

Eliciting design patterns for E-Learning Systems

S. Retalis¹, P. Georgiakakis¹, Y.Dimitriadis²

¹University of Piraeus,
Department of Technology Education and Digital Systems
80 Karaoli & Dimitriou
185 34 Piraeus, Greece
Tel: 0030 210 414 2765
{[retal](mailto:retal@unipi.gr), [geopet](mailto:geopet@unipi.gr)}@unipi.gr

²University of Valladolid
Department of Signal Theory, Com. and Telematics Engineering
Camino del Cementerio s/n
47011 Valladolid, Spain
Tel: 0034 983 423666
yannis@tel.uva.es

Eliciting design patterns for E-Learning Systems

Abstract

Design pattern creation, especially in the e-learning domain, is a highly complex process that has not been sufficiently studied and formalized. In this paper, we propose a systematic pattern development cycle, whose most important aspects focus on reverse engineering of existing systems in order to elicit features that are cross-validated through the use of appropriate, authentic scenarios. On the other hand, an iterative pattern process is proposed that takes advantage of multiple data sources, thus emphasizing a holistic view of the teaching-learning processes. The proposed schema of pattern mining has been extensively validated for Asynchronous Network Supported Collaborative Learning (ANSCL) systems, as well as for other types of tools in a variety of scenarios, with promising results.

Keywords: E-learning design pattern elicitation, design pattern development, authentic learning scenarios.

Motivation

The introduction of networked technologies in the fields of education and training is not new. E-Learning is an established educational paradigm which has stopped being the poor sibling of education. A lot of initiatives, stemming from both academia and industry have emerged. As an effect a lot of systems (e.g. Learning Management Systems, etc.) are being engineered today in increasing numbers by various institutions and companies or even by open source initiatives (see Moodle, Claroline, etc.) to support teachers and learners in performing computer-supported pedagogical scenarios (Horton & Horton 2003). The design and implementation of computer-supported pedagogical scenarios includes planning the educational

situation and selecting and deciding on the e-learning systems or tools that will be used in the actual scenario. The learning results of a scenario will largely depend on the “quality” of the employed e-learning systems.

E-learning systems are basically sophisticated and complex web-based applications. They offer different services that empower learners, improve learning, connect learners to peer and domain experts as well as resources supportive of their needs, and to integrate learning with performance and individual with organisational goals (Goodyear 2001). Their design and implementation is not an easy task, since they incorporate a variety of organizational, administrative, instructional and technological components (Moore & Kearsley, 1996; Carlson, 1998). Therefore systematic, disciplined approaches must be devised in order to leverage the complexity and assortment of the systems and achieve overall product quality within specific time and budget limits.

Nevertheless, even though the construction of such systems has been taking place for many years, they are still designed and developed from scratch. As a consequence, systems of similar genre (e.g. learning management systems, groupware systems, etc.) might contain the same set of features but vary in the usability aspect. We agree with the remark made by Lehtinen, et. al. (1999) about CSCL systems: “It is not only the features of the applied technology but especially the way of implementation of the technology which support student collaboration” (p. 18). Usable e-learning systems should be designed taking into account a combination of issues that concern pedagogical orientation, learning task characteristics, as well as choices and constraints of learning settings.

Thus, developers need advice from experts, experienced peers and users so as to avoid “re-inventing the wheel”. Advice, which is too prescriptive, or reliant on just one model, can stifle the sort of innovation that can make the most of evolving technology (Goodyear et al., 2004). Thus, researchers and practitioners are attracted to the potential of design patterns for

facilitating the capturing and sharing of aspects of engineering expertise, for representing successful models for the implementation of systems as well as for encoding known usable solutions which have been tested by users (both learners and educators).

Design patterns are all about reusability, which seems to be the keyword in achieving the economies of scale for building affordable and usable systems. This is a recurrent objective of e-learning systems, that focus on reusability, tailorability, etc. while at the same time aim to meet market and scales requirements (Roschelle, J., et al., 1998).

Design patterns, according to Alexander's work (Alexander, 1997), capture both (i) problems which recur in their environment and (ii) design guidance. That is, they should be abstractions based on empirical observation of recurring phenomena in learning environments. The empirical aspect of design pattern construction is of great importance – especially with regard to convincing practitioners of their validity.

Design patterns are not created by a certain person. Pattern writing is in reality a team effort. Patterns are usually drafted by someone and then shared, analyzed, commented, assessed-evaluated in action and refined through an extended process of collaboration; the well-known process of shepherding (Mezaros & Doble, 1998).

Although the process of design pattern creation is well documented (e.g. the pattern writing language (Mezaros & Doble, 1998)), how is a pattern conceived? More importantly, how is a pattern language constructed? As Alexander said in his famous keynote speech at OOPSLA'96 "...we were always looking for the capacity of a pattern language to generate coherence, and that was the most vital test used, again and again, during the process of creating a language. The language was always seen as a whole" (Alexander, 1996). Thus patterns' construction in isolation does not fit into the scope of this paper. Especially in the e-learning domain, we are looking for patterns that form structured sets. E-learning systems' design pattern languages should be related to real functional structure in the ordinary technical and practical

sense, and simultaneously be related to intrinsic values of convivial learning places. This feature is closely related to the holistic view that should be applicable to learning patterns. For example, when developing an e-learning system which will offer functionality for online debates the designers should be aware of the pedagogical issues related to online debates in order to specify the appropriate set of features. In (ELEN 2004), the pattern “DEBATE” contains the rationale for using the pedagogical strategy of online debate as well as offers a solution for building a system that will support this strategy. During the online debate, users need usable mechanisms for annotating messages exchanged, as the pattern ANNOTATION_ON_POSTED_MESSAGES explains which is shown in Figure 1 (TELL, 2005b).

[Insert Figure 1 here]

The main question that this paper tries to answer is how a pattern language for e-learning systems is constructed. Certainly not in a big bang! The aim of this paper is to propose an approach for identifying design patterns for e-learning systems. This approach is both bottom-up and top-down. It is more or less a process of reverse-engineering the systems that embed good design in order to make that design explicit, and be able to communicate it to other designers, so that it becomes common practice. It also suggests that patterns are discovered or mined after numerous implementations of learning scenarios in authentic places, usually by different people. These scenarios are pedagogically sound and designed considering the multitude of variables that may affect the instructional process, such as those found in collaborative learning” (NISE, 1997) (Hernández et al, 2005). The proposed approach has been evaluated in practice for the construction of a pattern language for asynchronous network supported collaborative systems. Details about this language can be found in (TELL, 2005, Georgiakakis & Retalis, 2006).

The paper is structured as follows: Section 2 describes how patterns can be used for designing web-based systems. E-learning systems are a specific genre of web-based systems, the

proposed approach encourages the adoption and adaptation of DPs in the e-learning context from other disciplines, especially Human Computer Interaction (HCI) and hypermedia systems. Section 3 illustrates the design pattern mining approach and Section 4 explains in more detail the first step, and more difficulty one, i.e. the identification of a pattern. We explain this step by making references to a case study made for validating the proposed approach. More specifically, we show how this approach can be applied for eliciting patterns for Asynchronous Network Supported Collaborative Learning (ANSCL) systems. Finally, Section 5 makes references to similar suggestions for discovering design patterns and discusses the innovative issues of the proposed approach.

Design Patterns for Web-based Systems

Design practices can codify best design solutions, making them explicit and available to a whole community of developers and users, so that it can become common practice. Designers of new or existing systems, especially inexperienced ones, can take advantage of design expertise, past experience and save precious time and resources in their effort to build usable systems.

The history of design patterns and their proliferation is well known and broadly documented. It all began in the field of built architecture, when Christopher Alexander invented the idea of capturing design guidelines in the form of design patterns (Alexander et al., 1977, Alexander, 1996). The ‘Alexandrian’ patterns found many followers in the computer science discipline, especially after the so-called ‘GOF’ book for object-oriented design (Gamma et al., 1994). Some of the fields that have adopted patterns are: hypermedia engineering (Garrido, 1997, Garzotto, 1999), Human-Computer Interaction (HCI, 2005), pedagogy (Fincher, 1999, Lyardet et al., 1999, Carlson, 1998, Haberman, 2006, Bennedsen and Eriksen, 2006) and e-learning (Avgeriou et al., 2003, ELEN, 2004).

Design patterns are not ready-made pluggable solutions. They have to be carefully selected and possibly adapted to the particular system's needs. One of the factors that play a crucial role, in addition to objective/purpose, is the user. For designing any web application it is important to know who the users are and what they are coming for. Different users have quite different needs. Since design patterns codify core solutions to problems that recur in different contexts, a designer will have to adapt the solution to the given context.

When designing an e-learning system, we have to consider our students' lives, how we cause their work to be distributed over time and space, the conviviality (or otherwise) of the learning environments in which they find themselves, and the stresses and tensions we visit upon them. Alexander mentioned that "Bad buildings create stress. Good building nurtures". The same happens with e-learning systems "Bad technology exacerbates the tensions in students' lives. Usable technology, which meets the needs of learners and educators ,provides ways of softening a harsh environment" (Goodyear et. al., 2004). Of course, the development of e-learning systems is definitely pedagogical driven rather than technology driven.

As a consequence, design pattern languages for e-learning systems are unique. Also they should be regarded as such. These languages could adopt issues from HCI design patterns since usability is of prime importance. Nevertheless, e-learning system design patterns are a new means for effective, efficient and transferable instructional design in technology enhanced learning environments. Creating patterns for e-learning systems thus signifies an explicit commitment to a set of values which characterise well-being, well-learning, and a good social and physical convivial learning environment. In this attempt, we exploit knowledge of patterns from other domains, but also take into account the particularities of learning, e-learning, etc.

Design Pattern Elicitation approach

Forming a design pattern language for e-learning systems is a collaborative activity, usually extending over several years. It involves iterative work, working in two directions. From the bottom up, we can sketch individual design patterns, to capture recurrent problems and solutions from our collective experience as e-learning practitioners, interpreting these also through the lens of research-based evidence and theory. From the top down, we can try to structure the problem space of design, scoping out the largest and smallest patterns, and sketching relationships between patterns (written and unwritten). The main innovative aspect of the proposed approach is the various techniques and sources used when identifying the idea of a design pattern and sketching it. We do reverse engineering of well known systems of specific genre thus finding out details about the features of these systems. Nevertheless, we analyse the way users (learners and educators) use the systems in practice within authentic learning scenarios in order to see which features best support the needs the users, as well how they are interrelated. Moreover, we also check how instructional designers suggest the way pedagogical strategies should be implemented.

The stepwise proposed approach is described in Figure 2. It contains the following steps:

- Identifying and sketching design patterns
- Drafting a design pattern
- Critiquing patterns
- Identifying related patterns (e.g. patterns needed to complete and embellish an existing pattern).

[Insert Figure 2 here]

Although for drafting and making criticism on identified patterns one can look at well known approaches published in the literature (e.g. Mezaros et al., 1998), the most difficult step of the pattern language development process is, perhaps, the identification of a pattern and its relationships with other ones. General ideas for patterns can be derived from:

- Experts' (instructional designers and system designers) experience
- Observation of user tasks
- In-depth analysis of the functionality of e-learning systems
- Review of the literature about pedagogical strategies
- User log files analysis of e-learning systems
- Study of other patterns already published in areas such as HCI.

It can be easily seen in the aforementioned list that multiple data sources can be used, mostly of qualitative nature (observations, interviews, etc.). Not all sources could be available (e.g. log files). However, we emphasize the need for a holistic, situated view of e-learning patterns, which have to be studied within carefully chosen and authentic scenarios. Information coming from these data sources should be triangulated in order to be able to reach credible proposals. On the other hand, these multiple sources are revisited in several phases of the complete lifecycle in order to have a second, more thorough view, as well as collecting more data that could reinforce the proposed patterns.

Input from the aforementioned sources can be priceless material that can generate new ideas for design patterns. These thoughts must then be drafted in a more formal way; a draft of a pattern should contain:

- A name that can be meaningful and easy to remember covers the content and if possible gives rise to associations that are related to the described problem and solution.

- A description of the recurrent problem.
- Its context, like its empirical background, evidence for its validity.
- An analysis explaining what makes the specific problem a problem, and why a solution is needed.
- The forces that play a role in coming to a solution.
- The solution, the field of physical and social relationships that are required based in the stated context.
- Known solutions based on existing practices, drawn from theory, or stemming from well-known ideas in the literature.

Verification and evaluation of each pattern has to be done once it is drafted. Designers and developers must thoroughly evaluate and review it by implementing it in practice and measuring its success. The process of “shepherding” is also a way of improving the production of patterns since expert designers provide guidance to the pattern writer in order to refine and actually perfect the pattern.

A pattern cannot be stand isolated. Once drafted and/or reviewed, it should be related to other patterns already written or just captured as thumbnails (Harrison & Coplien 2002). Patterns are interrelated in several different ways. There are some fundamental relationships distinguishable (Welie, 2003):

- Aggregation: We can find this relationship when a pattern aggregates several other patterns. This is a form of ‘has-a’ relationship.
- Specialization: It concerns patterns that are a more specific version of another version. This is a form of ‘is-a’ relationship.

- Association: A pattern may be associated to other patterns because they also often occur in the larger context of the design problem, or because the patterns are alternatives for the same problem.

The proposed approach has been extensively validated for Asynchronous Network Supported Collaborative Learning (ANSCL) systems, as well as for other types of systems such as Wiki systems, Learning Object Repositories and Synchronous Conferencing Systems. This validation process was performed by the Greek and Spanish partner of an EU funded project, called TELL (TELL 2005a). The TELL project team made a methodical and systematic effort to support the design of new effective technological tools for collaborative learning.

In all the above cases, the cold-start problem was the most crucial one: how can one identify an idea for a pattern or a pattern language which meet all the “functional requirements”: capture of practice, abstraction, value system, structuring principle, and presentational form (Fincher & Utting 2002)?

Solving the cold-start problem: The identification of a design pattern

A design pattern should not only provide facts. Instead it should “narrate” with the most descriptive way the experience it tries to convey. A pattern should help its stakeholders to: comprehend existing systems; customize systems to fit user needs; construct new systems. Pattern developers should also focus on practicability of their patterns since description of proven solutions to recurring design problems is the aim.

In the e-learning domain, pattern languages should guide the design of convivial learning environments, where tangible objects and people sit alongside virtual computer-based spaces,

objects and interactions. The construction of convivial e-learning environments is a demanding task, which requires creativity and a significant amount of expertise about theoretical issues of both learning task design, and human computer interaction besides other disciplines that are strongly involved in this domain (e.g. cognitive sciences, etc).

In order to identify the design patterns for this specific kind of system one can use the opportunistic method. It takes as its starting point an observation about an actual situation, i.e. analysis of the functionality of existing systems, and then tries to pin down exactly the quality of these systems over a series of scenarios. On the other hand, one can start by using a top-down approach. This could be by brainstorming within a group of experienced practitioners.

For identifying a pattern, which is the most innovative aspect of our proposed approach, we apply a mixture of top-down and bottom-up methods in four steps:

- i. The first step is to make an **analysis of the functionality offered** by the most well-known e-learning systems of a specific genre. Each system implements or supports a number of *features* (specific actions or services that the system offers to the prospective users, such as a posting of a message in a discussion forum). By studying the systems' functionality – a reverse engineering process – a superset of all features categorized in groups as well as an analytical table are created. As an example, when trying to construct a pattern language for Asynchronous Network Supported Collaborative Learning systems (ANSCL), we have analyzed the functionality of a number of well known ANSCL systems such as FLE3, IBM Lotus Learning Space, etc. These systems have been examined thoroughly in order to create a set of functions offered both at tutors' and learner's side'.
- ii. The second step is to **create scenarios for e-learning activities** (in the exemplar case: collaborative learning) in order to obtain the *requirements* that the e-learning

systems must meet for the chosen scenarios in concrete (most preferably authentic) learning situations. In our case, we created scenarios for CSCL activities based on collaborative learning strategies such as pyramid, jigsaw, etc. (NISE, 1997). The proposed scenarios were also based on some existing and successful ANSCL uses found in the literature. The outcomes of this step are the specific requirements about the e-learning systems and their correlation with the features already specified.

- iii. The third step is to **compare the way different e-learning systems support the chosen scenarios** and identify the pros and cons of each system. For example, in the case of the pattern language for ANSCL systems, in the previous step, we had examined the most well known ones. Now, we had to pick some specific features from different systems and study the usability of each one of them by performing the proposed scenario. The objective of this task was to examine the behavior of each system, to further elaborate the requirements for these systems, to compare the way different systems support the chosen scenarios and to identify the pros and cons of each studied system. Moreover, helpful information related to design patterns can be found such as: the problem that a pattern is aimed to solve can be extracted from the situations the actors in the scenarios have faced; the context can be seen as a generalization of the description of each scenario; some forces are picked up from the underlying behaviour of the actors; the way actors successfully used the e-learning system for performing a task can be taken as known uses. A snapshot of the most usable system was taken in order to include it as an example in the respective design pattern. Finally, some relations among patterns can be taken by observing the actual behavior of the actors for accomplishing the objectives of the scenarios.

- iv. The fourth step is to construct the pattern language for this specific genre of e-learning system, which could look like a hierarchical task analysis model (Duyne et al., 2003).

Figure 3 shows how each step in the pattern identification process contributes in its actual drafting.

[Insert Figure 3 here]

Two other steps occur in parallel:

- Reuse design expertise that has been codified as design patterns in other domains. Adopting and adapting design patterns that have been created for other genres of system can save time and energy as well as avoid us from “reinventing the wheel”. In our case we adapted some patterns for HCI design patterns repositories like the “News” design pattern (Welie, 2005).
- Brainstorming with e-learning researchers and practitioners, observing user tendencies via log files analysis, evaluation reports, etc. In this way, one can get useful information about user needs and behavior. In the TELL project, we performed a series of systematic evaluation studies of collaborative learning activities in authentic situations. For example, in one of the real CSCL scenarios studied within the TELL project, pedagogical patterns corresponding to learning activities with structured collaborative flows have been tested in a real authentic scenario. In this case, the Synergeia asynchronous collaboration tool has been used in the Faculty of Education at the University of Valladolid. Initial results concerning the process of using and adapting these collaborative learning flow patterns are reported at (Hernández et al, 2006).

Discussion

The approach of using patterns as design vehicle is employed by designers of interactive systems. This is the reason for the proliferation of HCI pattern repositories. Researchers have worked towards capturing their expertise in web design in the form of design patterns.

The hardest part of finding a good pattern is to distill out of our everyday experience that thing that all good solutions to the problem have in common. Alexander claims that:

“The task of finding, or discovering, such an invariant field is immensely hard. It is at least as hard as anything in theoretical physics.”(Alexander, 1997).

Only few articles have appeared in the literature that discuss ways of identifying a pattern. Most authors give general tips on how to start with constructing patterns (e.g. Lea 2001, Cunningham 1994, Vlissides, 1996), and evaluate them by means of a shepherding process (Harrison 2000).

The most interesting papers about the actual identification and derivation ('pattern mining') processes which are necessary to construct a pattern can be found in (Baggetun, et al. 2004, Kreimeier et al., 2003).

Moreover, many authors (Kopper, 2004, Derntl & Motschnig-Pitrik 2004) state that patterns are not solely created by human cognitive processing, but can be partly detected from patterns in the data. They propose both automated and no-automated methods of this kind of derivation. (Schümmer, 2003) has adapted some Alexandrian patterns, such the “door”, “bell”, “room” to the “Patterns 4groupware”. Cunningham (1994) and Cave (2003) propose the use of mindmaps for creating associations of patterns. In the proposed approach, the mindmap is replaced by hierarchical task analysis models which could be regarded as alike. Patterns can also be taken verbatim from an application postmortem. This is an approach followed by Kreimeier (2002) for harvesting some of the game patterns (e.g the pattern “FILTER”).

In general, the proposed approach for pattern mining incorporates aspects from all the aforementioned methods. It innovates in the fact that it suggests the use of authentic scenarios in order to identify ideas for patterns, most often interrelated, that capture deep insights into design. Till now, usage scenarios have been proposed as an aid for validating the solution embedded in the pattern (Cagnin, 2005).

The proposed approach has been formally applied in practice in four cases with very good results. The most complete one concerned the construction of a pattern language for asynchronous computer supported collaborative systems and occurred as part of the TELL project (Georgiakakis & Retalis, 2005). This language contains 36 interrelated design patterns. Most of them are new and only few are adaptations of HCI design patterns. The reason is that the team made a methodical and systematic effort to provide design solutions for effective technological tools for collaborative learning using all the sources described in the previous sections. Smaller scale application within the framework of students' theses was also made. The outcomes were pattern languages for synchronous conferencing systems (Gonzalez, 2005), learning objects repositories (Martín, 2005), and wiki systems (Martínez, 2005) that are currently in the process of further analysis and refinement. Those languages are quite rich (they contain 38, 17, 24 respectively) but the quality of the patterns needs improvement. The majority of the patterns are new and some are adaptations of existing HCI patterns. The experimentation with this approach was successful since all patterns do not stand isolated but they are interrelated according to a well justified rationale.

Closing, we believe that patterns are conceptual tools that permit communication between designers (in his domain, architects) and users (Fincher 1999). There is no silver bullet but systematic approaches should be found in order to construct high quality pattern languages.

Acknowledgements

This work has been partially funded by through the EU e-Learning project TELL (EAC/61/03/GR009). We are glad to acknowledge the work of our partners in the TELL project, particularly in the context of our shared work on evaluation methodologies.

References

Alexander, C., Ishikawa, S., Silverstein, M., Jacobson, M., Fiksdahl-King, I. and Angel, S. (1977). *A Pattern Language*. Oxford University Press, New York.

Alexander, C. (1996). *The Origins of Pattern Theory: the Future of the Theory, And The Generation of a Living World*, Keynote speech at the 1996 ACM Conference on Object-Oriented Programs, Systems, Languages and Applications (OOPSLA), retrieved from <http://www.patternlanguage.com/archive/ieee/ieeetext.htm>

Avgeriou, P., Papasalouros, A., Retalis, S., & Skordalakis, M. (2003). Towards a pattern language for learning management systems. *Educational Technology & Society*, 6 (2), 11-24.

Baggetun, R., Rusman, E., & Poggi, C. (2004). Design patterns for collaborative learning: From practice to theory and back. In Cantoni, L. & McLoughlin, C. (Eds.) *Proceedings of the World Conference on Educational Multimedia, Hypermedia and Telecommunications*, Lugano, Switzerland: 2493-2498.

Bennedsen, J., Eriksen, O. (2006). Categorizing Pedagogical Patterns by Teaching Activities and Pedagogical Values. Computer Science Education, Special themed issue: Pedagogic patterns (in press)

Cagnin, M. I., Braga, R. T. V, Germano, F. S. R., Chan, A., Maldonado, J. (2005). Extending Patterns with Testing Implementation. In: Fifth Latin American Conference on Pattern Languages of Programs, 2005, Campos do Jordão - SP. Proceedings do SugarLoafPLoP 2005, 2005. v. 1

Carlson, P. (1998). Advanced Educational Technologies – Promise and Puzzlement, Journal of Universal Computer Science (JUCS), (Special Issue), 4(3).

Cunningham, W. (1994). Tips for writing pattern languages. Retrieved April 22, 2004, from <http://c2.com/cgi/wiki?TipsForWritingPatternLanguages>

Derntl, M., Motschnig, R (2004). Patterns for Blended, Person-Centered Learning: Strategy, Concepts, Experiences, and Evaluation. Proc. SAC'04 (Symposium on Applied Computing), ACM Press, Nikosia, Cyprus (2004).

Van Duyne, D. K., Landay, J. A., & Hong, J. I. (2003). The design of sites: Patterns, principles, and processes for crafting a customer-centered Web experience. Boston: Addison-Wesley.

E-LEN (2004a). Design patterns and how to produce them, retrieved 2005a from http://www2.tisip.no/ELEN/documents/ELEN-Deliverables/booklet-e-len_design_experience.pdf.

Fincher, S. (1999). Analysis of Design: an exploration of patterns and pattern languages for pedagogy. *Journal of Computers in Mathematics and Science Teaching: Special Issue CS-ED Research*, 18(3), 331-348.

Fincher, S., & Utting, I. (2002). Pedagogical Patterns: their Place in the Genre. Paper presented at the ITiCSE 2002, Aarhus, Denmark.

HCI design patterns web site <http://www.hcipatterns.org/>, retrieved 2005.

Gamma, E., Helm, R., Johnson, R. and Vlissides, J. (1994). *Design Patterns: Elements of Reusable Object-Oriented Software*, Addison-Wesley.

Garrido, A., Rossi, G. and Schwabe, D. (1997). Pattern Systems for Hypermedia. In *Proc. of PLOP'97 - University of Illinois, Monticello, USA*.

Garzotto, F., Paolini, P., Bolchini, D. and Valenti, S. (1999). Modeling-by-Patterns of Web Applications", In *Proc. International Workshop on the World Wide Web and Conceptual Modeling, WWW CM'99, Paris, November 1999*.

Georgiakakis, P., & Retalis, S. (2005). Desmistifying the asynchronous network supported collaborative learning systems. *International Journal of Computer Applications in Technology*, Special Issue on Patterns for Collaborative Systems, (in press).

E. Gonzalez (2005). Design Patterns for Synchronous Computer Supported Collaborative Environments, Final project thesis, Telecommunications Engineering, Universidad de Valladolid, August, 2005

Goodyear, P. (2001) Learning and digital environments: lessons from European research, in O’Fathaigh, M., ed., Education and the information age: current progress and future strategies, Cork: Bradshaw Books, 1–25.

Goodyear, P., Avgeriou, P., Baggetun, R., Bartoluzzi, S., Retalis, S., Ronteltap, F., & Rusman, E. (2004). “Towards a pattern language for networked learning”, Proceedings of Networked Learning 2004, pp. 449-455.

Haberman, B (2006). Pedagogical Patterns – a Means for Communication within the CS Teaching Community of Practice. Computer Science Education, Special themed issue: Pedagogic patterns (in press)

Harrison, N., and Coplien, J. (2002). Pattern Sequences. In Andreas Rüping and Christa Schwanninger, eds., Proceedings of the 6th European Conference on Pattern Languages of Programs 2001, Konstanz, Deutschland: UVK Universitätsverlag Konstanz GmbH, 2002.

D. Hernández, E.D. Villasclaras, I.M. Jorrín, J.I. Asensio, Y. Dimitriadis, I. Ruiz, B. Rubia (2006). COLLAGE, a Collaborative Learning Design Editor Based on Patterns, IEEE LTC Educational Technology and Society, 9 (1), 2006, 58-71.

D. Hernández, J.I. Asensio, Y.A. Dimitriadis, M.L. Bote, I.M. Jorrín, E.D. Villasclaras (2005). Reusing IMS-LD-formalized best practices in collaborative learning structuring, *International Journal on Advanced Technology for Learning*, 2, 2005, 223-232

Horton, W and Horton, K. (2003). *E-Learning Tools and Technologies*. John Wiley and Sons, Inc., ISBN 0-471-44458-8, 2003

Koper, R. (2005). An Introduction to Learning Design. In: R. Koper and C. Tattersall (Eds.). *Learning Design A Handbook on Modelling and Delivering Networked Education and Training* (pp. 3-20). Berlin Heidelberg: Springer-Verlag.

Lea, D. (2001). How do I go about writing a pattern? Retrieved April 22, 2004, from <http://gee.cs.oswego.edu/dl/pd-FAQ/pd-FAQ.html#qwrite>

Lehtinen, E., Hakkarainen, K., Lipponen, L., Rahikainen, M., & Muukkonen, H. (1999). Computer supported collaborative learning: A review of research and development (The J.H.G.I. Giesbers Reports on Education No. 10). Nijmegen, the Netherlands: University of Nijmegen, Department of Educational Sciences.

Lyardet, F., Rossi, G. and Schwabe, D. (1999). Patterns for Adding Search Capabilities to Web Information Systems. In *Proc. of Europlop'99*, pp. 134-147, Kloster Irsee, Germany, IEEE Press.

G. Y. Martín (2005). Wiki tools for collaborative learning environments, Final project thesis, Telecommunications Engineering, Universidad de Valladolid, August, 2005

C. F. Martínez (2005). Design patterns for Learning Objects Repositories, Final project thesis, Telecommunications Engineering, Universidad de Valladolid, August, 2005

Mezaros, Gerard, and James Doble(1998). A Pattern Language for Pattern Writing,” in Pattern Languages of Program Design-3, Addison-Wesley, Reading MA, 1998, pages 529-574.

Moore, M. G., & Kearsley, G. (1996). Distance Education: A Systems View, Wadsworth Publishing Company.

NISE (1997). Doing CL: CL Structures, retrieved October, 2004 from <http://www.wcer.wisc.edu/nise/cl1/CL/doingcl/clstruc.htm>”,

Pedagogical Patterns Project site, Retrieved 2005 <http://www.pedagogicalpatterns.org/>.

Roschelle, J. Kaput, J., Stroup, W., and Kahn, T. M.(1998). Scalable Integration of Educational Software: Exploring the Promise of Component Architectures. Journal of Interactive Media in Education (6) (Oct, 1998) pp. 1-31

Schümmer, T. (2003). GAMA - A Pattern Language for Computer Supported Dynamic Collaboration, , Proceedings of the 8th European Conference on Pattern Languages of Programs, EuroPLoP 2004, UVK, Konstanz, Germany, 2004.

TELL (2005a). "Introducing a Framework for the Evaluation of Network Supported Collaborative Learning", (1.37 MB .pdf) TELL Project, Deliverable of WorkPackage 1. Retrieved 2005 from TELL project website from <http://cosy.ted.unipi.gr/tell/>.

TELL (2005b). " Design patterns for teachers and educational (system) designers: Pattern book", TELL Project, Deliverable of WorkPackage 3. Retrieved 2005 from TELL project website from <http://cosy.ted.unipi.gr/tell/>.

Vlissides, J. (1996).Pattern Hatching: Seven Habits of Successful Pattern Writers. C++ Report, November/December 1996. <http://st-www.cs.uiuc.edu/users/patterns/papers/7habits.html>

Van Welie, M. and Van der Veer, G. C. (2003). Pattern Languages in Interaction Design: Structure and Organization. Interact 2003.

Van Wilie, M (2005). The News Box. Retrieved 2005 from "The collection of Web Design patterns", <http://www.welie.com/patterns/>

Captions to illustrations

Figure 1. The ANNOTATION_ON_POSTED_MESSAGES pattern

Figure 2. The process for forming a design pattern language for e-learning systems

Figure 3. The pattern identification process in steps

AnnotationOnPostedMessage

Problem

How can the user decide if a message is interesting, urgent, or just junk?

Does the user have to read all the messages that arrive in the inbox?



Forces

- The user needs to know characteristics and/or modifications of each message, without spending time opening the messages and reviewing their contents
- Users need reminders for actions taken on past messages (especially those opened long time ago)
- Users need information on the usefulness of the contents of each message
- Users need to get (notification) about every alteration in the material or the contents of the postings
- Users need comprehensive annotations so that they don't spend more time trying to understand what each icon means than the time needed to read the message
- Users must be notified whether community members have posted remarks or questions on a specific topic

Solution

Let the users know of actions taken in previous sessions (either by themselves or by other participants).

Each message must be annotated regarding its importance (e.g. urgent, innovative) in order for community members to be able to choose to access those that are closer to their interests. Also in the cases when posted messages have been modified from their initial version, new data are contained, they have already been read, remarks have been made from other members of the community or notes have been placed on their contents, annotation should exist, in order to avoid spending time reading or trying to remember what were the previous interactions with each posting.

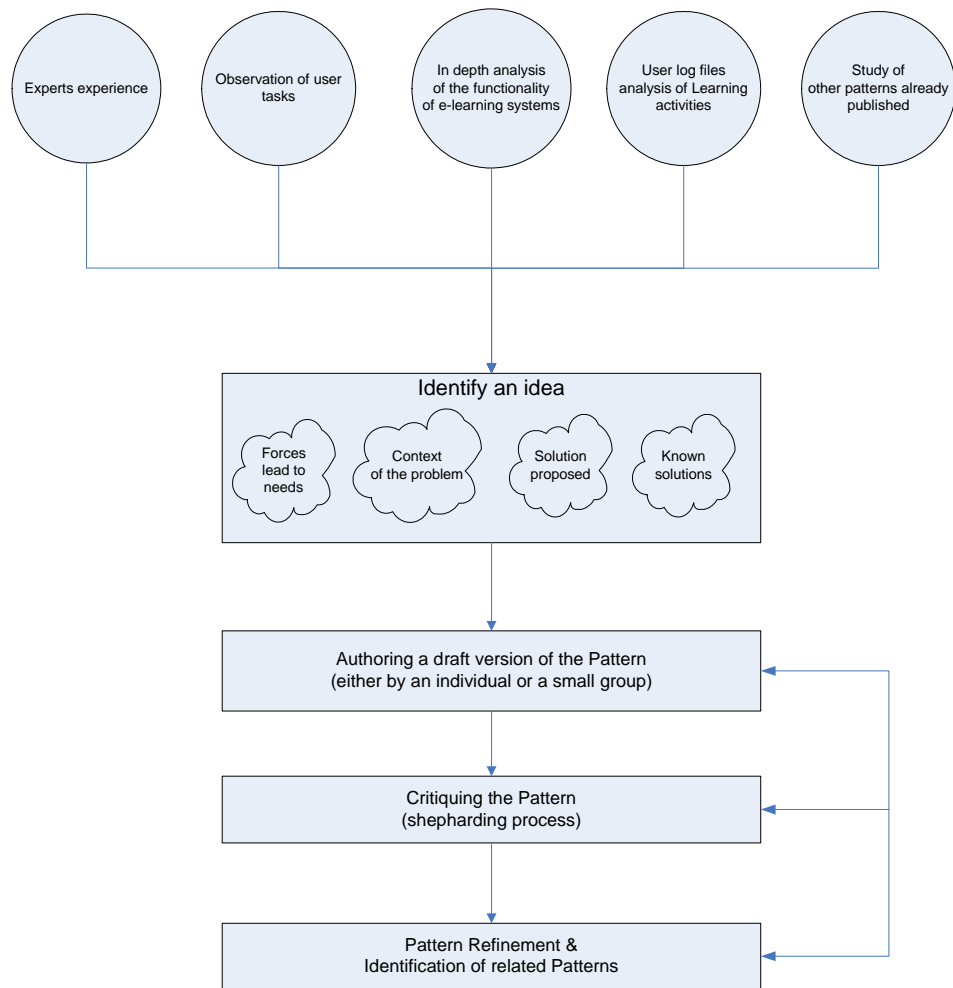
Example(s)

patras_literature_papers	6	jamarco	2004-12-13	Annotation
piraeus_literature_papers	7	s	2004-09-15	Annotation
Templates of papers reviewed by partners	67		2005-01-11 12:40	Annotation
uva_literature_papers	4		2004-06-07	Annotation
Comments about Patras_literature_papers	32.5 K	jamarco	2004-12-13	Annotation
Comments about APriori_literature_papers	29.5 K	jamarco	2004-06-29	Annotation
Comments about Athens_literature_papers.do	...	nickle	2004-07-08	Annotation
Comments about mansticht_literature_papers	...	yannis	2004-12-04	Annotation
Comments about milano_literature_papers.do	...	Caterina	2004-09-21	Annotation
Comments about phraeus_literature_papers	...	retal	2004-06-03	Annotation
Comments about uva_literature_papers	32.0 K	jamarco	2004-06-04	Annotation
Global list of external documents to review together with assignment to partners	1.35 K	yannis	2004-12-17	Annotation

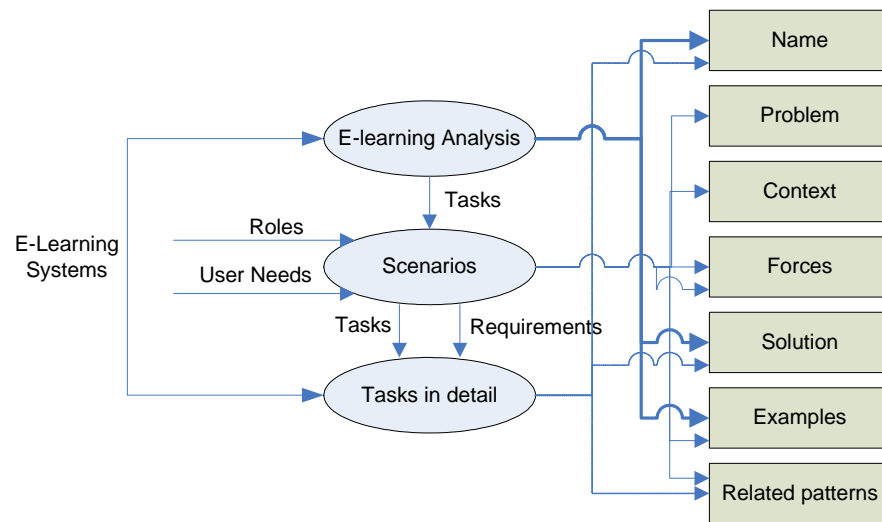
BSCW annotation: System categorizes postings as: read, unread, contain new data, contain inverts inside, changed. Even if the system provides for each annotation meta-data, the annotation itself is not very descriptive. We suggest the use of more comprehensible annotations, even a use of a "pool" of acceptable annotations, which each user is familiar with, and meta-data need not be used as much for time saving. There should also be a pop-up window, as shown below, with the summary of the annotations on the messages.

Summary of annotation	
POSTED: J.J.	
AUTHOR: ...	
QUESTIONS POSTED: NUMBER & BY WHOM (LINKS)	
REPLIES: NUMBER & TO WHOM (LINKS)	
READ: YES/NO	
DATE: J.J.	
FORWARDED TO: ...	
NEW DATA ADDED: YES/NO	
BY: ...	
EVENTS UNTIL NOW: ...	
NEW EVENTS: ...	
CHANGED: J.J.	
BY: ...	
RATING: "SCALE"	
PEOPLE READ IT: NUMBER OR %	
READERS FROM COMMUNITY: NUMBER, OR NAMES OR %	

[Figure 1]



[Figure 2]



[Figure 3]