Palette
Pedagogically sustained Adaptive Learning through the Exploitation of Tacit and Explicit Knowledge

Information Society Technologies

Project no. FP6-028038

PALETTE

Pedagogically sustained Adaptive LEarning Through the exploitation of Tacit and Explicit knowledge

Integrated Project

Technology-enhanced learning

D.MED.12 –Developing collaboration services for CoPs: Lessons learned and future work directions

Due date of deliverable: December 31, 2008
Actual submission date: February 02, 2009

Start date of project: February 01, 2006 Duration: 36 months

Organisation name of lead contractor for this deliverable: CTI, EPFL

Project co-funded by the European Commission within the Sixth Framework Programme		
Dissemination Level		
P	Public	**PU**

Keyword List: Collaboration services, CoPe_it!, eLogbook.
Responsible partners: CTI & EPFL

MODIFICATION CONTROL			
Version	***Date***	***Status***	***Modifications made by***
1.0	January 19, 2009	Draft	CTI & EPFL (version sent to the evaluators)
2.0	January 26, 2009	Draft	CTI & EPFL (version sent to the SC)
3.0	February 02, 2009	Final	CTI & EPFL (version sent to SCO and AFC)

Deliverable managers

- Manolis Tzagarakis, CTI
- Sandy El Helou, EPFL

List of Contributors

- Nikos Karacapilidis, CTI
- Nikos Karousos, CTI
- Vassilis Kallistros, CTI
- Denis Gillet, EPFL

List of Evaluators

- Amaury Daele, UNIFR
- Aida Boukottaya, UNIFR

Summary

This deliverable reports on fifteen practical lessons learned while developing the collaboration services of Palette, namely CoPe_it! and eLogbook. The lessons reported concern issues related to the process of developing the above tools, the collaboration *per se* and the technologies used. Various sources were taken into consideration in order to articulate these lessons, including evaluation studies based on questionnaires, interviews and discussions in teams, as well as indirect observation of the interaction in CoPs, based on the analysis of appropriate log files. The lessons are presented in a way that could aid people engaged in various phases of the development of Web-based collaboration support services.

Table of contents

1. Introduction

Aiming at supporting collaborative learning within Communities of Practice (CoPs), two collaboration support tools were developed and put into use in the context of the PALETTE project, namely CoPe_it! and eLogbook.

CoPe_it! (http://copeit.cti.gr/) is a Web-based tool, which aims at assisting and enhancing collaboration activities held among CoP members. It provides a cognitive argumentation environment which stimulates reflection and discussion among CoP participants, giving emphasis on the provision of various visualizations (views and projections) of the argumentation discourses, use of reasoning mechanisms on more formal depictions of the collaborative workspace. Users may join or create new communities, collaborate in different workspaces and also have the ability to work in private spaces in order to better organize and depict their view of the ongoing knowledge process. Emphasis is put on allowing members of the community to depict their arguments in varying levels of formality: one may start with more informal depiction of its arguments and then start refining, structuring and linking them with those of other users using semantics which are interpretable by the system, and thus enabling the availability of advanced decision making support mechanisms.

eLogbook (http://elogbook.epfl.ch/) is a Web 2.0 social software application that aims at sustaining collaboration and learning in CoPs. It offers community members a networking and communication platform, a repository for sharing and managing resources, a task and activity management system, as well as a community structuring tool that allows one to define roles and distribute tasks. It also provides different types of notifications (via email, or RSS feeds) in order to motivate contribution and sustain collaboration.

Both tools were developed following the participatory design approach, which was carried out in teams comprising CoPs members, CoPs mediators, service mediators, software developers, and scientists from the fields of education and pedagogy. Their services were defined according to the needs of the various CoPs involved in PALETTE. During the project's duration, the tools evolved (and continue to evolve) based on the feedback received from the above teams. This process allows each tool to constantly improve its usefulness and

usability, hence increasing its acceptability among users. The composition of the teams, and in particular the balance maintained as well as the cooperation pursued between CoPs members and software developers, eradicated cases where there was a single-sided (or biased) interpretation of what CoPs needs should be developed. Thanks to the PD approach, the usefulness and usability of both tools have been constantly improved during the course of the project and successfully met the CoPs requirements, hence increasing their acceptability among users[*].

In this deliverable, we report on fifteen practical lessons learned while developing and deploying CoPe_it! and eLogbook through the abovementioned PD process[†]. Similar work carried out in the context of WP1 has been taken into account[‡]. Moreover, we comment on related future work directions.

2. Lessons learned

The lessons described below are classified according to three perspectives: (i) the process of developing collaboration services, (ii) the collaboration concept *per se*, and (iii) the technologies used to implement the respective tools. Their presentation aims at making them serve as best practices for people involved in building applications for CoPs or any other type of online communities. For each lesson, we outline its context and discuss characteristic instances from various stages of the project's development. It is noted that some of these lessons may also apply to other types of services, also developed in the context of the project, such as information and knowledge management services.

2.1. Collaboration Service Development Process

An important success factor in the development of the Web 2.0 collaboration services was the adoption of the participatory design (PD) methodology to

[*] For details about these requirements as well as about how these were met, see [1].
[†] It is noted that a detailed description of how collaboration and learning within CoPs can and should be effectively supported can be found in [2].
[‡] See the work about methodological instruments and good practices, as described in [3].

help ensuring that CoPs needs are fulfilled and appropriately supported. During PD, end-users were continuously and actively involved into the design process, being considered as an integral part of the development team.

<u>Lesson 1:</u> For PD to work in the context of CoPs, a continuous negotiation of usefulness involving developers and CoP members should be carried out. One promising approach to make PD effective is to consider the development team as an online community with its own mediator, namely the service mediator, able to negotiate with the CoP mediator. For Distributed Participatory Design (DPD) to be effective, collaboration tools such as wikis and synchronous communication tools are essential as they help in harvesting and explicating design- related considerations.

Since the beginning of the project, CoP representatives worked closely with service mediators in order to identify and then convert CoPs needs into design specifications [4]. Teams of developers and CoPs representatives were formed in order to work closely and negotiate the different features that would be offered or not. Once the teams were set up, the question of how to coordinate the collaboration among all participants came up. To address this issue, the idea to conceive the development team as a multidisciplinary online community proved helpful. This approach led to the use of collaboration tools that CoPs already use such as Wikis. This greatly improved the collaboration and coordination of the work within teams. It permitted the gathering of the necessary requirements, as well as the discussion and review of design decisions. In addition, the resources created in the course of collaboration acted as a project memory capturing the design rationale of features. In the same way, as soon as a first version of eLogbook was put online, the idea of forming a community around the tool for its design, evaluation and continuous evolution was pushed further. CoPs mediators and members, as well as tools' evaluators, were encouraged to use the tool itself to express their satisfaction or dissatisfaction vis-à-vis of the usefulness and usability of the different features it offers. This is how one project member, whose task was to

study eLogbook's usability, created a list of reported bugs and questions within a dedicated space and used the eLogbook chat feature to discuss the different issues with eLogbook developers. This helped her to get acquainted with the tool and discuss its pros and cons simultaneously.

<u>Lesson 2:</u> Frequent and early tool releases help in identifying design problems and bugs. In addition, such a strategy stimulates CoP members in using the tool.

Although this principle has been adopted in the open source community, it was especially critical for services such as CoPe_it! and eLogbook. In particular, the various ways provided by the services for sharing and exchanging resources led to using functionalities in ways that were unanticipated by the design team and that have been witnessed only during deployment by real users and in real situations. In addition, trivial functionalities that may have gone unnoticed during the design phase but are essential for collaborative work were spotted early.

CoPe_it! scheduled roughly two releases per year. Each release was announced in advance to all CoPs and the new functionalities were explicitly presented on the service's support Web site. eLogbook pushed further this release paradigm by continuously updating the public online version every time a new feature was added or an existing one was improved. The frequent releases gave the opportunity to all CoP members to comment on the existing features and to propose new ones, which could augment their collaboration, as the appropriation of a new service in part of the identity building of a CoP. For example, by early releasing new versions of CoPe_it!, feedback of users identified the need to easily reference resources that were brought into the workspace, in order to reuse them across different workspaces. The design team of CoPe_it! did not initially consider this feature, but its usefulness to the CoPs led to the implementation of a REST-based service of identifying any individual resource within CoPe_it!. As such shortcomings were very early discovered in the process, the cost of adding and implementing them was low, without impacting the development plan of CoPe_it!.

The frequent and consistent releases of CoPe_it! and eLogbook had another profound impact on the attitude of users towards the tools: it gave the impression of tools that are "alive", constantly evolving and improving, which stimulated the user's interest on how to use them and what features the next release would have. This can be considered as an additional awareness feature. This interest was documented by users' messages that asked to get informed whenever a new version or a new feature was made available. CoPs members were establishing a personal relationship with the tools and were interested in their evolution.

<u>Lesson 3:</u> Log early, log often, log everything. Logging should be a built-in feature of any social software.

Although logging the actions of users during their use of a service is in general conceived as an important feature of any tool, it is nevertheless a feature that is typically targeted rather late in the development process. Yet, being able to fully log all user actions even in very early releases of the tools is crucial for the assessment of existing functionality and the design of new ones. This is especially true for collaboration support software, as some collaborative features, such as how people share and exchange, are difficult to be assessed in a live or 'in vivo' setting. The experience showed that log files provide valuable feedback for the evaluation of the collaborative features of the tools.

The first releases of CoPe_it! paid little attention to logging functionalities, which resulted in poor visibility on the team's side with respect to the features users exploited most, or how exactly they exploited the features provided. Lack of logging information made it also difficult to spot potential performance problems, which for CoPe_it! that utilizes AJAX technologies was a critical factor. Subsequent releases integrated logging of all user actions that gave valuable insights on how to improve its features. For example, logging all user actions revealed operations that executed slowly, which in turn proved harmful for the collaboration of users that were unable to achieve smooth and unobtrusive interaction.

2.2. Augmenting collaboration

While the previous subsection reported on lessons learned related to the process of developing collaborative services, this one deals with the experiences gained as far as the aspects of the services that contribute to achieving effective collaboration within CoPs are concerned.

<u>Lesson 4:</u> Collaboration services must offer engaging environments to users.

The advent of the so-called Web 2.0 era saw the coming of Web applications that provided their services in ways that differ substantially from traditional Web applications. These new applications avoided form-based interaction and introduced new ways that mimic desktop applications. These new ways of interaction appear under the umbrella term of 'rich user experience'. The plethora of Web applications that adopt this style of interaction are increasing, and is in general considered as one of their success factors. Users in this new Web 2.0 era are accustomed to such interaction styles thus making them ad hoc standards for applications to come. CoPe_it! and eLogbook adopted these new interaction paradigms in an attempt to create engaging environments to support collaboration. In particular, both tools focused on providing easy-to-use user interfaces that exhibit great responsiveness to user actions. For instance, AJAX pop-up windows replaced traditional forms of interaction. Feedback showed that such interaction styles were well received by users that had no difficulties in using the services.

Another factor determining the success and the appropriation of services is the creation of participation incentives. This requires having low entry and participation barriers and can be achieved by offering quick registration and easy collaborative authoring (e.g. as it is the case in wikis).

An additional important aspect affecting a tool's appropriation is making the user aware of what is private and what is public. Sharing is an essential feature in any collaborative application and should be made easy. But what is even more important is alerting the user of who has access to his/her own resources

and making it easy for him/her to revoke the access right, when deemed necessary. For instance, in eLogbook sharing a space, an asset or a tag is easily performed. Moreover, when a resource is chosen as a focus element, its different contextual accesses are displayed and the access rights can be easily revoked. This makes the user trust the tool as he/she has full control over what his/she owns. This encourages him/her to rely on this tool to share resources with others.

Lesson 5: ***"Empowering" users does not mean to give them full control; some choices must remain within the system.***

The trend in today's Web 2.0 user-centric applications is to follow a bottom-up approach and allow user to self-design their collaboration spaces. However, allowing users (especially novice ones who are not yet acquainted with an application) to tune everything can quickly become confusing and overwhelming, unless it is done in an ergonomic way. For instance, managing assets ownership rights is very flexible in eLogbook. Users can grant ownership on a person-dependent, role-dependent or activity dependent-basis. The view is done using AJAX, which makes the response to the user's action when it comes to adding, updating or even removing a right very easy and quick. The first users experience with these features revealed a side effect related to the ease of deleting assets rights: users «mistakenly» deleted their own rights. Sometimes when they were the only asset owners, it became impossible to manage the asset in question, as only its «owner(s)» can do it. Usual warnings («Are you sure you want to delete this right?») were not enough to prevent users from doing such actions. Thus, a new rule that prevents the last owner of an asset or a space was imposed. The lesson is that when designing a system, one should keep in mind that people will try everything, something that may result to both expected and unexpected behavior.

Lesson 6: ***Innovative metaphors of collaboration must be introduced in a way that is close to what users are expecting.***

Whenever radical new and innovative metaphors to collaboration are provided to users, these must be carefully introduced, as there lurks the danger of tool rejection due to encountering new and unexplored territory. In general, when users get to use collaborative systems they expect (based on their experience) to fall on traditional categories of wikis, discussion forums and tagging systems, as these are the prevailing systems nowadays on the Web. Our experience showed that radical new ways to collaboration are initially causing confusion rather than excitement.

In the case of CoPe_it! for example, initial evaluation feedback showed that the adoption of a spatial metaphor to facilitate discussions was not immediately grasped by users, and caused great confusion with respect to how to use the tool (Fig 1).

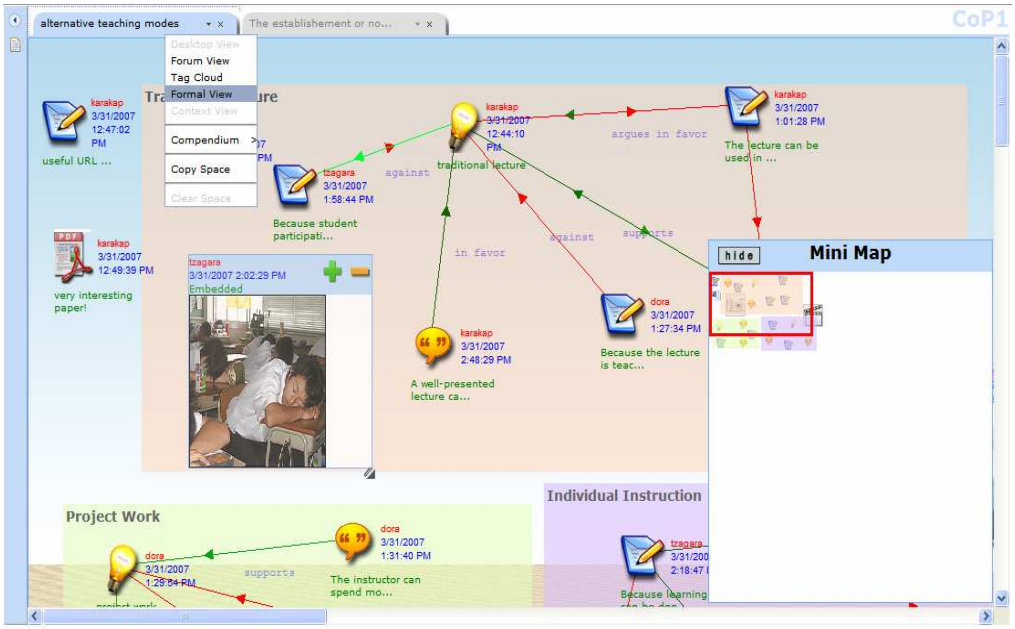

Figure 1: Spatial metaphor of discussion as provided by CoPe_it!

To address such concerns CoPe_it! made improvements on two fronts: first it revised and enhanced its help and tutorials emphasizing on providing examples in order to convey its use. Second, it extended its functionality and included the ability to render the discussion in a way that is familiar to users. For this, CoPe_it! enabled a forum-like view of the discussion (called time-order-view in CoPe_it!) that displayed the discussion in a way that is found in

traditional Web based discussion forums. This functionality proved very helpful as the spatial metaphor of workspaces (constituting CoPe_it! main way of supporting collaboration) was now regarded as simply another way of viewing and conducting the discussion, amongst others, with which the user was already acquainted to. This also enabled users to easily see the value added aspects of conducting discussions spatially by comparing it to traditional ways to supporting discussions.

With respect to eLogbook, in its first releases, the interface consisted of a mirror of eLogbook 3A model influenced by the actor-network and activity theory [5]. Even though applying a general theory appeared to be nice at the conceptual level, it ended up confusing the first eLogbook users. As a matter of fact, they were expecting familiar labels such as "group", "community", and "community space" and found all these constructs or labels fused into one general term "activity". In fact, the initial idea was to focus on the objective of a collaboration space rather than on the space itself. So the community members would be gathered in one mother activity which objective was nothing but the reason behind the creation of the community. In turn, this mother activity can consist of different activities corresponding to different project within the community. Nevertheless, at the presentation level, this naming was not suitable and had to be replaced by more familiar and concrete terms such as "collaboration space". Moreover, the tools acceptability increased when users were allowed to distinguish different collaboration spaces, define new ones and reuse the ones created by others like "communities of practice", "group", "group of interest", "team", "theatrical club", "committee" so on and so forth.

<u>Lesson 7:</u> Formality in collaboration management should not be considered as a predefined and rigid property, but rather as an adaptable aspect that can be modified to meet the needs of the tasks at hand.

Generally speaking, when engaged in the use of existing technologies and systems supporting collaboration, users have to follow a specific formalism.

More specifically, their interaction is regulated by procedures that prescribe and - at the same time - constrain their work. This may refer to both the system-supported actions a user may perform (e.g. types of discourse or collaboration acts), and the system-supported types of collaboration objects (e.g. one has to strictly characterize a collaboration object as an idea or a position). In many cases, users have also to fine-tune, align, amend or even fully change their usual way of collaborating in order to be able to exploit the system's features and functionalities. Such formalisms are necessary towards making the system interpret and reason about human actions (and the associated resources), thus offering advanced computational services. However, there is much evidence that sophisticated approaches and techniques often resulted in failures [6]. This is often due to the extra time and effort that users need to spend in order to get acquainted with the system, the associated disruption of the users' usual workflow, as well as to the "error prone and difficult to correct when done wrong" character of formal approaches.

Complex contexts, as those that are usually encountered during collaboration, imply additional disadvantages when using formal approaches. Such approaches impose a structure which is not mature enough to accommodate the management of huge amounts of data coming from diverse sources. They do not allow users to elaborate and digest these data at their own pace, according to the evolution of the collaboration. Instead, a varying level of formality should be considered. This variation may either be imposed by the nature of the task at hand (e.g. decision making, deliberation, persuasion, negotiation, conflict resolution), the particular context of the collaboration (e.g. medical decision making, public policy making), or the group of people who collaborate each time (i.e. how comfortable people feel with the use of a certain technology or formalism).

Incremental formalization of collaboration, which was adopted in the development of CoPe_it!, proved to be a successful approach to address the above concerns. In this approach, formality and the level of knowledge structuring is not considered as a predefined and rigid property, but rather as an adaptable aspect that can be modified to meet the needs of the tasks at hand. By the term formality, we refer to the rules enforced by the system, with

which all user actions must comply. Allowing formality to vary within the collaboration space, incremental formalization, i.e. a stepwise and controlled evolution from a mere collection of individual ideas and resources to the production of highly contextualized and interrelated knowledge artifacts, can be achieved.

<u>Lesson 8:</u> Collaboration services do not operate in isolation and must not be regarded as "application islands". Data and services accessibility through seamless interoperability with existing tools is a crucial factor for their adoption and success.

From the users initial needs and ongoing feedback, openness and seamless interoperability appears to be a primordial need and constitute one of the biggest challenges of today's social software applications [7]. Users want to gain "real ownership" over the information that they have provided and/or that belongs to them (e.g. their profile information, projects, friends). They want to be able to easily import/export from one environment to another. They want to be able to synchronize information across different tools and visualize it in different ways via different applications. Moreover, the learning and knowledge management processes in CoPs always exhibit stages that typically require different sets of tools. A seamless integration of distributed tools and services is instrumental for providing of new collaboration solutions. As a plethora of resources are already available on the Web and used by CoP members during their day-to-day tasks, collaboration services must explicitly address issues regarding the integration of these resources into their environments. Otherwise, the danger of becoming isolated may surface and ultimately lead to their rejection.

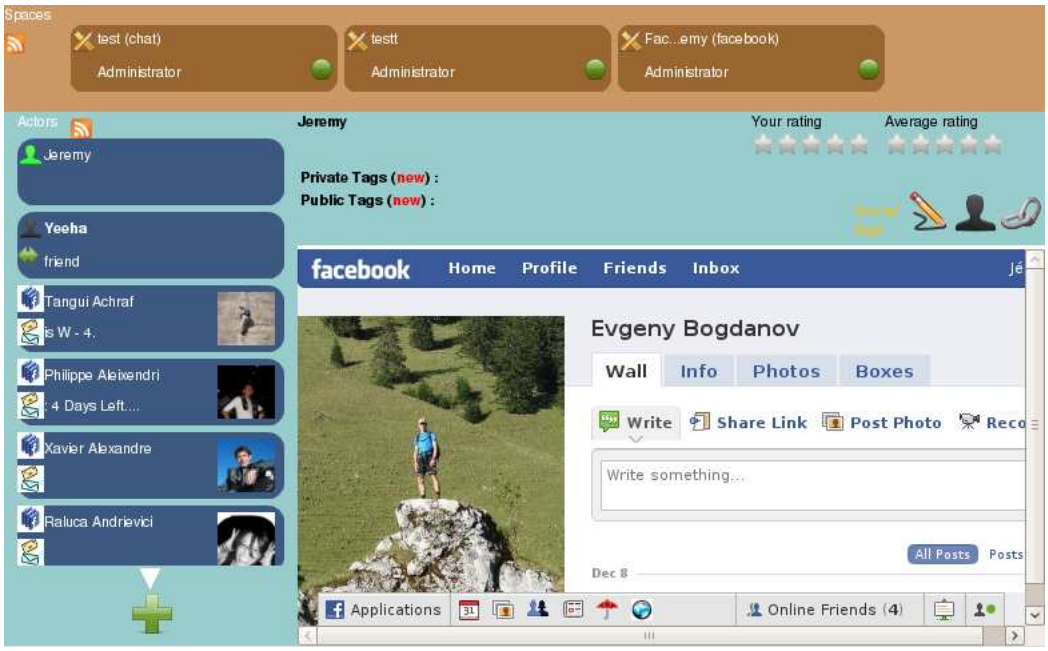

Figure 2. eLogbook with Facebook friends integration.

To respond to the data openness and portability need, different useful interoperability scenarios were designed and implemented. For instance, the possibility to export or propagate profile updates from eLogbook to CoPe_it! is now possible. In the same way, the CoPe_it! profile can be imported to eLogbook. The implementation of OpenID in eLogbook is also intended to respond to the need of having one identity across different services. Moreover, the ability to visualize Facebook friends within eLogbook was included and to combine a Facebook user and a eLogbook user into one single entity (Fig. 2). The final goal was to supply a user with means to manage different social software accounts in one place and have a login-transparent access to different applications.

In CoPe_it! the efforts focused on giving the ability to integrate resources from popular and relevant applications. More specifically, from Compendium [8], a tool to capture design rationale, and from popular Web-based discussion forums, built on phpBB and Moodle. In addition, every resource in CoPe_it! or eLogbook was given the ability to be referenced via a unique and simple URI that enables their use in other applications.

<u>Lesson 9:</u> Awareness must be an integral part of collaboration services.

During the design phase, the study of CoPs requirements revealed the need for notifications services and came to enforce previous studies of the field of computer supported collaboration [9] on the importance of awareness in collaboration environments. Awareness services can immensely improve the collaboration and must therefore be considered as an essential characteristic of these systems. As both tools, CoPe_it! and eLogbook, aim at providing advanced collaboration means to CoPs, which exceed the abilities of existing tools, services are required to help users coping with the sheer number of events that take place. Consequently different types of awareness services were classified according to the literature and the CoPs needs and were implemented in both tools [10]. Push and pull delivery mechanisms were available by offering notifications via email, RSS feeds as well as through the regular Web interface cues.

<u>Lesson 10:</u> Delivering relevant and user-customizable notifications and recommendations are effective ways to cope with the interaction and information overflow.

The role of any social software application in general is to sustain collaboration. This is achieved by providing different features for interactions such forums, wikis, blogs, and chat, as well as keeping users aware of what is happening in the community and motivating them to participate. Nonetheless, overwhelming the user with unnecessary notifications may lead to adverse effects [11], discouraging him/her from following up what is happening in the workspace and reacting to it, which decreases his/her collaborative activity. Moreover, with the low-entry and low-participation barriers of the new Web 2.0 applications (namely, ease-of-use, ergonomic design, and engaging environment), the information flow can quickly become overwhelming, especially when it comes to open communities. Even in the case of small and private communities, the assets produced increase fast. For instance, in spring

2007, eLogbook was used by a group of higher-education students taking a project-based course that spans over one semester. The total number of assets produced during only 3 sessions was 177. So, while the main challenge of earlier CSCW applications was to trigger participation incentives, the new challenge of the new Social Software applications, is to be able to provide each user with relevant and personalized news and resources from among the numerous ones available. To elaborate, the aim is to rank the online resources available and the events related to people, spaces or knowledge artifacts according to their importance for a particular user in a particular context. The importance can be calculated based on explicit as well as implicit factors. Explicit factors take into account the student preferences, while implicit ones are inferred from the user's interaction with the environment. Currently, in eLogbook, a modified version of the Google PageRank algorithm is being implemented, in order to get a set of important events, spaces, and knowledge artifacts potentially relevant to the user. The algorithm takes as input a matrix representing all the different relations that are deemed significant for computing the importance of an item with respect to the user. For instance, the actions of frequently visiting a space, modifying a document, tagging a person are taken into account. What makes this approach unique is that it takes several parameters into account and allows the user to graphically visualize how the importance is calculated and directly tune the weight of the different types of relations according to his/her preferences.

<u>Lesson 11:</u> When it comes to usability, less is sometimes more; choosing whether or not to show an advanced feature is crucial for its acceptability.

According to the Technology Acceptance Model [12], for a tool to be accepted, it has to be useful but also usable. Ease-of-use plays a very important role in the service acceptability, and same as features evolve with time based on users' feedback, usability issues should also be depicted and handled to guarantee user's satisfaction and stickiness to the service.

eLogbook was designed and implemented with the focus on making it general and flexible enough to satisfy the needs of different types of CoPs. As a matter

of fact, CoPs differ in the nature of their practices and their objectives. For instance, the CoPs involved in the European project ranged from CoPs of students and tutors to CoPs of managers and engineers. These CoPs have different level of exchange including messages, common products, and structures (e.g. FAQ). Some CoPs are hierarchical in nature with clear roles and tasks distribution, while others have a more flexible and horizontal structure.

		Standard of procedure	
		+	-
Standard of situation	+	Scientific evidence	Story telling
	-	Consultancy	Reflective

Table 1. Classification of CoPs according to standard of procedure and standard of situation.

Moreover, CoPs can be classified into four different categories, as shown in Table 1, according to two parameters: standard of procedure and standard of situation [13]. First, the reflective CoPs are mostly based on social exchange and have very low standard of procedure and situation. Second, CoPs based on scientific evidence, have a high standard of situation and procedure. They have a specific ontology and rely on expertise to take decisions based on every situation at hand. Third, story telling CoPs interact mostly through social exchange and detailed narration. They have low procedural standards, but high standards of situation. Fourth, the CoPs based on consultancy have a low standard of situation and a high standard of procedure. They propose immediate solutions. Moreover, CoPs can evolve with time from one category to another. This motivated the design of an application that is able to answer different CoPs needs, which not only vary from one CoP to another but also evolve over time within the same CoP. Therefore, the different eLogbook features for collaboration, activity and knowledge management were designed in such a way to be general and flexible enough to dynamically respond to the evolving CoPs needs. For instance, when an activity is created, default roles

such as administrator and member are automatically created. Still, the self-nominated administrator can define any number of new roles, with different labels and granular rights associated. This feature is very helpful for hierarchical CoPs. However, it might confuse other less structured CoPs, as they will not understand its purpose or see its usefulness. Here comes the challenge of usability: when is the best time to suggest those specialized feature and where best to put them. At the beginning, these features were shown on the creation of the activity. Then, after some mediators reported that they were confused and that they did not understand the purpose of this feature, it was skipped during the activity creation process. Still, it was made available within the list of icons for managing activities.

2.3. Technologies

This subsection reports on lessons learned with respect to the technologies used to implement the PALETTE's collaboration tools. As dictated by the project's main objectives, both tools placed emphasis on the use of standards. In addition, both adopted state-of-the-art technologies and environments to implement the planned features. The lessons learned report on the effectiveness and impact of the adopted technologies on the development process, i.e. whether they perplex or simplify it.

Lesson 12: ***AJAX can be a sound framework to implement responsive and synchronous collaboration over the Web. However, the lack of proper development tools makes it difficult to achieve the level of robustness required.***

The use of asynchronous Javascript and XML (commonly referred to as AJAX) is nowadays a popular technique to achieve interactive and responsive Web applications. The use of AJAX permits overcome page-based navigation, enabling only parts of a page to be refreshed with data from the server.

CoPe_it! used AJAX not only as a substitute to traditional forms but also as the core mechanism to implement the spatial aspect of workspaces. Every

time users make an action on the workspace, an appropriate AJAX request is sent to the server indicating the operation carried out . CoPe_it! took also AJAX to the extreme and implemented synchronous collaboration within workspaces over the Web: any change in a workspace was made visible to all users that were viewing the particular workspace. Due to specific characteristics of the synchronous collaboration (anticipated work load, number of participants, rate of events etc), CoPe_it! adopted a polling approach. The use of AJAX to support synchronous collaboration over the Web's infrastructure is a new trend and reports on its effectiveness are rather sparse. The development of CoPe_it's gave us opportunity to assess AJAX more thoroughly in this regard.

The smooth and robust functioning of the synchronous collaboration mechanism as it was witnessed during its use in real collaboration sessions of CoPs, showed that when specific collaboration conditions exist, AJAX-based polling approaches can be a sound architecture to implement synchronous collaboration over the Web. The specific collaboration conditions refer to the number of users collaborating in a workspace and the frequency of events generated during the collaboration. The evaluation showed that if the number of collaborating users is small and the frequency of events is low, AJAX based polling is a feasible and robust solution to the problem of synchronous collaboration. When however the number of concurrent users and events increase, then AJAX based polling reaches its limits. In such situations, loss of events is possible which prohibits synchronous collaboration.

Although AJAX can be a robust framework for advanced collaboration modes and is in general easy to use, achieving the required level of robustness is a difficult task. Two reasons contribute mainly to this: the lack of development tools to support AJAX based development and the differences in its implementation in different browsers. The lack of proper tools to support the development and in particular the debugging of AJAX based approaches complicates the process, as efforts must to be put into setting up the proper environment that will enable catching malfunctions and tracing of bugs. Another aspect that complicates the process is the fact that the implementation of the AJAX mechanisms vary in different browsers. In particular, the same AJAX calls may exhibit different behaviour when

executed in different browsers. In order to achieve consistency of the application across different browsers, these differences must be addressed which nevertheless requires great efforts.

<u>Lesson 13:</u> Standards are a catalyst for interoperability. They greatly reduce the time and effort required to develop interoperable systems.

The PALETTE project had been early committed to the adoption and use of widely adopted standards. In line with this, both PALETTE collaboration support tools based their interoperability efforts on such standards. Appendix A summarizes a list of standards that both tools adopted along with a short description reporting on the interoperability context they were used. The adoption of standards aided significantly the development of both tools for two reasons. First, they minimized the communication between the distributed development teams. Second, the use of standards opened the road to employ a plethora of open source tools, which greatly simplified the development tasks.

<u>Lesson 14:</u> In the context of collaboration services, REST and SOAP-based services were both equally expressive in capturing and publishing part of the tools' functionalities. While SOAP-based approaches provide the mechanisms to address a wider range of concerns, they introduce in general a greater overhead to the development process than respective REST-full approaches.

Both tools employed REST and SOAP based approaches to make part of their functionalities available as services to external tools. CoPe_it! for example provided SOAP-based services for its repository services as well as for its user management services. In addition, it integrated knowledge management tools via SOAP-based services. CoPe_it! provided also a set of REST-full services. For example, the ability to synchronize profiles with eLogbook is solely based on REST-full services.

What the use of both approaches showed was that although both were equally capable in capturing and publishing functionalities, REST-full services in general achieve this at a lower cost. Our experience showed that this is due to two reasons: first, REST-full approaches are much simpler and align perfectly well with the WWW infrastructure than SOAP-based ones. Second, SOAP-based Web Services attempt to cover a wide range of concerns (from simply describing services to their orchestration) and hence feature more complicated options and formalisms that in general place additional overhead onto the development process.

<u>Lesson 15:</u> Open source software can be reliably used for developing challenging collaboration systems.

Both tools based their development on existing open source software and libraries that were available for download on the internet[§]. What their use showed is that they exhibit high quality, which is comparable to the quality of commercial counterparts. Here the term quality is used to denote whether the software actually achieves what it states, along with what resources are provided to support the development process. In some situations, with respect to issues concerning the compliance to standards, open source outperformed commercial software.

3. Future work directions

Both tools achieved to develop all foreseen and necessary functionalities that permit CoPs to address their collaboration needs. Yet, as these tools represent innovative ways to support collaboration in online communities, they are under constant development and evaluation. In this section, we present some future work directions related to both tools, the overall aim being to further improve the collaboration in diverse types of online communities. These directions focus on three issues: ubiquitous computing, context-sensitive

[§] For instance, as far as CoPe_it! is concerned, more details can be found in [14] (Appendix B).

computing and personalization, and Serious Games. The next paragraphs give further details on each of them[**].

3.1. Ubiquitous computing

With respect to ubiquity of the collaboration, future work will concentrate on permitting participation to the collaboration via small handheld devices such as mobile phones. Although the design of CoPe_it! and eLogbook take the presence of such devices into consideration, they do this only at an very simple level by providing only notification services such as RSS feeds. Participating to the collaboration with the use of such devices is only possible in an read-only mode (i.e. users cannot actively participate). Future work will concentrate on how active collaboration of users is conveniently possible with the use of such devices.

3.2. Context-sensitive computing and personalization

Related to context-sensitive computing and personalization, both tools will focus on taking into consideration parameters of the user's context with which the collaboration environments may be adapted and personalized. These parameters include location-based information so as to let users discover who is in their vicinity in order to give additional incentives for collaboration as well as technological ones, in order to adapt the entire collaboration environment to technological settings of the users. Technological settings may include the device used or the dimension of the screen.

3.3. Serious Games

One additional direction into which both tools will further look into is the convergence of traditional forms of collaboration and serious games. In general, serious games a unique learning experience aiding engagement and immersion. The overall aim of this future direction is to integrate certain gaming elements into social applications. For example, some discussion

[**] A more comprehensive discussion of these issues can be found in [15].

forums deploy some elements of role playing game. Forum members may gain experience points and virtual money through participation in discussion. The members can level up and acquire various titles if they have accumulated enough experience points. They can also use virtual money to exchange various virtual items to decorate their avatars. Deployment of gaming elements is an approach to provide incentives of participation.

4.References

[1] D.IMP.08: Instances of implementation of PALETTE scenarios. Public Deliverable, PALETTE project, 2009.

[2] D.EVA.06: Final Report focusing on describing PALETTE practices and approach, lessons learned about formative evaluation and lessons on how learning within a CoP can be effectively supported. Public Deliverable, PALETTE project, 2009.

[3] D.PAR.05: Participatory Design Methodological Instruments and Good Practices. Public Deliverable, PALETTE project, 2009.

[4] De Paula, R. A. The Construction of Usefulness: how Users and Context Create Meaning with a Social Networking System. Doctoral Thesis. University of Colorado at Boulder (2004).

[5] El Helou, S., Gillet, D., Salzmann, C. and Rekik, Y. Social Software for Sustaining Interaction, Collaboration and Learning in Communities of Practice. In Solutions and Innovations in Web-Based Technologies for Augmented Learning: Improved Platforms, Tools, and Applications, Advances in Web-based Learning (AWBL) Book Series. Information Science Reference, 2008..

[6] Shipman, F.M. and McCall, R. "Supporting knowledge-base evolution with incremental formalization", Proc. CHI '94 Conference, 1994, pp. 285-291

[7] Buytaert, D. From infinite extensibility to infinite interoperability. Feb. 8, 2008. [Online]. Available at: http://buytaert.net/from-infinite-extensibility-to-infinite-interoperability. [Accessed: Dec. 10, 2008].

[8] Shum, S., MacLean, A. Forder, J. and Hammond, N. (1993). Summarising the Evolution of Design Concepts within a Design Rationale Framework. In Adjunct Proceedings of InterCHI'93: ACM/IFIP Conference on Human Factors in Computing Systems, April 24-29, Amsterdam, pp. 43-44.

[9] Carroll J., Rosson M.B., Convertino G., and Ganoe, G.H. Awareness and teamwork in computer-supported collaborations. Interacting with Computers, Vol. 18, No 1, (2006), 21-46.

[10] El Helou, S., Tzagarakis, M., Gillet, D., Karacapilidis, N. and C. Yu Man. Participatory Design for Awareness Features: Enhancing Interaction in Communities of Practice. In 6th International Conference on Networked Learning, NLC 2008.

[11] Spira, J.B., and Feintuch, J.B. The Cost of Not Paying Attention: How Interruptions Impact Knowledge Worker Productivity. Basex (2005).

[12] Davis, F.D. Perceived usefulness, perceived ease of use, and user acceptance of information technology. MIS Quarterly, Vol. 48, No. 4, (1989), 319-340.

[13] Kuenzel, M., Charlier, B., and Daele, A. Modeling activity and development of communities of practices. In Proc. EIAH (2007).

[14] D.MED.06: Second operational version of CoPe_it!. Public Deliverable, PALETTE project, 2008.

[15] D.MED.08: Analysis and exploration of innovative collaboration means within CoPs. Public Deliverable, PALETTE project, 2008.

APPENDIX A: Standards used

XML

Purpose: The XML description language was used both as an exchange and storage format for data.
Interoperability context: CoPe_it!'s content repository for example stores all objects into its database as XML documents which made it easy to provide repository services to external tools such as DocReuse and Amaya. As an exchange format it greatly simplified the profile synchronization scenario between eLogbook and CoPE_it!. During such synchronization, profile information is represented as XML and exchanged between the two tools.

RDF /SPARQL

Purpose: RDF was used and an exchange format for semantically annotated data. SPARQL was used as to query semantically annotated data.
Interoperability context: RDF and SPARQL both where used in CoPe_it!'s to achieve interoperability with the Knowledge Management tools (SweetWiki) of the project's WP3. In particular RDF and SPARQL gave the opportunity to successfully integrate ontologies that were defined, stored and managed by SweetWiki, into CoPe_it! workspaces. This integration enabled the tagging of items in CoPe_it! workspaces with concepts from ontologies .

Web Services

Purpose: Web Services were used to publish functionalities of the tools which in turn were available for invocation from third party tools.
Interoperability context: CoPe_it! published services related to its content repository and its user management modules as Web Services, which were invoked by tools such as DocReuse and Amaya. Web Services were also used by CoPe_it! to invoke semantic services published by SweetWiki.

Open ID

Purpose: Open ID was used to achieve uniform user authentication across PALETTE tools.
Interoperability context: Each of the three tools, eLogbook, CoPe_it! and SweetWiki required users to open an account, which proved to be a discouraging factor for the use of the tools. In order to overcome such issues, all three tools permitted login of users via their Open IDs.

HTTP

Purpose: HTTP is the primary mechanism with which operations upon resources, managed by the collaboration tools, are requested.
Interoperability context: All collaboration tools put great effort in making the resources they maintain available via proper HTTP methods. The term 'proper' is used to denote the fact, that the efforts concentrated on following precisely the HTTP standard and use the appropriate method in every situation. This led for example in supporting the PUT method to upload resources via Amaya into the content repository of CoPe_it! although it is common practice to use the POST method in these situations.