Enabling Project-Centred Learning through Flexible Processes: the COOPER Experience

Stefano Ceri*, Florian Daniel*, Maristella Matera*, Alessandro Raffio*, Howard Spoelstra[+]

* DEI – Politecnico di Milano
P.zza L. da Vinci, 32 - 20133 – Milano - Italy
+39-02-23993408
[ceri, matera, raffio]@elet.polimi.it

[+] OUNL – Open University of the Netherlands
Valkenburgerweg 177 - 6419 AT Heerlen – The Netherlands
+33-45-5762391
howard.spoelstra@ou.nl

Abstract. This paper proposes a model-driven, extensible platform, delivered on the Web, which is able to support long-distance collaboration of students' teams working on complex projects. The main merit of this proposal is the ability to support end-users to self-organize processes, by using a simple Web interface and a library of activities that cover most of the needs arising in collaborative environments. In this way, students can organize processes in a flexible way, and at the same time their work is well-organized, well-understood by all team members. This underlying paradigm for the management of dynamic processes is very general and can be applied to other application contexts, different from e-learning, after understanding and modelling the relevant collaboration activities.

Keywords: Flexible Processes, E-learning, Web Application Design.

Introduction

Computer Supported Collaborative Learning (CSCL) provides learning environments where learners' teams collaborate by means of computer-mediated services, with the aim of reaching a common learning goal [4]. In this context, collaboration implies some form of coordination [3][7] enabling the definition of processes guiding the learners' activities.

Process-oriented collaboration in e-learning systems is an important challenge today. Collaboration processes should indeed be *flexible* enough to be adapted to the preferences of learners, as well as to the learners' evolution of background knowledge and competencies. So far, very few approaches have been proposed for the management of flexible e-learning processes that can be specified by end-users at runtime, and can also be modified during their execution [3][7][9].

In the context of the COOPER EU project [1], we are currently addressing the problem of supporting the definition of *flexible processes*, to enable long-distance collaboration of teams working on complex projects. This paper illustrates a reference model for teamwork flexible collaborative processes, and describes a platform enabling end-users to dynamically (i.e., at runtime) define and modify their processes according their collaboration needs.

The paper is organized as follows: after illustrating the motivation of our research, we illustrate the main ingredients of our approach, namely a library of collaboration activities and a Web-based interface for the composition of flexible processes. We then give an overview of the COOPER platform. We finally review the main related works and draw our conclusion.

Rationale and Motivation

The design of the collaboration environment described in this paper has resulted from the analysis of project-based education at the users' institutions involved in the COOPER project [1], namely: ALaRI (Advanced Learning and Research Institute – www.alari.ch), a master school of the Università della Svizzera Italiana, ASP (Alta Scuola Politecnica – www.asp-poli.it), founded by Politecnico di Milano and Politecnico di Torino, and CoWare (www.coware.com), a leading supplier of system-level electronic design automation (EDA) software and services.

ALaRI and ASP are academic institutions. The learning activities of their master students are organized around geographically distributed teams working on projects. In particular, teams need to create their own work plan and use communication tools and self-defined processes to meet their goals.

CoWare addresses an industrial context. However, since it licensees work in distributed communities around the world, having a common training environment would assure that all the users (employees, partners, customers, etc.) are equally able to exploit the power of automation in electronic design. Currently there is no unified training curriculum, and quality of training packages varies by region. There are "centres of excellence", but there is no systematic program to disseminate the training.

In COOPER, we are also investigating the implementation in our framework of the *virtual company* scenario [10], which situates learning in a virtual business environment, enabling learning-by-doing. The virtual company supports reflection on learning processes of individuals, of teams and of the organizations in which they operate. Such a learning scenario can take great advantage from a flexible collaborative environment, in which learning can take place by allowing students to define (and then possibly adapt) their project plans or assessment criteria, thus leading to personalization and evolution in their working and learning processes [10].

In all the previous scenarios, learning activities are centered on the development of projects by virtual team of students. Therefore a major requirement is to facilitate the organization of collaboration tasks. We realized that the plans and the processes generally followed during project development are the result of consensus after discussions between the team members. Collaborative processes are indeed *user-driven* [3], since they may vary between different teams, in accordance with their preferred cooperation style and are bound to change due to learning.

As a consequence, we also realized that traditional methodologies for workflow design cannot easily accommodate such flexibility requirements: once the workflow application is produced and deployed, it becomes difficult (or even impossible) to change the workflow at runtime. We have therefore developed a Web-based collaborative environment, which overcomes such limit by providing learners with mechanism to define and flexibly modify their collaborative processes, even during process execution.

Collaboration Activity Libraries

Supporting the dynamic composition by learners of flexible processes requires the availability in the environment of collaboration activities that can potentially be combined into processes. Our framework is based on a library of *atomic activities*, which we have identified by analyzing the typical tasks that are performed on a regular basis during project work and that are reusable in several process contexts.

Currently the library includes some forty atomic activities, formulated as “stand-alone”, but still composable, components that can be classified as follows:

– *Teamwork planning:* activities in this category support the organization and the scheduling of the team activities. The corresponding atomic activities are: “Assign roles”, “Collect team member competencies”, “Define tasks”, “Assign tasks”, “Agree on task division”, “Define milestones”, “Plan deliverables”.
– *Resource management:* this category refers to the activities for publishing, accessing or also recommending resources (i.e., documents, forum messages, wikies, etc.). The related atomic activities are: “Publish resources”, “Acquire resources”, “Recommend resources”.
– *Communication:* this category groups the activities related to the invocation of synchronous communication services and asynchronous communication tools. The atomic activities in this category include making a VOI call; creating, opening, moderating, joining, and closing of synchronous activities (such as one-to-many video-conference), moderating meetings, chat rooms, co-browsing or co-editing-sessions; voting through polls.

– *Reviewing and Assessing:* this category covers some reviewing activities, as well as assessment for team members, for themselves and also in the context of the project team. Activities in this category are “Creating Review Reports”, “Designating reviewers”, “Submitting reviews”, and also “Define assessment criteria”, “Define performance indicators”, and “Plan assessment”.

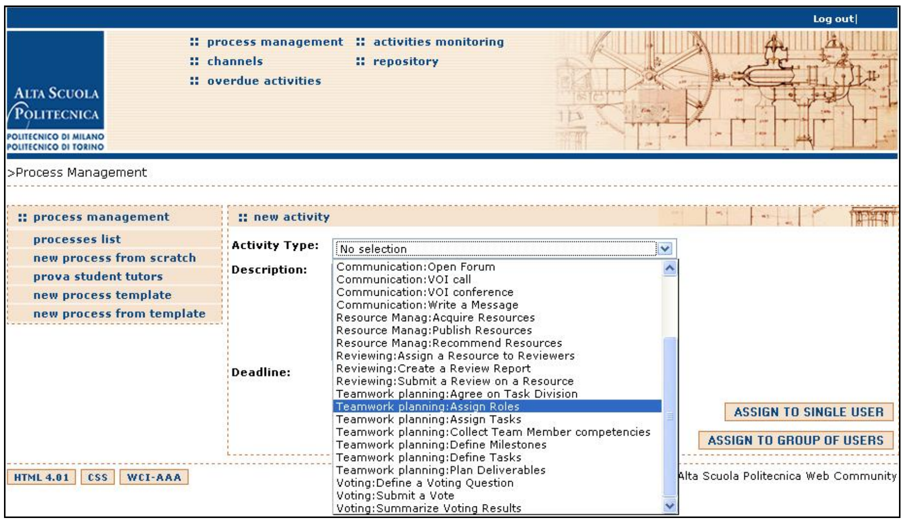

Figure 1. A Web page for activity selection

Web-Based Definition of Processes

Process definition by end users requires the selection of atomic activities from the library, and the definition of some constraints controlling the activity assignment to users and resources, as well as the flow of activity during process execution. As also recognized in literature [3][4], the processes that the teams might need to define are in general simple. Our current implementation of the collaborative environment therefore supports process definition by means of form-based Web pages.

An example of process definition is shown in Fig. 2, where the user selects the type of activity (e.g., “Assign Roles” from the library) and enters a short activity description. S/He is then required to assign the activity to other users. In case of multiple actors executing an activity, the user then needs to indicate the kind of parallelism governing the activity execution. Depending on the type of activity, the user might also associate the activity to some resources to manage possible documental flows. Through similar form-based pages, the user can define complex processes, composed of arbitrarily nested blocks of parallel activities. S/He can also modify the definition of existing processes.

The semantics behind process definition and modification can be explained by means of the composition constructs that can be used, which we illustrate in the following.

Shared activities. Given an atomic activity, some constraints might be required to synchronize multiple actors enrolled for that activity. When a same activity is assigned to several actors, the process definer needs to specify that, in order for the activity to be completed, all the enrolled actors must execute them (AND logic), or at least one actor must execute an activity instance (OR logic), or one and only one actor must execute an activity instance (XOR logic).

Activity Composition. Activities can be composed with one another to build complex processes according to two main composition patterns:

- *Sequence: Activity n* can be executed only after *Activity 1* up to *Activity n-1* are over.
- *Block of activities:* a block is a group of branches of activities that can be executed in parallel, without any precedence constraints. Therefore, a single user assigned to multiple parallel activities can choose to execute them in any order; different users can start performing different activities in parallel. Blocks are delimited by *gateways*, which act as synchronization points for parallel activities: the activities following an opening gateway (*split*) become ready only when that gateway is activated by the conclusion of some precedent activity (if any); activities afterwards the closing gateway (*join*) can begin only after the join is successfully evaluated, according to its nature which depends on the required type of activity parallelism (AND, OR, XOR).

Data Flow. Documental resources can be shared among activities and can have different scopes:

- *Single Activity:* resources are created/accessed by a single activity.
- *Process:* resources are visible to all the activities in a process; if a resource is still to be created, it will be visible to all the activities still in progress after its creation.
- *Group of activities:* Resources are made visible only to a selected set of activities, thus originating a *data flow* inside the process.

Modifying Processes. Different dimensions may be involved when modifying an already existing process:

- *Activities:* an activity may be replaced by another one (e.g., at the end of a Voting process, the activity “Summarizing Voting Results” may become “Open a VOI Meeting”).
- *Users:* activities may be re-assigned to different users or user parallelism rules in shared activities may be changed (e.g., “all” vs. “at least one”).

– *Activity Composition:* activities may be deleted; new atomic activities or compound blocks may be inserted between two existing ones. Gateways may change their nature (e.g., from “and” to “xor”).
– *Data flow:* unnecessary resources may be removed; new, previously unexpected ones may be inserted. The scope of existing resources can be changed.

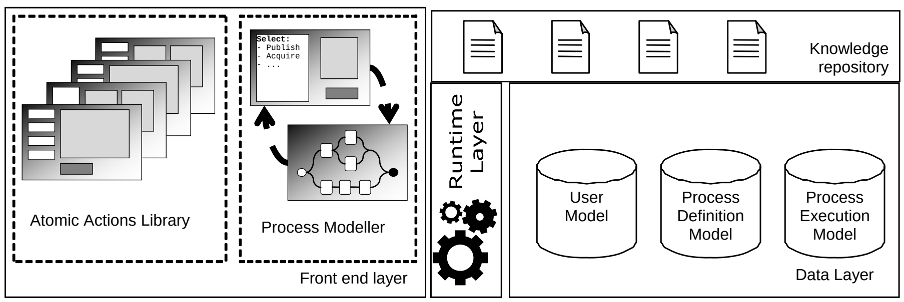

Figure 2. The architecture of the framework supporting flexible processes

Modifying past activities may invalidate the new process and discard the work that has already been done; we therefore only allow for the modification of all the activities that are yet to be started.

System Overview

Figure 2 illustrates the architecture of the framework supporting the flexible management of teamwork collaborative processes. Our approach in particular addresses processes to be delivered on the Web. Therefore, in line with the classical architecture of Web applications, our proposal is characterized by a *data layer*, a *runtime layer* and a *hypertext layer*. The architecture also relies on the availability of a knowledge repository that stores the resources needed by team members for developing projects, which can therefore be the objects of the activities composing a process.

Data Layer

The different concepts underlying the process definition and execution are represented explicitly at the data layer, in form of *process metadata*. As illustrated in Figure 3, the process metadata represent the actors involved into the process (*User Model*), the process model (*Process Definition Model*), as defined by end-users by composing atomic activities, as well as some execution data to control the process execution and also monitor users' activities (*Process Execution Model*). Such a *data-driven* paradigm, which does not bury process data in the application code, allows us to manage the dynamic definition and evolution of processes.

Hypertext Layer

In our framework, which aims to deliver cooperative processes on the Web, the front-end layer consists of the Web pages for the definition and modification of processes (see Figure 1 for an example of such pages), grouped in the *Process Modeler* area, and also of the pages supporting the execution of the atomic activities, which form the so called *Activity Library*.

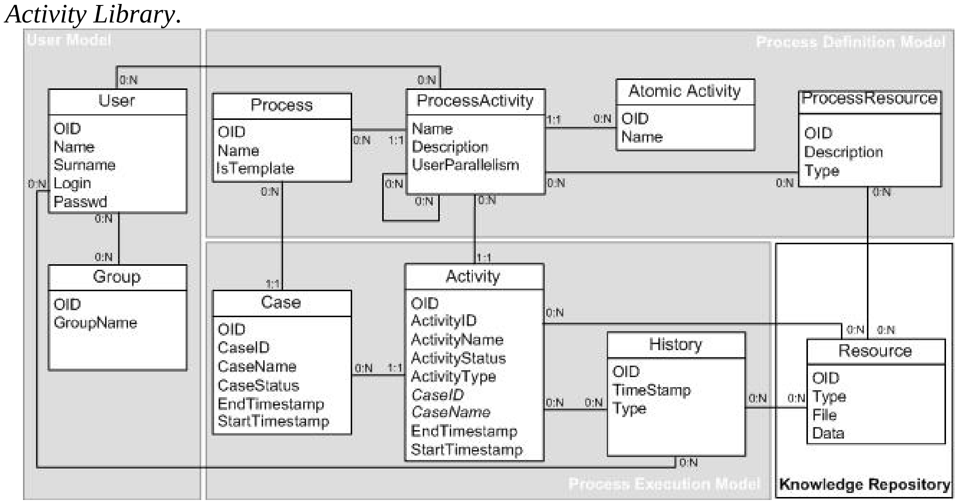

Figure 3. Process metadata, including the User Model, the Process Definition Model, and the Process Execution Model

The interface is developed by means of a model-driven paradigm, which leverages the WebML visual model to specify page design [2], and on its accompanying CASE tool (http://webratio.com) to automatically transform the page visual schemas into running code. This model-driven approach facilitates the addition of an atomic activity, which just requires adding new pages, by modelling them at a high level of abstraction, without taking care of their possible interconnections with the other existing activities, or with the runtime layer modules managing the process execution.

Figure 4 shows the organization of the *Atomic Activity Library*, expressed according to the WebML visual notation. Each atomic activity is provided with an area including the pages for the activity execution. According to the WebML style, the specification of each area makes use of pages, which in turn cluster the content units rendered inside the pages, as well as operation units that model some business logic actions.

Runtime Layer

The runtime layer is in charge of computing and serving the Web pages that allows the users to define the processes. Given the PDM metadata, fed through process definition, the application runtime is also able to control the process execution, even in absence of a workflow management system. Figure 4 schematically represents the operations that, during process execution, the runtime layer performs to dynamically compute the sequence of activities to be followed by each user. A user accesses his/her Home page (the rectangle labelled with an “H” in the left part of the figure) by logging into the platform. By analyzing the PDM metadata, the platform determines the activity the user is enrolled for, and therefore provides her/him with a personalized list of ready activities.

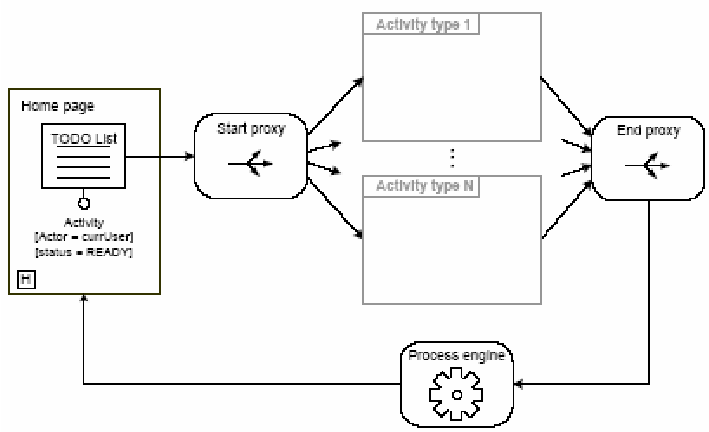

Figure 4. The organization of the atomic activity library

Selecting one of the activities invokes the *Start proxy operation* (representing a business logic operation in WebML), which selects the hypertext area associated with the activity chosen by the user, and forwards the user to the area start page.

Once executed, each activity can be completed by means of a *Complete* command, which connects the activity hypertext area to the *End proxy unit* (at the right hand side in Figure 4) that forwards the hypertext computation to the *Process Engine* operation. This operation is in charge of keeping up-to-date the PEM metadata, required for controlling the process executions. The state of executed activities is set to “complete”. The next activities in the process flow are thus retrieved and managed according to their nature:

– *Atomic activities* are set to “ready” and proposed to users.

– *Splits* are recursively opened to activate parallel branches;
– *Joins* are closed depending on the nature of the block parallelisms:
  1. If the parallelism nature is OR or XOR, the process proceeds by processing and activating the activities following the join;
  2. If the parallelism nature is AND, the process can continue if and only if all parallel branches converging into the join have been completed. If some branches are still open, the join is not processed and the control is passed to the users.
– If an activity is part of a block of parallel branches, if the parallelism nature of the block is “XOR”, all the other branches are deactivated (since only one branch in a XOR block can be active). Otherwise, the state of the first activities of all the branches is set to “ready”.
– After completion of an activity, the user is presented with an updated Home page, showing the new list of ready activities.

Related Work

Some previous works have focused on flexibility and run-time personalization of collaboration processes:
– *Learning platforms* (as for example the one proposed by the IMS-LD Design initiative, WebCT, BlackBoard, IBM LearningSpace) especially offer support to schedule course activities;
– *Ccollaborative platforms* and *shared hypermedia workspaces* offer opportunities for creating task objects and specify some kind of control logic to organize collaboration (see for example CURE [5] and XCHIPS [11]).

Very often, such proposals offer facilities for resource sharing, synchronous and asynchronous communication, course planning and help desk. However, they are still “task-oriented”, not “process-oriented” [5]. Also, (i) very often they are designed to support individual activities, while they do not sustain the schedule and organization of collaborative processes; (ii) they provide a limited set of cooperative tools and the addition of possible new tools and task is difficult or even impossible, especially when such extensions require the definition of data and control flow.

Some proposals address “flexible e-learning”, and introduce environments where collaboration is driven by flexible, yet controlled, means of progressing through processes ([5][7][9]). Also based on workflow technologies, such approaches ensure flexibility, and also allows for monitoring of learners activities. However, they make use of user interfaces for coordination specification through notations that require users to learn concepts and primitives related to process design.

Besides the support for flexible processes, a distinguishing feature of our framework, with respect to the above specialized systems, is the use of an easy-to-use Web interface, whose development leverages a model for the conceptual design of Web applications. The adoption of a conceptual model also introduces a different design paradigm, which does not require dedicated software support, even when the extension with new tools and task is required. In fact, differently from the collaborative software development kits, which mostly focus on the implementation-level productivity, conceptual modelling leverages a platform-independent and abstract view of requirements and design, in which all the aspects of a collaborative application are expressed in a declarative manner.

Conclusions

In this paper we have presented a modelling solution, enabling the run-time user-driven definition of flexible collaborative processes. This solution has been implemented in an educational scenario for academic and industrial training, which supports collaboration in team-based project learning.

Currently, we are conducting some experiments to evaluate the collaborative environment; in particular, the current prototype is in use within a restricted sample of ASP students. The ASP users found the facility we offer for dynamic process definition useful [8]. As expected, they however found the form-based Web interface not adequate for the composition of complex, nested processes. Based on this feedback we are now implementing a client-side module supporting a light BPMN notation.

Our collaborative environment has resulted as particularly useful in the academic and industrial domains where project-based learning is crucial. We however believe that the conceived solution and the proposed framework architecture have a general value for the management of dynamic process, and can be replicated as well in other domains requiring process flexibility.

Acknowledgments

This work is founded by the FP6 EU Project COOPER (IST – 027073). We are really grateful to Dr. Jan van Bruggen and to Drs. Ellen Russman for the help offered in the identification of atomic activities.

References

[1] A. Bongio, et al. COOPER: Towards a Collaborative Open Environment of Project-Centred Learning. Proc. of EC-TEL 2006, 2006, Springer Verlag, pp. 561-566.

[2] S. Ceri, P. Fraternali, A. Bongio, M. Brambilla, S. Comai, M. Matera. *Designing Data Intensive Web Applications*. Morgan Kaufmann, 2002.

[3] F. Charoy, A. Guabtni, M. Valdes Faura. A Dynamic Workflow Management System for Coordination of Cooperative Activities. Proc. of the First Int. DPM '06 Workshop, 2006.

[4] P. Dillenbourg, M. Baker, A. Blaye, C. O'Malley. The Evolution of Research on Collaborative Learning. In E. Spada and P. Reiman (Eds), *Learning in Humans and Machine: Towards an interdisciplinary learning science*, Oxford: Elsevier, 1996, 189-211.

[5] J. M Haake, T. Schmmer, A. Haake, M. Bourimi, B. Landgraf. Supporting flexible collaborative distance learning in the CURE platform. Proc. of HICSS'04, IEEE Press, 2004.

[6] J. Lin, C. Ho, W. Sadiq, M.W. Orlowska. Using Workflow Technology to Manage Flexible e-Learning Services. *Educational Technology & Society*, 5, 4, 2002.

[7] T. Nodenot, C. Marquesuzaa, P. Laforcade, C. Sallaberry. Model-based Engineering of Learning Situation for Adaptive Web Based Educational Systems. Proc. of WWW 2004, 2004, 94-103.

[8] V. Posea, S. Trausan-Matu, V. Cristea. "Online Evaluation of Students' Opinions about the Collaborative Learning System they Use", submitted for publication to ICCP 2007 (International Conference on Intelligent Computer Communication and Processing).

[9] S. Sadiq, W. Sadiq, M. Orlowska. Pockets of Flexibility in Workflow Specifications. Proc. of ER'01, 2001.

[10] H. Spoelstra, M. Matera, E. Rusman, J. van Bruggen, R. Koper. Bridging the gap between instructional design and double loop learning. Proc. of m-ICTE'06, 2006.

[11] W. Wang, J.M. Haake, M. Wessner. Supporting Cooperative Learning in Distributed Project Teams. Proc. of CoopIS 2002, Springer, 2002.