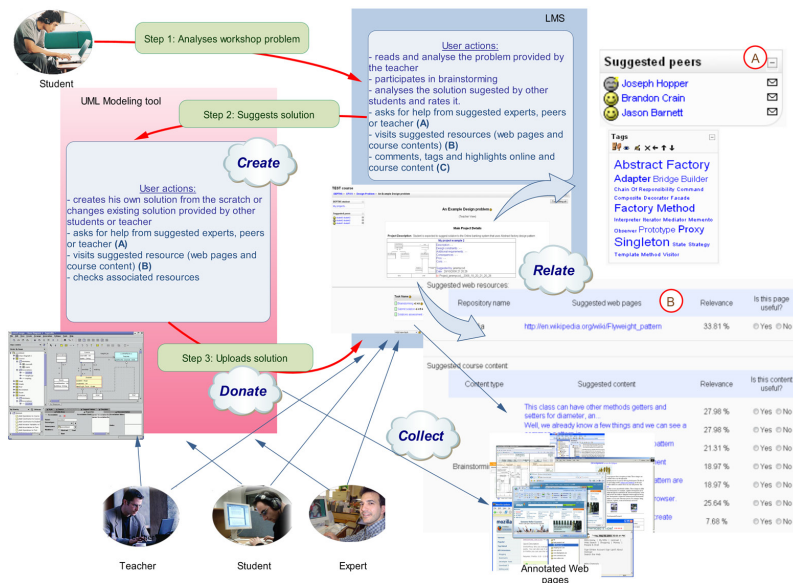


Figure 1. An example learning scenario with DEPTHS: problem-based learning with collaborative learning support (titles in clouds indicate Genex's framework phases)

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.

Earlier works on similar problems could be useful for students as it gives them the opportunity to learn from positive examples, provides them with new facts and information as well as an idea how to apply the same approach (design patterns) in a similar situation. Moreover, exploring previous works provokes critical thinking as it helps student to think about alternatives – their advantages and disadvantages. DEPTHS context-aware learning services for discovery of relevant learning resources (both external and internal) greatly facilitates 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. 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 ONTOLOGICAL FOUNDATION

DEPTHS is based on the Learning Object Context Ontology (LOCO) ontological framework [Jovanovic et al, 2007]. LOCO allows one to formally represent the notion of learning context which is defined as a specific learning situation, determined by the learning activity, the learning content, and the student(s) involved. To support DEPTHS, we use ontologies of the LOCO framework to interrelate information about learning objects, learning activities and learners collected from various tools relevant for learning software patterns, as specified in the introduction.

The core part of the LOCO framework is the LOCO-Cite ontology, which comprises a number of classes and properties aimed at formally representing learning

context. In addition, this framework integrates a number of learning-related ontologies, such as:

- User Model ontology – enables one to model the participants in a learning process.
- Learning Design ontology – formally represents the basic building blocks of an instructional design; inspired by the IMS Learning Design specification¹.
- Learning Object Content Structure ontology – allows for explicitly defining the structure of a learning object with the goal of making each of its components directly accessible, and thus reusable.
- Domain ontology – formally represents the subject matter of the learning content.

To address the requirements of the DEPTHS framework, we fully adopted the LOCO-Cite ontology, and connected it with the ontology of software patterns domain.

4.1 THE LOCO-CITE ONTOLOGY

The LOCO-Cite ontology allows for semantic representation of the data about a student's overall interactions with learning content and other students during different learning activities. Based on this data, DEPTHS can perform context-aware retrieval of software patterns resources from online repositories and its own repository of software artifacts (which may also contain artifacts produced by other students and shared by the community); identify and draw students' attention to the related threads in discussion forums; and identify peers that could help in a specific situation.

The LOCO-Cite ontology is based on the notion of learning context, represented with the *LearningObjectContext* class (Figure 2). This class is, in accordance with our definition of the learning context, related to the activity (an instance of the *Activity* class) that a learner or a teacher (an instance of the *um:User2* class) undertook while interacting with a learning content (an instance of the *ContentItem* class).

Activities are very important part of the learning process in DEPTHS as they lead to realization of learning objectives. Examples of such activities are reading lessons, visiting online pages, participating in a workshop or doing an assignment, solving design problems, quizzing and collaborating with other participants. In the LOCO-Cite ontology, these activities are recognized and grouped as three basic types of activities: reading, doing an assessment, and collaborating. However, we found that the LOCO-Cite ontology does not allow for capturing and representation of some specific types of activities and events typically occurring within software modeling tools. Accordingly, we decided to extend this ontology with a set of classes and properties, which would enable collecting data about user interactions in software development tools, in a similar way as described in [Jovanovic et al, 2007b].

Each instance of the *LearningObjectContext* class has a reference to an appropriate instance of the *ContentItem* class (Figure 2), which represents any kind of con-

tent available in a learning environment (e.g. a lesson, a diagram, and a discussion forum post). In order to annotate semantically a piece of learning content containing a software pattern, we established a relation between the *ContentItem* class and classes from the domain ontology through the *Metadata* class and its property *dc:subject3*.

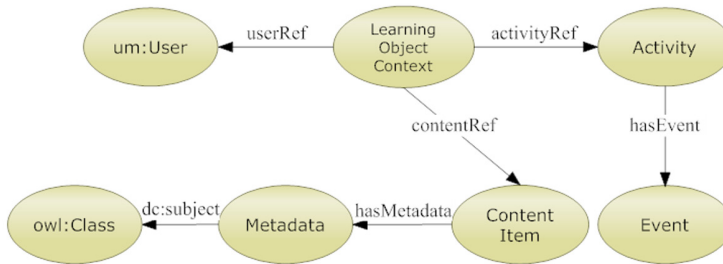


Figure 2. The LOCO-Cite ontology (basic classes and relations)

4.2 DOMAIN ONTOLOGY

Since DEPTHS is devised as an environment for teaching/learning software patterns, it leverages an ontology of software patterns as its domain ontology. DEPTHS uses this ontology to annotate relevant resources and extract metadata that is subsequently used for finding resources appropriate for a student's current learning context. On the other hand, it also annotates the products of learning activities, such as chat messages or discussion posts. In this way, DEPTHS can easily connect these products of learning activities with learning resources, and use this information to further improve its context-aware support by being able to mash-up knowledge scattered in different activities.

Rather than developing new design pattern ontology from scratch, we decided to (re)use an existing ontology. Among the available ontologies of the design patterns domain [Dietrich, 2005],[Kampffmeyer, 2007],[Montero, 2003],[Henninger, 2007], we have chosen the set of ontologies suggested in [Henninger, 2007] to serve as the domain ontology of the DEPTHS framework. Comparing these with the other ontologies, we found that they provide a very intuitive and concise way to describe design patterns and patterns collections, and more information on usability knowledge and the contextual factors that impact this knowledge. This approach to pattern representation has the ability to federate distributed pattern collections. These ontologies include a set of pattern forms (e.g., Coplien Form [Coplien, 1996], Gang of Four Form [Gamma et al, 1995]) arranged in an inheritance hierarchy.

The central point of this ontology is the *PLForms* class which is the common super class for all other pattern forms (Figure 3). It can be easily extended with new pattern forms, by adding new subclasses in this hierarchy. For example, *CoplienForm* is defined as a subclass of *EssentialForm* (which is in turn subclass of the *PLForms*

class). *CoplienForm* inherits *EssentialForm*'s properties (e.g., *hasAuthor*, *hasContext*, and *hasProblem*) and adds some new properties (e.g. *hasRationale*). Of course, it is important to note that we are currently experimenting with this ontology, and any other ontology of software patterns may be used instead of or in collaboration with this one.

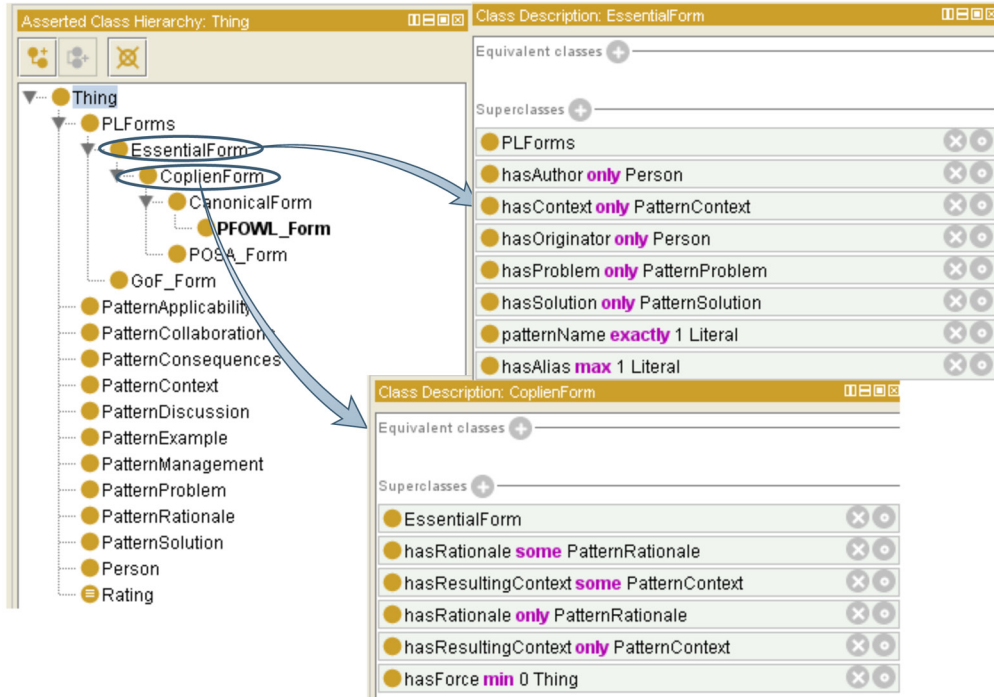


Figure 3. The class structure of the Software Patterns ontology

5 SYSTEM ARCHITECTURE

In this section, we present a high-level architecture of our DEPTHS framework. The framework comprises five basic components: a Learning Management System (LMS), a Collaborative Learning Modeling tool, a Teachers' Feedback tool, Online Repositories of Software patterns and a Semantic Management System (Figure 4). In the rest of the section, each of these components is addressed in turn.

5.1. LEARNING MANAGEMENT SYSTEM

Today's LMSs have an extensive set of tools and features aimed at facilitating the learning process (e.g., quiz, assignment, chat room, discussion forum, and glossary). However, they do not fully address the requirements of a comprehensive learn-

ing framework such as DEPTHS. One of those requirements is the integration of the usage tracking data from all the systems/tools students use. We address this requirement with the LOCO framework (as explained in Section 4). As most of the existing LMSs use classical databases for data storage, it is necessary to transform the data stored in their databases into semantically enriched data compliant with the LOCO-Cite ontology. The transformed data is stored in the **Repository of Learning Object Contexts (LOCs)** which is fully based on the LOCO-Cite ontology.

Apart from the existing collaborative learning support that is usual in most LMSs (such as discussion forums and chat-rooms), we found that it would be very useful if student(s) had a tool for collaborative annotation of learning content (such as tagging, commenting and highlighting). The more the content is annotated, the easier it becomes to later find it and retrieve it. This is also in the line of leveraging of potentials of the emerging area of social computing and approaches such as collaborative tagging. Accordingly, we decided to integrate such a collaborative tool in the LMS that is used in our framework.

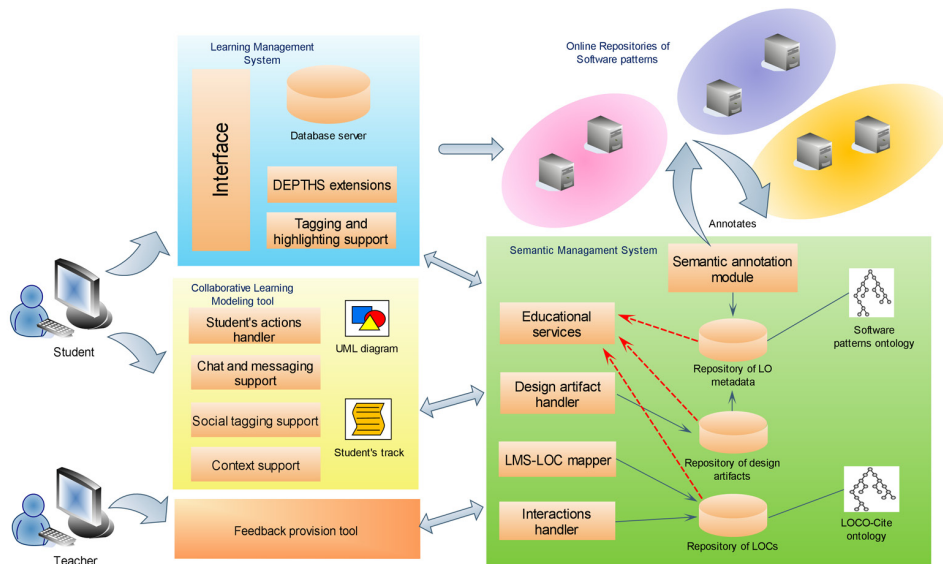


Figure 4. The architecture of the DEPTHS learning framework

5.2. COLLABORATIVE LEARNING MODELING TOOL

We have identified that the framework for software engineering education should necessary have the support for software modeling using diagrams, especially UML (Unified Modeling Language) diagrams. However, most of the existing software modeling tools

does not provide all necessary support for collaborative learning. We refer here to the set of features that should be supported by these tools, beside those that they usually include:

- An easy way for presenting a description of the suggested solution (i.e. software documentation that contains valuable information of both functional and non-functional requirements, information related to the problem domain).
- Collaborative Tagging support module enabling students to create either social (public) or private annotations of learning content (e.g., publicly available online resources on the Web, design diagrams, and forum messages). That way, a student begins to create a network of content that can be later accessed and searched, for example, through a tag cloud view.
- A chat room and messaging tools that support collaboration with other students even if they are not using the same tool in the given moment (e.g. some are using the modeling tool, whereas the others are working within the LMS).
- Ability to keep track of students' actions during learning sessions (**Student's actions handler**). These tracks are sent to the **Interactions handler** (see section 5.5) which is responsible for integrating them into the **Repository of LOCs** where they are stored for later analysis.
- Context-aware learning. Based on a student's learning context the system should suggest him/her the most suitable learning objects, publicly available online resources on the Web, similar problems, or discussion threads that could be useful for the specific problem he/she is facing. This should basically help students to better comprehend relations between the theoretical knowledge that they have learned and experiences of others with the practical problems at hand.

5.3. FEEDBACK PROVISIONING TOOL

In order to help a teacher to improve the learning experience of his/her students, DEPTHS incorporates a tool that provides teachers with feedback about all kinds of activities their students performed during the learning session. This tool is also built on top of the LOCO framework, so that it has access to the learning context data created in all learning tools used in DEPTHS. The feedback tool provides teachers with contextualized feedback and relevant information about students' learning.

5.4. ONLINE REPOSITORIES OF SOFTWARE PATTERNS

One of the main advantages of this framework is that it leverages existing online learning resources, rather than requiring teachers to create new ones. There are a plenty of such repositories which could be used, such as Yahoo! Design Pattern Library⁴, Portland Pattern Repository⁵, and Hillside.net Pattern Catalog⁶.

DEPTHS leverages the domain ontology to provide both teachers and students with resources from these repositories that are relevant for the current teaching/

learning context. In particular, the domain ontology is used for annotating semantically the resources available from these repositories and the resulting semantic metadata is stored in the **Repository of LO Metadata** (Figure 4). This metadata is used for indentifying the resources relevant for any given learning situation. An additional advantage of DEPTHs is that these resources are made accessible from which ever tool of the DEPTHs framework a teacher or a student is using. Moreover, students are able to tagg and highlight these resource. We believe that these content annotation activities can improve some typical activities in learning (e.g. revisiting learning material, personal note taking, and connecting with peers), but moreover, they provide valuable data that can quaranty higher quality of context-aware learning.

5.5. SEMANTIC MANAGEMENT SYSTEM

This module is the integration point of the whole framework. In particular, it leverages the semantic web technologies to support integration of all the above mentioned modules. In order to acomplish this, it uses a set of repositories and a set of software compentents. In particular, it comprises the following three reporitories:

Repository of LO metadata stores semantic metadata about online resources available from online repositories, as well as about internally created content, such as software design diagrams, discussion forum postings and chat messages besides regular LOs used in the courses under study. This metadata consists of topics⁷ defined in the software pattern ontology and we refer to it as semantic metadata as it formally defines the semantics of the learning content it is attached to.

Repository of design artifacts keeps students' solutions in different formats: a format appropriate for their latter reuse, and another one that is suitable for their presentation in the LMS.

Repository of LOCs stores learning objects' context-related data in accordance with the LOCO-Cite ontology.

In addition, Semantic Management System integrates the following components:

LMS-LOC mapper has the role to transform data from the LMS database of logs of learners's activities into the format compliant with the LOCO-Cite ontology and to store the resulting data in the Repository of LOCs. This data mapping is performed throughout each learning session in order to keep the semantic repository updated (with data about the events occuring durign that session).

Semantic annotation module is used for annotating online repositories of design patterns, as well as, diagrams (created by students) stored in the Repository of design artifacts. This module automatically extracts metadata based on the domain ontology and stores them in the Repository of LO metadata.

Design artifact handler manages the diagrams in the Repository of design artifacts. It takes diagrams from the modeling tool and stores them in the Repository of

design artifacts in different formats. It is also responsible for keeping track of different versions of the same diagram.

Educational services provide all necessary support for context-aware learning and are accessible from all tools integrated in the DEPTHS framework. These services are based on Semantic web technologies, and include (but not limited to):

- *Web resource finding.* Based on the student's current LOCO context, DEPTHS suggests Web resources from publicly accessible repositories of software DPs that it finds relevant for the student's given context. The relevance of a resource is determined through an algorithm which combines topic relatedness and collaborative filtering (i.e., students' previous ratings of the resource's relevance for the given context).
- *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 LOCO context. The computation of relevancy is done in a similar manner to the one applied for external, Web resources.
- *Experts, teachers and peers discovery.* Based on the current LOCO context, DEPTHS suggests other students or experts as possible collaborators. Collaborators are selected and sorted using an algorithm which considers their relevance for the current problem, relevance for the current course and relevance for the related LOCO contexts. 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 LOCO contexts [Tornia et al, 2008]; ii) connecting students' tags with appropriate concepts of the domain ontology (i.e. disambiguation of the tags with the domain concepts); iii) resolution of students' queries containing both, tags and domain concepts relevant for the given learning context.

6 IMPLEMENTATION OF DEPTHS

In this section, we describe the tools that we are using to implement the proposed framework and argument our decision to use specifically these tools.

6.1 LEARNING MANAGEMENT SYSTEM

As the LMS component of the DEPTHS framework (Section 5.1), we have decided to use Moodle8 LMS for many reasons. First, Moodle is a popular open-source LMS, which requires only hosting costs, and thus provides us with an inexpensive but

reliable learning environment. Moodle has an extensive set of tools and features. In particular, it includes: assignments, chats, choices, forums, glossaries, journals, labels, lessons, quizzes, resources, surveys and workshop modules. In addition, a comprehensive evaluation of this LMS (including aspects related to virtual course management, user friendliness, authors' learning curve and an economic analysis) gave very positive results [McMullin, 2004]. Another aspect determining our decision is related to the open source nature of this system. This feature is important for us because we want to extend Moodle with Semantic web technologies.

One of the most eminent advantages of Moodle that influenced our decision is that it facilitates collaborative work, that is, it is designed under the social constructivist theory. This theory argues for a student-centered environment where learners are able to work independently, reflect on their own work and on the work of other students, while at the same time being connected to a group of learners with whom they can share ideas and reflect on each other's work [Dougiamas, 1998]. As we indicated in the introduction, getting the students involved in the learning process is essential to effective learning of software patterns.

However, the manner in which Moodle stores data about students' interactions with the system is inappropriate for DEPTHS. Rather, DEPTHS requires semantically enhanced interactions data, that is, RDF data stored in a format compliant with the LOCO-Cite ontology. In order to resolve this issue, during the initialization of the system, we do the mapping of the interactions data stored in Moodle's database into the required ontological format and store the resulting RDF data into the Repository of LOCs (see Section 5.5). This task is performed using D2RQ9 – an open source platform that facilitates the mapping of relational databases into ontological models. The D2RQ Mapping Language is a declarative mapping language aimed for defining mappings between a database schema and an RDFS vocabulary or an OWL ontology. This way a lot of valuable data that currently resides in Moodle database are made accessible to DEPTHS in the form of RDF statements. Later, throughout each learning session, DEPTHS uses Sesame10 Java API to update the semantic repository with data about the events occurring during that session. Sesame is an open source Java framework for storing and querying RDF data and RDF Schema based reasoning. Apart from updating the Repository of LOCs, DEPTHS uses Sesame API to query this repository in order to retrieve the data required by its educational services (see Section 5.5). As two distinct technologies are used (PHP for Moodle and Java for DEPTHS), we use PHP/Java bridge11 to provide the connection.

We have decided to integrate OATS (The Open Annotation and Tagging System) [Bateman et al, 2006] in Moodle in order to provide students with a tool to collaboratively create and share knowledge, by using highlights, tags and notes. OATS is an open source tool which was created to further enrich the functionalities provided by an LMS. The aim is to motivate students to engage more with the learning content by facilitating self-organization. It is expected that the application of self-organization

principles will help empower students to move beyond passive consumption of e-learning content towards active production [Fischer et al, 2002]. We have already made an extension of the LOCO-Cite ontology to enable formal representation of events occurring in collaborative content annotation tools such as OATS and we intend to use it to capture and store data about students interactions with OATS into the Repository of LOCs.

This way DEPTHS employs Moodles advantages but also adds some new possibilities provided as DEPTHS's educational services, among which the most important are: finding web resources that could be useful in the current learning context, finding relevant internally produced resources stored in DEPTHS repositories, and finding appropriate peers.

6.2 SEMANTIC ANNOTATION OF LEARNING CONTENT

One of the functionalities provided by the DEPTHS framework is to suggest to students appropriate online resource about design patterns. In addition, it is capable of suggesting relevant pieces of content produced during the learning process (such as software models produced by students and messages exchanged in discussion forums). The realization of these functionalities depends on the availability of a critical mass of metadata for the learning content, linked to formally represented knowledge about the design patterns.

Among many available tools for content annotation that we have tested, we decided to use the KIM framework¹², a semantic annotation platform, that provides an automatic semantic annotation, indexing and retrieval of documents. Semantic annotation in KIM is based on the PROTON¹³ ontology. We have extended PROTON with our domain ontology (i.e. the ontology of software patterns, see Section 4.2) in order to make KIM aware of the concepts from the software patterns domain. As a result, we use KIM annotation facilities to automatically annotate diverse kinds of learning artifacts with relevant domain topics. This further facilitates semantic interlinking of diverse kinds of learning artefacts: online resources, lessons from the LMS, students software models, and exchanges messages. Thus, enables us to integrate previously fragmented knowledge artifacts students used or created in learning activities.

6.3 DOMAIN MODELING TOOL

ArgoUML is a Computer-Aided Software Engineering (CASE) tool suitable for the analysis and design of object-oriented software systems. It allows for designing all kinds of UML diagrams. Another advantage of ArgoUML is that it supports open software standards: XML Metadata Interchange (XMI) and Scalable Vector Graphic (SVG). The former facilitates exchange of UML diagrams among students, whereas the latter is suitable for content presentation in Moodle.

Due to its open-source nature, ArgoUML can be extended to enable capturing of user interaction data and storing that data in the common ontological format of the

DEPTHS framework (i.e. the LOCO-Cite ontology). Actually, we are currently working on this extension. Specifically, we are using Sesame Java API to extend ArgoUML, so that it continually updates the Repository of LOCs with data about events generated during students' interactions with the tool. Subsequently, we are going to extend ArgoUML, so that it can make use of the DEPTHS's educational services, such as finding solutions to the similar problems suggested by other students or finding appropriate online resource about design patterns that could be used in the student' current context.

6.4 FEEDBACK PROVISION TOOL

Finally, we have decided to integrate LOCO-Analyst in the DEPTHS framework. This tool provides teachers with feedback regarding all kinds of activities their students performed during a specific period of time [Jovanovic et al, 2007]. It is built on top of the LOCO framework so we can easily use it in this framework without any intervention on it. It provides teacher with comprehensive reports and relevant information about different aspects of students' learning.

7 CONCLUSION

Collaborative learning through project-based work provides motivating activities that keep students' attention. It is an effective way of helping them to 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. By working in the projects, students get the experience of working in software development teams on a real world application and how to apply the skills that they have learned when confronted with real software problem.

Following this paradigm, we have developed a learning environment for software DPs which leverages semantic technologies to integrate several existing learning systems and tools. We believe that this environment could significantly contribute to effective teaching and learning of DPs. This approach is tied directly to a pedagogical approach that promotes active learning. Semantic Web technologies provide beneficial educational services that makes search for relevant resources and possible peers fast and effective. Additionally, these services make learning inventive and at the same time more productive and enjoyable.

However, our beliefs are not based on the evaluation results. So far, we had only few students using DEPTHS. Their feedback has been very positive, but the number is too small to draw conclusions. For a more comprehensive validation, we are planning to perform a comprehensive evaluation study with the group of software engineering students at the Belgrade University.

8 REFERENCES:

1. [Bateman et al, 2006] Bateman, S., Farzan, R., Brusilovsky, P., McCalla, G.: OATS: The Open Annotation and Tagging System. In the Proceedings of the 3rd Annual International Scientific Conference of the LO Repository Research Network, Montreal (2006)
2. [Coplien, 1996] Coplien, J. O.: Software Patterns. SIGS Books, New York (1996)
3. [Dietrich, 2005] Dietrich, J., Elgar, C.: A formal Description of Design Patterns using OWL. In Proceedings of ASWEC, IEEE Comp. Soc. (2005)
4. [Dougiamas, 1998] Dougiamas, M.: A journey into constructivism. (1998) <http://dougiamas.com/writing/constructivism.html>
5. [Fischer et al, 2002] Fischer, G., Ostwald, J.: Transcending the Information Given: Designing learning Environments for Informed Participation. In the Proc. of ICCE Int'l Conf. on Comp. in Educ., New Zealand (2002)
6. [Gamma et al, 1995] E. Gamma, R. Helm, R. Johnson, J. Vlissides, "Design Patterns: Elements of Reusable Object-Oriented Software", Addison-Wesley, Reading, MA, 1995.
7. [Henninger, 2007] Henninger, S.: A Framework for Flexible and Executable Usability Patterns Standards. 31st IEEE Software Engineering Workshop (SEW-31), pp.23—34. USA (2007)
8. [Jazayeri, 2004] M.Jazayeri, "The Education of a Software Engineer", Proceedings of the 19th IEEE international conference on Automated software engineering (ASE '04), IEEE Comp. Society, USA, 2004, pp 18-xxvii.
9. [Jeremic et al, 2008] Z. Jeremic, J. Jovanovic, D. Gasevic, "Towards a Semantic-rich Collaborative Environment for Learning Software Patterns", Proceedings of the 3rd European Conference on Technology Enhanced Learning, Maastricht, The Netherlands, 2008, pp. 155-166.
10. [Jovanovic et al, 2007] J. Jovanović, D. Gašević, C. Brooks, V. Devedžić, M. Hatala, T. Eap, G. Richards, "Using Semantic Web Technologies for the Analysis of Learning Content", IEEE Internet Computing, Vol. 11, No. 5, 2007.[Jovanovic et al, 2007b] Jovanović, J., Rao, S., Gašević, D., Devedžić, V., Hatala, M.: An Ontological Framework for Educational Feedback. In Proc. of the 5th Int'l Workshop on Ontologies and Semantic Web for Intelligent Distributed Educational Systems, pp. 54-64. USA (2007)
11. [Kampffmeyer, 2007] Kampffmeyer, H., Zschaler, S.: Finding the Pattern You Need: The Design Pattern Intent Ontology. In Proc. of 10th International Conference on Model Driven Engineering Languages and Systems, (MoDELS 2007), pp.211--225. USA (2007)

12. [Kearsley et al, 1999] G. Kearsley, & B. Schneiderman, "Engagement theory: A framework for technology-based learning and teaching", 1999, Originally at <http://home.sprynet.com/~gkearsley/engage.htm>
13. [Kolodner et al, 1998] J.L. Kolodner, D. Crismond, J. Gray, J. Holbrook, & S. Puntembakar, "Learning by Design from Theory to Practice", Proceedings International Conference of the Learning Sciences ,98, 1998, pp.16-22. <http://www.cc.gatech.edu/projects/lbd/htmlpubs/lbdtheorytoprac.html>
14. [McMullin, 2004] McMullin, B., Munro, M.: Moodle at DCU. Dublin City University, Office of the Dean of Teaching and Learning (2004) <http://odtl.dcu.ie/wp/2004/odtl-2004-01.html>
15. [Montero, 2003] Montero, S., Diaz, P., Aedo, I.: Formalization of web design patterns using ontologies. In Proc. of 1st Int'l Atlantic Web Intell. Conf. (AW-IC), pp. 179--188. Spain (2003)
16. [Shneiderman, 2000] B. Shneiderman, "Creating Creativity: User Interfaces for Supporting Innovation", ACM Press, Transactions of Computer-Human Interaction, Vol. 7, No. 1, 2000, pp 114 – 138.
17. [Torniai et al, 2008] C. Torniai, J. Jovanovic, D. Gasevic, S. Batemen, M. Hatala, "E-Learning Meets the Social Semantic Web", The 8th IEEE Int'l Conf. on Advanced Learning Techn. (ICALT 2008), Spain, July, 2008.
18. [Warren, 2002] I. Warren, "Migrating to a Teaching Style that Facilitates Active Learning", CiLTHE Stage 1 Dissertation, Lancaster University, 2002.

ENDNOTES

1. <http://www.imsglobal.org/learningdesign/>
2. *um* identifies the namespace of the user model ontology
3. *dc* stands for the namespace of the Dublin Core metadata schema
4. <http://developer.yahoo.com/ypatterns/index.php>
5. <http://c2.com/ppr/>
6. <http://www.hillside.net/patterns/onlinepatterncatalog.htm>
7. To be more precise this metadata consists of instances of the concepts defined in the ontology of software patterns.
8. <http://moodle.org/>
9. <http://www4.wiwiss.fu-berlin.de/bizer/D2RQ/spec/>
10. <http://www.openrdf.org/>
11. <http://php-java-bridge.sourceforge.net/pjb/>
12. <http://www.ontotext.com/kim/index.html>
13. <http://proton.semanticweb.org/>