UPGRADE
The European Journal for the Informatics Professional
http://www.upgrade-cepis.org
Vol. IV, No. 5, October 2003

UPGRADE is the European Journal for the Informatics Professional, published bimonthly at <http://www.upgrade-cepis.org/>

Publisher
UPGRADE is published on behalf of CEPIS (Council of European Professional Informatics Societies, <http://www.cepis.org/>) by NOVÁTICA <http://www.ati.es/novatica/>, journal of the Spanish CEPIS society ATI (Asociación de Técnicos de Informática <http://www.ati.es/>).
UPGRADE is also published in Spanish (full issue printed, some articles online) by NOVÁTICA, and in Italian (abstracts and some articles online) by the Italian CEPIS society ALSI <http://www.alsi.it> and the Italian IT portal Tecnoteca <http://www.tecnoteca.it/>.
UPGRADE was created in October 2000 by CEPIS and was first published by NOVÁTICA and INFORMATIK/INFORMATIQUE, bimonthly journal of SVI/FSI (Swiss Federation of Professional Informatics Societies, <http://www.svifsi.ch/>).

E-mail addresses for editorial correspondence: <rfcalvo@ati.es>, <nicolet@acm.org> or <rcarniel@dgt.uniud.it>

E-mail address for advertising correspondence: <novatica@ati.es>

Upgrade Newslist available at <http://www.upgrade-cepis.org/pages/editinfo.html#newslist>

ISSN 1684-5285

Next issue (December 2003):
"IS Security and Contingency Plans"

e-Learning – Borderless Education

Guest Editors: **Ángel Fidalgo-Blanco and Martín Llamas-Nistal**

Joint issue with NOVÁTICA*

* This monograph will be also published in Spanish (full issue printed; summary, abstracts and some articles online) by NOVÁTICA, journal of the Spanish CEPIS society ATI (Asociación de Técnicos de Informática) at <http://www.ati.es/novatica/>, and in Italian (online edition only, containing summary abstracts and some articles) by the Italian CEPIS society ALSI and the Italian IT portal Tecnoteca at <http://www.tecnoteca.it>.

Component-Based Software Engineering and CSCL in the Field of e-Learning

Yannis A. Dimitriadis, Juan-Ignacio Asensio-Pérez, Alejandra Martínez-Monés, and César A. Osuna-Gómez

The use of Information and Communication Technologies in the education domain has been characterized by the need of providing flexible systems that are adaptable to particular learning situations. In this sense, Component-Based Software Engineering (CBSE) has emerged as a software development paradigm suitable for obtaining reusable, flexible, and customizable distributed applications, which would provide great benefits to the e-Learning domain. Nevertheless, this CBSE-education relationship has not coped with the collaborative aspects and the pedagogic theories underlying the social constructivism that constitutes the basis for Computer-Supported Collaborative Learning (CSCL). This article describes the process undertaken by the authors when applying CBSD principles to the CSCL domain, emphasising the lessons learned during this experience. The article is particularly focused on the problem of 'translating' the highly dynamic requirements posed by educators, as well as by participants and their educational context, into the CSCL applications.

Keywords: computer-supported collaborative learning, CSCL, component-based software engineering, component framework, e-Learning.

Yannis A. Dimitriadis is Associate Professor at the School of Telecommunications Engineering of the Universidad de Valladolid, Spain. He has a BSc from the National Technical University of Athens, Greece (1981), an MSc from the University of Virginia, USA (1983), and a PhD from the Universidad de Valladolid, Spain (1992, 1995), all of them in Telecommunications Engineering. His research interests include the technological support to learning and work processes, as well as machine learning. He has published more than 100 book chapters, journal articles, as well as papers in Spanish and international conferences and workshops. He is reviewer in several journals and member of the editorial board of the Applied Intelligence journal, Kluwer Academic. <yannis@tel.uva.es>

Juan-Ignacio Asensio-Pérez is Associate Professor at the School of Telecommunications Engineering of the Universidad de Valladolid, Spain, where he got his MSc (1995) and PhD (2000) degrees in Telecommunications Engineering. His research interests include the integrated management of network and systems, as well as component-based software engineering and distributed processing technologies applied to the educational field. <juaase@tel.uva.es>

Alejandra Martínez-Monés is Assistant Professor at the Department of Computer Science of the Universidad de Valladolid, Spain. She got her MSc (1997) and PhD (2003) degrees in Computer Science by the same University. Her research activity is focused on the CSCL field, mainly on the technological support to the evaluation in CSCL environments. She has published several journal articles and papers in Spanish and international conferences. <amartine@infor.uva.es>

César A. Osuna-Gómez is researcher at the Computing and Applied Mathematics Program of the Mexican Petroleum Institute, Mexico. He has a MSc in Computer Science from the Universidad Autónoma de Guadalajara, Mexico, and a PhD (1999) from the University of Valladolid, Spain. His research interests include the conceptual formalization of the CSCL field and its relation with technologies from the distributed systems world. <cosuna@imp.mx>

1 Introduction

Attempts to apply technological innovation in order to enhance education have always been present. In the case of ICT (Information and Communication Technologies) these efforts have resulted in several paradigms of educational computer-based systems: Computer Assisted Instruction (CAI), Intelligent Tutoring systems (ITS), simulations or microworlds, and, more recently, due to the generalised use of computer networks, e-Learning and CSCL (Computer-Supported Collaborative Learning) [1].

This diversity reflects, often implicitly, the evolution in technologies as well as in educational trends. For instance, the term e-Learning, that represents the current dominant paradigm, has incorporated web technologies and has put more emphasis on a student-based, autonomous, and flexible learning (typically in distance education).

1.1 CSCL: A New Paradigm of Educational Software

CSCL, partially derived from an evolution of Computer-Supported Cooperative Work (CSCW), is based on a new and strongly interdisciplinary paradigm of research and educational practice [2]. Its main features include highlighting the importance of social interactions (collaboration) as an essential element of learning, the preference for an interpretative approach to the evaluation of the learning process (as opposed to traditional positivist proposals), as well as the role of participative analysis and design of the whole community when creating new technological environments [3]. On the other hand, CSCL has been based on distributed systems technologies in order to support some of its main characteristics, i.e.: communication, collaboration, and coordination.

Several of the aforementioned elements have been embedded in the main commercial products or innovative proposals of e-Learning, although in a marginal way. For example, generic tools that promote collaboration have been introduced (without a precise objective and environment), the importance of designing activities and associated workflows has been assumed by IMS-LD (IMS Learning Design, <http://www.imsproject.org>), or the main roles and collaborative activities have been modelled through standards such as IEEE LTSC (Institute of Electrical & Electronics Engineers – Learning Technology Standards Committee, <http://ltsc.ieee.org>). However, the main stream within e-Learning is still centred around the concepts of knowledge transmission as the basic educational paradigm, and the new proposals are dominated by the immediate application of the new technological 'affordances' and the expected market benefits. Therefore, it is still necessary to advance in order to analyse and embed all pedagogical and technological elements that define CSCL.

1.2 CSBD and Education: The Necessity of Reuse and Adaptation

In spite of the aforementioned differences, educational software in general has been traditionally exposed to the necessity of adaptation and personalisation. Such requirements have been expressed by educators who need to use the software in different educational and social contexts, or even, with different pedagogical styles. Thus, too many specific applications have been developed in order to meet the above requirements. Due to the fact that these applications are usually monolithic, dependant on particular technologies and incompatible among them, teachers usually face great difficulties in order to integrate them in the classroom [4]. These projects present a high failure rate since they are not able to get adapted to new educational situations and to incorporate technological innovations that are continuously emerging.

Software component technology [5] offers the promise of composing tools from elements that may come from different providers. Therefore, it is a reasonable candidate as a potential solution of the aforementioned problems of the educational domain, since it provides the capacity of application reuse and adaptation. When dealing with the problem of reuse in Software Engineering and particularly in CBSD (Component-Based Software Engineering), it is essential to take into account the concept of component framework [6]: an extensible set of reusable software components in a particular application domain together with a number of software design patterns that document their use. Components included in a framework can be reused, instantiated and assembled with additional components provided by developers in order to obtain concrete applications faster and with a lower cost.

CBSD has been employed in several projects in which the idea of component framework has been successfully applied in developing educational applications [4]. However, the issue of supporting collaboration, inherent to the particular CSCL domain, has not been taken into account.

1.3 Objectives and Structure

It is precisely the above objective of obtaining a software component framework for CSCL that guided the work of the authors during the last years, within a multidisciplinary group formed by educators, as well as telecommunications and computer engineers.

Nevertheless, building a component framework is not an easy task. A framework developer must face different problems related to both the particularities of the framework domain and the technologies used to support the derived components [6]. One of the most important problems to take into account in this context is the identification and dimensioning (i.e. level of granularity) of components. The fulfilment of this task largely depends on how the key concepts and principles of the domain of interest are understood by software developers [7]. In the CSCL domain, this problem is particularly important due to the big separation among abstractions used by experts in collaborative learning (pedagogues, psychologists, education practitioners,…) and those used by software developers.

The authors have employed three different approaches, partly complementary, in order to deal with the aforementioned problem of domain understanding by the software developers. The objective of this paper if to present and discuss the following three approaches in a unified way: first, a top-down approach that led to the telematic-educational conceptual framework DELFOS; second, a bottom-up approach focused on the development of concrete component-based collaborative learning applications; and, finally, as a compromise between the previous ones, an intermediate approach based on the definition of the so-called Collaborative Learning Patterns (CLP), detailed descriptions of collaborative learning techniques defined by experts in this field that could help software developers in the task of identifying components to be employed in various types of CSCL applications.

The work presented in this paper covers several points of interest: the study of the CBSD's potentialities and limitations through its application in an interesting and complex domain such as CSCL; the creation of a sufficiently large set of components and patterns, i.e. a framework for CSCL; the search for solutions to the problem of lack of shared knowledge among technologists and educators. Additionally, these proposals are validated by experience derived from real scenarios of design, development and evaluation of significant applications.

Accordingly, Section 2 presents the main features, problems and limitations of each approach, and discusses their relative merits. The principal conclusions obtained in this work are shown in Section 3, together with the most interesting research lines currently being dealt with.

2 Approaches and Results

Figure 1 shows an overview of the three approaches undertaken by the authors in order to achieve the goal of bringing together CBSD and CSCL. Each of the approaches is presented in a separate subsection, while the last one (2.4) presents a synthesis of the results obtained, based on the authors' experience in real CSCL scenarios.

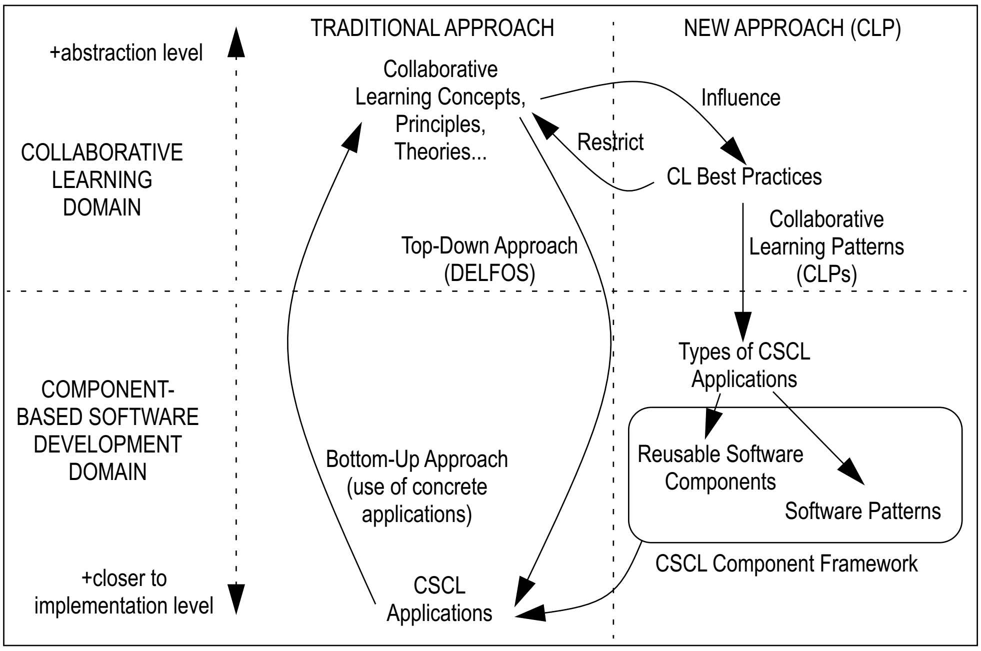

Figure 1: Three Approaches for Communicating the CBSD and Collaborative Learning Domains.

2.1 Top-down Approach: The Telematic Educational Framework DELFOS

In the first case, authors used a top-down approach, in which they aimed to solving the aforementioned problems through a general understanding of the CSCL domain, and a subsequent detection of useful elements for analysis and development of such applications. The outcome of this approach was the DELFOS (Description of Educational Layer Framework Oriented to learning Situations) framework [8], composed of the following three main elements: a *learning model* based on the ideas of social constructivism, a *hybrid architecture* composed of layers and objects for application development, and a *development methodology* based on participative analysis and design. DELFOS proposed the concept of *learning situation*, that models a specific learning environment and includes fundamental aspects for an educator, such as the context the and pedagogical objectives. A learning situation as well as its constituent elements (activities, roles, objects and interactions) are modelled with the help of a set of templates that were defined in the framework proposal.

The framework was validated via the development of three CSCL applications of different characteristics. These experiences have shown that DELFOS provides useful tools for the requirement analysis phase, thus allowing to structure the domain, and therefore simplify the communication between final users and software developers. However, it was evident that the idea of learning situations, essential in defining CSCL applications within DELFOS, is hardly reusable, due to the fact that it is highly dependant on the specific context. In other words, a learning situation defined through the DELFOS tools could not be applied without significant changes to different educational settings. Therefore, authors came to the conclusion that it was necessary to identify context-independent aspects, that could be adapted to other educational scenarios without major changes. This new action line was carried out within the third approach, that is described in section 2.3.

2.2 Bottom-up Approach: Specific CSCL Applications

The bottom-up approach deals with the development of specific component-based CSCL applications. This development process tries to extract relevant elements from the CSCL framework. The authors, therefore, worked towards the selection of a CSCL application type whose complexity level was low enough so as to enable fast prototyping, and high enough in order to share most of the common problems of the domain thus leading to the identification and development of potentially reusable software component. The solution that was adopted (called *MagicPuzzle*) consisted of the development of a synchronous and collaborative application supporting the resolution of puzzles by pupils in primary education. This type of applications stands out due to its well-known educative and socialising benefits, as well as to its ability to reflect the knowledge-building process. Within the context of this type of applications the participants share pieces of knowledge needed for the resolution of a more complex problem (probably with no closed and single solution), justifying at the same time their actions. All this process can be understood as a metaphor that contains educational methods such as case-based and project-based learning. The functional requirements of *MagicPuzzle* were decided with the help of experts in primary education.

The EJB (Enterprise Java Beans) technology, together with its supporting J2EE (Java 2 Enterprise Edition) platform, was selected for the development of *MagicPuzzle* due to its distributed nature, its higher level of maturity, its computing platform

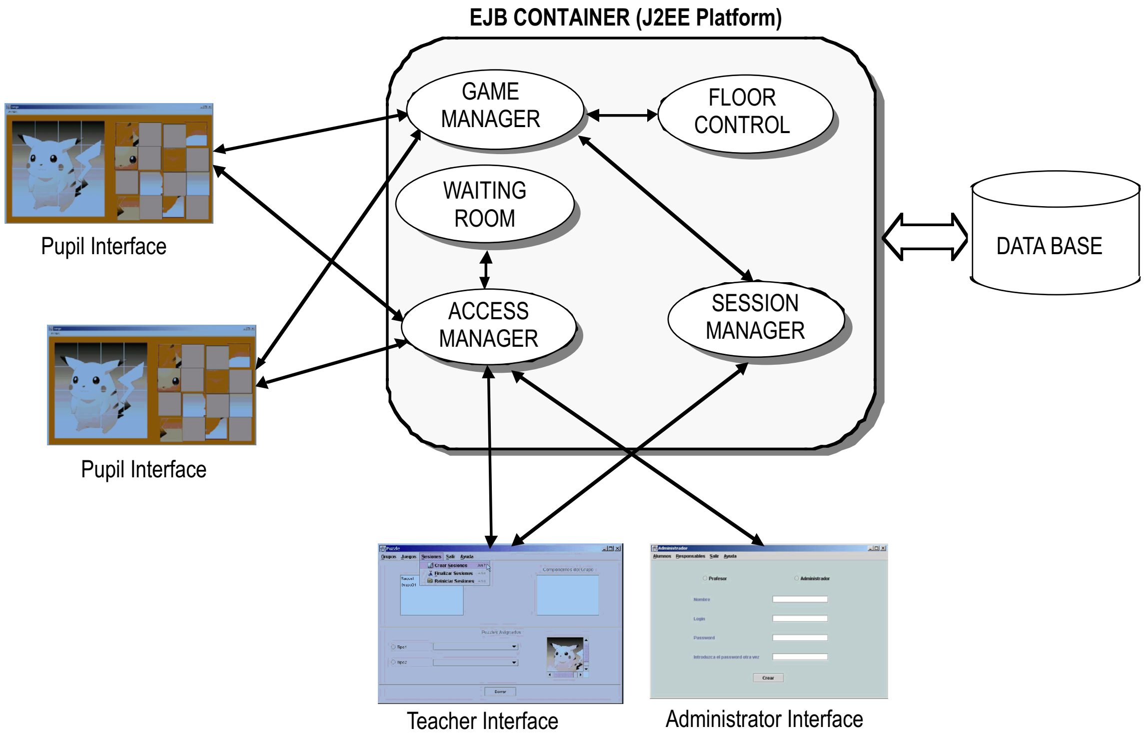

Figure 2: Software Architecture and Components of a CSCL Application that Spports the Collaborative Resolution of Puzzles by Primary Education Pupils.

independence, and the wide availability of implementations. During the *MagicPuzzle* development, several design patterns were applied as, for instance, the so-called TOP patterns (Ten Object Patterns) [9], focused on collaborative applications, and the MVC (Model-View-Controller) pattern that enables independence between data and presentation.

Figure 2 shows a schema of the architecture of *MagicPuzzle* as well as the software components it is composed of.

Once the prototypes of *MagicPuzzle* were finished, the authors explored whether the obtained components could be reused as building blocks for other CSCL applications. For solving this question, two new CSCL applications were developed: a collaborative application (called eLAO) supporting the course "Computer Architecture" in Telecommunications Engineering studies, in which the authors expected to reuse basic collaboration support components (Access Manager, Session Manager…), and a collaborative application supporting the well-known *Tangram* game, much more similar to *MagicPuzzle* and, therefore, more suitable for reusing of its components.

Nevertheless, during the first development stages it became clear that the expected reuse could not be achieved. The reasons for this setback pointed to a development process too biased towards a very particular learning problem: the collaborative resolution of puzzles. This problem, already known in the CBSD field, as well as the lack of a good formalisation of CSCL concepts, obstructs the path towards the identification and dimensioning of reusable components.

The search for common characteristics applicable to a broad set of CSCL applications in different learning contexts led the authors to the proposal described in the next subsection, based on the use of "Collaborative Learning Patterns".

2.3 Intermediate Approach: Collaborative Learning Patterns

A CLP can be understood as a way of describing a collaborative learning technique, easily understandable by software developers. Collaborative learning techniques dictate common ways of structuring interactions among participants in different collaborative learning activities, as well as the information they interchange and the objects they manipulate. The concept of CLP is derived from the notion of "Collaborative Design Pattern" introduced in [10].

CLPs are identified and formalised by Collaborative Learning practitioners (mainly teachers), and validated by pedagogy experts. They are intended to be used by software developers, in order to derive common requirements for CSCL applications that support collaborative learning activities based on the same technique. These common requirements are potentially more useful for helping developers to identify and dimension reusable software components. In spite of this final use of the CLPs, it is important to point out that the contents of the CLPs themselves do not include any technical information.

From the point of view of conceptual richness, CLPs can be considered as an intermediate approach between the two ones that were already described in the previous sections: they are not based on concrete applications but they do not try to take into account all the concepts and principles of the collaborative learning domain (as DELFOS did.)

Field	Explanation	Example
Name	Name of the CLP	Jigsaw
Problem	Learning problem to be solved by the CLP	Complex problem whose resolution implies the handling and/or collection of information that can be easily divided into disjoint sets and that can be used for the resolution of independent subproblems
Example	A real-world learning activity suitable of being structured according to the CLP	Collaborative design of a computing system where the study of each subsystem is assigned to a particular participant
Context	Environment type in which the CLP could be applied	Several small groups facing the study of a lot of information for the resolution of the same problem
Solution	Description of the proposal by the CLP for solving the problem	Each participant in a group (jigsaw group) studies a particular subproblem. The participants of different groups that study the same problem meet in an "Expert Group" for exchanging ideas. At last, jigsaw group participants meet to solve the whole problem. Each participant contributes with its "expertise"
Actors	Actors involved in the Collaborative Learning activity described by the CLP	• Teacher • Pupil • Evaluator
Types of Tasks	Types of tasks, together with their sequence, performed by the actors involved in the activity. (NOTE: due to space restrictions only types of tasks performed by pupil and teacher are shown)	*Pupil:* 1. Access to the information related to the subproblem 2. Individual study of the subproblem 3. Subproblem discussion in the experts group 4. Problem resolution in the jigsaw group 5. Result proposition 6. Process self-evaluation *Teacher:* 1. Global problem definition 2. Division of the problem into subproblems 3. Creation of jigsaw groups 4. Assignment of subproblems 5. Provision of useful information 6. Floor control system establishment 7. Decisions about control of time 8. Activity progress monitoring 9. Result evaluation
Types and structure of Information	Description of the types of information identified in the collaborative activity and how they are related	• Input information needed for global problem resolution • Partial information assigned to subproblems • Subproblem resolution proposal • Global problem resolution proposal • Correct global problem resolution (optional)
Types and structure of Groups	Description of the types of groups of pupils identified in the collaborative activity and how they are related	• Jigsaw groups • Experts groups in charge of subproblems

Table 1: Collaborative Learning Pattern Structure and Its Application to the Jigsaw Technique.

CLPs are represented according to the formalism that is shown in Table 1. It also shows a CLP defining a very well-known practice in Collaborative Learning: *jigsaw* [11]. Other CLP's (pyramid, simulation, etc…) have also been defined and applied by the authors.

The use of CLPs depends on the particular software development methodology that is employed. As a way of illustrating these ideas, if a software development methodology based on the widely accepted Unified Process (UP) [12] is chosen, the information provided by CLPs might be used as the basis for the derivation of actors and use cases, the conceptual model (also known as domain model), and the analysis of the use cases during the iterations of the so-called 'Inception Phase'.

Figure 3 shows UML (Unified Modelling Language) use case and class diagrams representing use cases and conceptual modelling for a software application that could eventually support a collaborative learning activity of the type described by the jigsaw CLP defined in Table 1. As it can be seen, the use case diagram focuses on the identification of the functionality needed for supporting the tasks performed by the different actors involved in the CLP. On the other hand, the conceptual or domain model reflects the types and the structure of the information and groups described by the CLP, as well as the interrelation among them. It can be seen, for instance, how *Jigsaw Group*, and *Expert Group* classes are associated to *Global Problem* and *Subproblem* classes which, at the same time, maintain a relationship of aggregation between them.

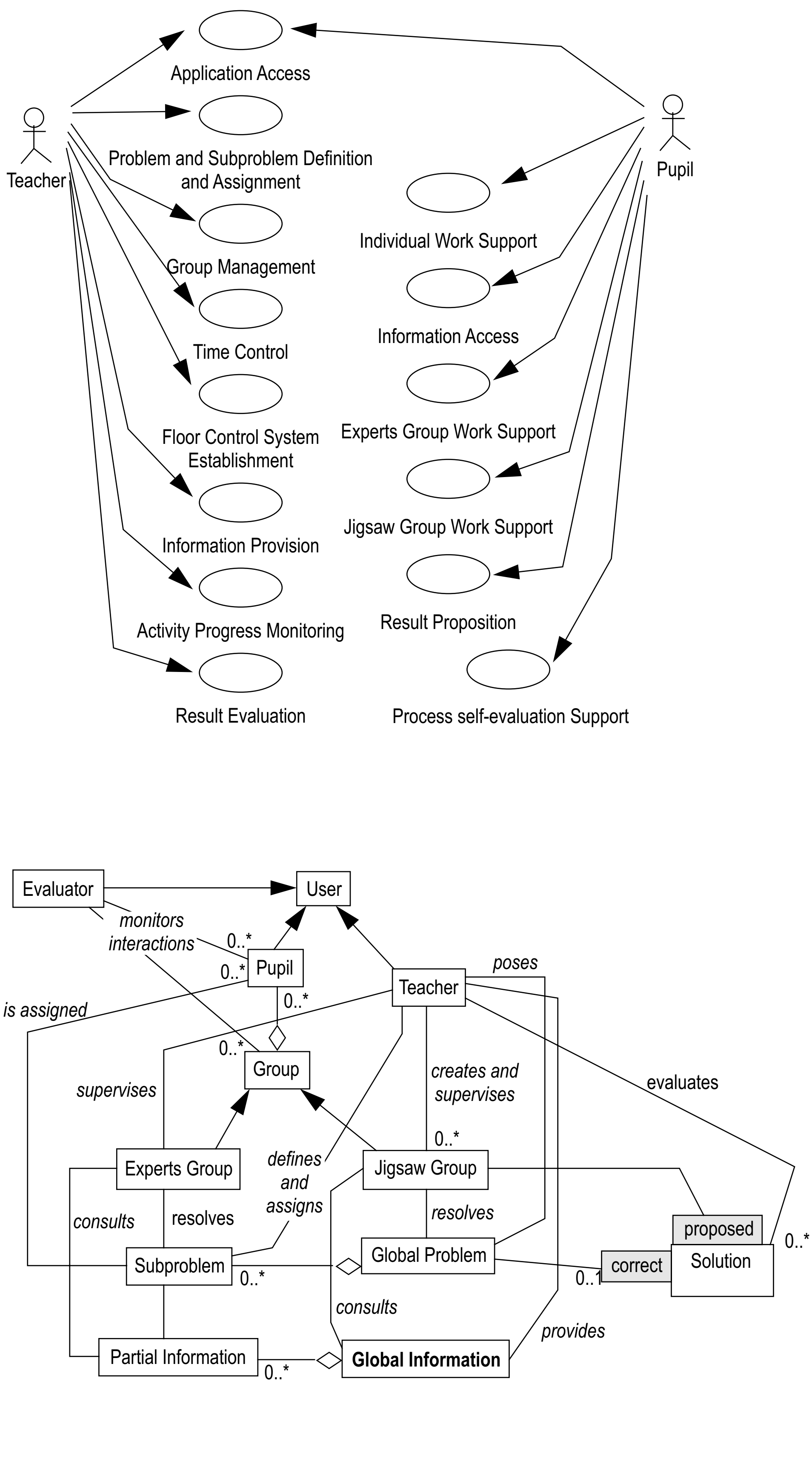

Figure 3: UML Use Case Diagram and Conceptual Class Diagram Derived from the Jigsaw CLP

After completing the UP inception phase using the information provided by CLPs, and using normal software development techniques prescribed (by UP in this example), it is possible to obtain a software design architecture for a jigsaw CSCL application. Obviously, the authors recognize that it is very difficult to prescribe a unique way for achieving a specific software design when starting from the definition of a CLP. CLPs provide clues and help for software development but they do not intend to dictate a complete software development methodology.

In terms of software reusability, the consequences of this approach are very important: CLPs help developers to understand concepts and requirements involved in the support of the different CSCL techniques. This makes easier the identification of common software components for CSCL applications based on the same techniques. These common components are, potentially, more reusable that those obtained from the development of a particular CSCL application.

CLP have been successfully applied by the authors to the development of the eLAO application, already mentioned in Section 2.2. Reuse of eLAO software components is under evaluation by developing new CSCL applications based on the same CLPs. Our first findings show that, for instance, those components that support tasks in charge of teacher role, as well as those related to information handling, are, potentially, the most reusable ones. Therefore, these components are the most suitable for becoming part of the pursued CSCL component framework.

2.4 Synthesis of Results and Discussion

Authors' experience, as described in the previous sections, indicates that the CBSD technology does not guarantee by itself the achievement of software reuse, flexibility and

	Top-Down (DELFOS)	Bottom-Up (Concrete CSCL Applications)	Intermediate (CLP)
Conceptual Base	High	Low	Medium
Developers comprehension	Low	High	Medium
Reuse/ Results generality	Low	Low	High
Capability for describing concrete situations	High	Low	Medium

Table 2: Comparison of the Three Approaches Described in This Article. (It can be seen how the CLP-based approach enables the reuse of the obtained components but requires an additional effort so as to be applied to specific scenarios.)

customisation. Domain-specific knowledge (collaborative learning in this case) is, in this sense, an unavoidable requisite. This fact leads to the conclusion that the availability of mechanisms for enabling communication between domain experts and software developers is more than desirable.

In the particular case of CSCL applications, the authors have noticed that neither pure top-down nor bottom-up approaches facilitate the identification of reusable software components: they focus on very specific learning situations, which are hard to generalize. Nevertheless, at least in the DELFOS-based approach, learning situations have a very strong conceptual base from the point of view of social constructivism (although this derives in a difficult comprehension by software developers, which is much easier in the bottom-up approach).

Within this context, CLPs arise as the best alternative for identifying and developing reusable CSCL components, as they do not focus their attention on specific learning situations but on common techniques to some of them. CLPs start from a pragmatic vision since they are defined by practitioners, not by pedagogues. However, the latter should validate them so as to make sure that the defined CLPs conform to the principles of social constructivism. Accordingly, CLPs can be considered as an intermediate solution in terms of intelligibility by software developers.

On the other hand, CLPs have a drawback: when generating specific CSCL applications, developers must make an additional effort for customizing and completing the components identified by means of CLP's (see Table 1). In these cases, the CLP-based approach can benefit from the capability of DELFOS for dealing with the descriptions of specific collaborative learning situations.

3 Conclusions and Future Work

CSCL is a rather new paradigm within educational software that takes into account pedagogic aspects (that should not be excluded from any approach to educational software development). In this article several approaches undertaken by the authors in order to identify and dimension software components for a CSCL framework have been reviewed. This framework aims to facilitating the development of reusable and customisable collaborative learning applications. This work has shown that the CLP-based approach is the best one in terms of reuse. Nevertheless, this fact does not imply the exclusion of the other two approaches.

In order to progress towards the achievement of a CSCL component framework, several research lines have to be explored. For instance: deriving the CSCL framework as an extension to existing CSCW component frameworks; coordinating collaborative learning activities, based on CSCL components, by applying workflow management principles to standards for the description of learning situations (as IMS-LD); and developing tools for enabling teachers and pedagogues to generate new and customized CSCL applications built from components belonging to the CSCL component framework.

Acknowledgements

The authors want to acknowledge the contributions from the rest of members of the EMIC (Education, Media, Information, and Culture) Research Group of the Universidad de Valladolid, Spain, specially Teresa Blasco, Rocío Anguita, Lino Barrio, Bartolomé Rubia, Ivan Jorrín (Faculty of Education); Eduardo Gómez, Miguel Bote, Guillermo Vega, Francisco J. Álvarez, Marta Heredia, Davinia Hernández, Sonia Díez (School of Telecommunications Engineering); Pablo de la Fuente (School of Computer Science); Pablo Orozco (UNAM, Mexico). Special thanks to the educators and students that make the evaluation of these technologies possible.

This work was partially financed by the Regional Government of Castilla and León, Spain (project VA117/01), and the Spanish Ministry of Science and Technology (projects TIC2000-1054 and TIC-2002-04258-C3-02).

References

[1] J. Arlow and I. Neustadt. UML and the Unified Process: Practical Object-Oriented Analysis and Design, Addison Wesley Professional, 2001.

[2] J. Carey and B. Carlson. Lessons learned becoming a framework developer Software Practice and Experience, vol. 43, pp. 789–800, 2002.

[3] C. DiGiano, L., Yarnall, C. Patton, J. Roschelle D. Tatar, and M. Manley. "Collaboration design patterns: conceptual tools for planning for the wireless classroom," Proceedings of the IEEE International Workshop on Wireless and Mobile Technologies in Education (WMTE'02), 2002.

[4] L. A. Guerrero D. A. and Fuller. A pattern system for the development of collaborative applications Information and Software Technology, vol. 43, pp. 457–467, 2001.

[5]
D. W. Johnson and R. T. Johnson. Learning together and alone: cooperative, competitive and individualistic learning, Allyn and Bacon, 1999.

[6]
T. Koschmann. Paradigm shift and instructional technology. In: CSCL: Theory and Practice of an emerging paradigm, ed. Koshmann, T. Lawrence Erlbaum, 1996.pp. 1–23.

[7]
A. Martínez. Method and Model for the computational support to evaluation in CSCL (in spanish), Tesis Doctoral. Universidad de Valladolid, 2003.

[8]
H. Mili, M. Fayad, D. Brugali, D. Hamu, and D. Dori. Enterprise frameworks: issues and research directions Software Practice and Experience, vol. 32, pp. 801–831, 2002.

[9]
C. Osuna Y. and Dimitriadis. “A framework for the development of educational collaborative applications based on social constructivism,” Proceedings of the CYTED RITOS International Workshop on Groupware (CRIWG'99), 1999.

[10]
J. Roschelle, J. Kaput, W. Stroup, and T. M. Kahn. Scalable integration of educational software: exploring the promise of component architectures Journal of Interactive Media in Education, vol. 98, Oct. 1998.

[11]
C. Szypersky. Component software. Beyond object-oriented programming, NY, USA: Addison Wesley, 1998.

[12]
A. Vaquero. Las TIC para la enseñanza, formación y el aprendizaje, Novática, vol. 132, pp. 4–14, Mar, 1998.