Strategies and Taxonomy, Tailoring your CSCW Evaluation

Kahina Hamadache , Luigi Lancieri

Orange Labs, 42 rue des Coutures, 14000 Caen, France
{kahina.hamadache, luigi.lancieri}@orange-ftgroup.com

Abstract. With the rapidly growing development of Computer Supported Collaborative Work technologies, the evaluation of these services becomes an essential aspect. This evaluation mixes technical, business, social, perceptive, ergonomic aspects which can't be considered independently. In this paper we propose a new taxonomy of CSCW evaluation methods based on previous works and on our analysis of current evaluation methods. With this new taxonomy and the description of CSCW development process and life-cycle we are able to propose evaluation strategies that can be adapted and tailored for most of systems.

Keywords: Evaluating CSCW, Groupware evaluation, evaluation strategy, evaluation adaptability

1 Introduction

Many different techniques have been used to evaluate groupware technologies, applying approaches that range from engineering to the social kind. Such methodological variety is due to the CSCW field, and up to now no consensus has been reached on which methods are appropriate in which context. Trying to apply conventional evaluation techniques to groupware applications without adapting them can be impossible or lead to dubious results. Applying the method of cognitive walkthroughs to the evaluation groupware, modifying traditional techniques in order to be able to apply them to the study of groupware applications can be complicated and expensive.

Many researchers believe that groupware can only be evaluated by studying real collaborators in their real contexts, a process that tends to be expensive and time consuming.

Others believe that it is more practical to evaluate groupware through usability inspection methods which do not utilize a real work situation. Groupware usability evaluation is difficult to perform because the common tradeoffs provided by different evaluation methods are constrained by the complex multidisciplinary nature of groupware systems. Traditionally accepted methods for assessing usability, such as laboratory experiments and field studies, become increasingly unmanageable because they involve multiple persons, which can be hard to find with the required competencies, may be geographically distributed, or simply unavailable for the

considerable time necessary to accomplish collaborative tasks. Traditional experimental and laboratory methods that remove the software from its context of use may obtain simplistic results that do not generalize well to real world situations.

As an alternative to the laboratory, many groupware researches advocate the use of ethnographic and sociologic methods that explicitly consider culture and context (e.g. Quick and dirty ethnography [1]). These methods have been successfully applied to real situations, but they tend to be expensive and somewhat limited. They demand considerable time and evaluator experience. They work best at the beginning of design to articulate existing work practices and at the end to evaluate how systems already deployed in the work setting are used.

These limitations led to the emergence of a collection of discount methods: Groupware Task analysis (GTA) [2], Collaboration Usability analysis (CUA) [3] and Heuristic Evaluation (HE) [4]. These methods lack the capability to quantitatively predict human performance.

New evaluation strategies are needed that uncover central issues associated with groupware success and failure, and they need to be more flexible than they currently are in order to adapt to a greater range of factors that need to be considered [5].

In this paper, we want to emphasize the necessity of a comprehensive evaluation taxonomy and strategy for applications in CSCW. In this perspective we will propose a taxonomy of evaluation methods helping us to characterize them. Exploring further this idea, we propose to organize evaluation of CSCW systems in Evaluation Strategies that provide the evaluation process for a given system, defining what kind of method to use at what time. With this solution we intend to be able to plan almost any evaluation of CSCW system.

The content of this paper is organized as follows: in section 2, we review the existing taxonomies of evaluation CSCW. In section 3 we present our own taxonomy. Section 4 presents how we can build a strategy and gives an example. Finally we outline some perspectives and discuss of our proposition.

2 Existing Taxonomies of Evaluation Methods

Due to a long history, relatively to computer sciences' one, many evaluation methods have been proposed and designed to evaluate CSCW systems. Despite this profusion of methods it's not always easy to know which method you should use for your evaluation. To solve this issue we need to elaborate evaluation strategies, composed of several methods combination. Elaborating such strategies is complex; it requires knowing what methods have to be employed for a given kind of system. Thus, the first step in this perspective is to construct an accurate taxonomy of evaluation methods, based on several aspects of CSCW. In this section we'll see the already existing taxonomies proposed on the literature. Then we will propose our own taxonomy based on the methods previously mentioned and CSCW features.

As a critical point and a bottleneck in term of CSCW evolutions, the evaluation has been an interesting but also one of the most complex research domain. In order to make it more understandable to mankind, several researchers decided to go above the evaluation domain and tried to organize the work that have been done by others.

Among them, we can cite Randall [6]; they have identified four orthogonal dimensions to classify the kinds of evaluation in groupware:

- Summative X Formative;
- Quantitative X Qualitative;
- Controlled Experiments X Ethnographic Observations;
- Formal and rigorous X Informal and opportunistic.

The authors state that the most used types of evaluations are the summative controlled and experimental (considered a formal technique); and the formative qualitative- opportunistic approaches (considered an informal technique).

Usually a distinction is made between *formative* and s*ummative* evaluation. Formative evaluation is meant to inform designers and developers designing the service or application and getting user feedback about preliminary versions. Summative evaluation is meant to inform the client or the external world about the performance of the service or application in comparison to a situation where there is no such service available, or to a previous version, or to competing services; in brief, to demonstrate the usefulness of the system.

This first taxonomy is relevant as it allows considering any evaluation method. However, for a proper classification, we should be able to consider methods as intervals over the specified dimension. Indeed, as we can vary methods settings, their classification can't be limited to a precise point in Randall's space.

CSCW evaluation is a vast domain, and as such it can be fathomed from many perspectives. In [7] the author proposes to consider it from five aspects as shown on Tab 1.

Ethnography	Qualitative	Psychological	Systems Building	Taking Advice
• Ethno-methodology • Conversational Analysis • Interaction Analysis • Distributed Cognition • Activity Theory • Breakdown Analysis • Others	• Interviews • Questionnaires • Group Discussion	• Lab Experiments • Analytic Approaches • GOMS Approach	• Iterative Prototyping • Participatory design • Beta Testing • Heuristic Evaluation • User Testing • Semi-Situated Ethnography	• Consumer Reports • Consultancy Reports • Marketing literature

Tab 1 Ramage evaluation methods taxonomy

- *Ethnography* is the study of an entire organization in its natural surroundings over a prolonged period of time.
- *Qualitative* methodologies ask people questions about their experiences and compare/ contrast the answers to other people surveyed.
- *Psychological* methods use either lab experiments that focus on the isolation and analysis of a very specific phenomenon or analytic approaches that attempt to describe human interaction using formal models.
- *Systems Building* focuses on the development of partial or complete systems with the goal of improving them based on the evaluation.

- *Taking Advice* uses oral, video, and written information about an application as an evaluation mechanism.

Ramage points out that the very nature of CSCW imposes his taxonomy to be imperfect. As this nature results of the intricate combination of various disciplines, it makes it even more complex to provide a unifying taxonomy of evaluation methods. Also, given the breadth of his taxonomy there is some overlap between the methodologies.

David Pinelle and Carl Gutwin [8] have an approach closer from Randall's as they consider CSCW systems from strictly unrelated aspects. They reviewed forty-five CSCW articles from 1990 to 1998 with the objective to evaluate the use of evaluations methods and their different categories. They classified the evaluations both in relation to the environment where they are accomplished (natural occurrence or simulation of the phenomenon), and the degree of the variables manipulation (rigorous or minimum control of variables). See Tab 2.

		Manipulation	
		Rigorous	***Minimal/None***
Setting	***Naturalistic***	Field Experiment	Field Study Case Study
	Controlled	Laboratory Experiment	Exploratory

Tab 2 Evaluation classification [3]

They report that most of the articles only include laboratory experiment or even no evaluation, only some articles provide an experiment in real settings. We can also point out the fact that the evaluation is not completely integrated into the development process, despite almost every book, article, teacher or expert laud it. Indeed, most of evaluation processes are held for a finished application, package or prototype, it is not yet a continuous task during the development process. Another part of this report states that the most classical mean for evaluation is the observation, with direct sight or videotapes. The second technique is composed of interviews and questionnaires. They point out that evaluations lack of interest for "*organizational work impact*" and that most of them only focus on "*Patterns of system use*", "*Support for specified task*", "*User Interaction through the system*", "*Specific Interface features*" and "*User Satisfaction*". They also point out the fact that the evaluation should include gradually more and more work settings during the development of the software. They stress the fact that it is really important to lead evaluation even at the beginning of the development, it can avoid serious problems or misunderstandings, allowing you to be sure of what you're doing and protect from "chain reactions", meaning that if you have created a part of the application without evaluation and when you finally test it in real conditions, users can tell you that "he didn't want to have a shared file storage, but a personal one", and then you can redevelop most of your application. Furthermore authors suggest that evaluations should be shifted around users and their organizations and those researchers should try to reduce time and cost of evaluations methods, making them more attractive for companies and researchers themselves. Their conclusion is that each work used different approaches, methodologies or techniques for conducting evaluations.

Recently, [9] have led an interesting survey on some evaluation methods. What they suggest is not to use only one evaluation method, but to divide it into three phases: the first one consist in formative lab-based methods which goal is to avoid main errors; the second are field methods where you have to consider users' context; finally the third ones are qualitative methods in real conditions.

In order to facilitate the planning of evaluation Herskovic et al propose a classification of evaluation methods depending on some simple, but still fundamental, characteristics with limited values:

- *People Participation* : can be users, developers, experts or any combination of them;
- *Time to Apply the Method* : the moment when the evaluation takes place (before, during or after the development of the application);
- *Evaluation Type* : describe if the evaluation is qualitative or quantitative;
- *Evaluation Place* : can be a laboratory or usual work place;
- *Time Span* : the time dedicated to the evaluation, it can be hours, days or weeks;
- *Evaluation Goal:* describes the purpose of the evaluation method, what it is aimed to. It can be the evaluation of the product functionality, the collaboration process of the system or the product functionality considering the collaboration context.

In addition to this first classification, Herskovic et al furnishes a second one to estimate the final cost of an evaluation method according to its characteristics. It is another tool facilitating the construction of the triple-phased evaluation process.

This work is particularly relevant as it is based on the analysis of existing methods. This characterization is a good step in the long walk to a better understanding and appreciation of evaluation.

3 Proposed Taxonomy

The first element we consider to build our taxonomy is the fact that it can't be composed of a simple dimension. On the contrary, it should be designed according to a complex space. However, in opposition to Randall, all the dimensions of this space are not mutually orthogonal, implying that some of them can partially overlap themselves.

The second step of this process is the identification of important aspects of evaluation methods. It gives us the list of characteristics that we'll be used in the taxonomy.

The third step consists in defining the "meta" aspects of CSCW evaluation; this step is done by analysing the previous list of characteristics and extracting the main categories.

Fourth, we sort the characteristics according to these categories with the possibility to have a given characteristics in several categories (but obviously not all the characteristics in all the categories).

Fifth, inside the categories we try to gather characteristics by discovering similarities between them and then building sub-categories.

These five steps have led us to the following taxonomy (Tab 3):

- *Development specific aspects:*
  - *Development process:*
    - *Type:*
      - *Iterative;*
      - *Waterfall;*
      - *Extreme Programming;*
      - *...*
    - *Development Step;*
    - *Goal:*
      - *Maintenance;*
      - *New System;*
      - *...*
  - *Final System:*
    - *Goal;*
    - *Scalability;*
    - *End-users type*
      - *Anyone;*
      - *Developers;*
      - *Scientists;*
      - *Government;*
      - *...*
  - *Current System:*
    - *Scalability;*
    - *Step in development process;*
    - *Automation capacity*
    - *Evaluation Cost:*
      - *Computational Cost;*
      - *Human Cost;*
      - *Time Cost;*
    - *Feature to evaluate:*
      - *Feature type;*
      - *Feature maturity;*
    - *Evaluation needs:*
      - *Modus operandi:*
        - *Exploration;*
        - *Evaluate some precise points;*
      - *Evaluators type:*
        - *Experts;*
        - *End-users;*
        - *Developers;*
        - *Diversified;*
        - *Representative Sample.*

- *Method specific aspects:*
  - *Goal:*
    - *Focus:*
      - *Usability;*
      - *Quality;*
      - *Performance;*
      - *Sustainability;*
      - *Utility;*
      - *Coherence;*
      - *Extensibility;*
      - *Scalability;*
      - *...*
  - *Type:*
    - *Formality:*
      - *Formal;*
      - *Informal;*
    - *Business consideration:*
      - *None;*
      - *Weak;*
      - *Average;*
      - *High;*
      - *Full;*
    - *Users multiplicity:*
      - *Single;*
      - *Multiple;*
  - *Cost:*
    - *Time Cost;*
    - *Human Cost;*
    - *Computational Cost;*
  - *Evaluation context aspects:*
    - *Evaluators type:*
      - *Experts;*
      - *End-users;*
      - *Developers;*
      - *Diversified;*
      - *Representative Sample.*
    - *Evaluation Place:*
      - *Laboratory;*
      - *Real location;*
    - *Evaluation Step:*
      - *Preliminary;*
      - *Main;*

• *Collaboration specific aspects:* - *Collaboration Model:* ➢ *Mode:* • *Asynchronous;* • *Synchronous;* • *Mixed* ➢ *Group Structure:* • *Size;* • *Scalability;* • *Members Coupling;* • *Members type:* o *Scientists;* o *Developers;* o *Diversified* o *…* ➢ *Evaluation Cost:* • *Computational Cost;* • *Human Cost;* • *Time Cost;* - *Evaluation focus:* ➢ *Single user behaviour;* ➢ *Multiple user behaviour;* ➢ *Mixed.*	

Tab 3 CSCW Evaluation Taxonomy

The proposed steps for building the taxonomy are not mandatory; their main goal is to provide guidance for us and users in the representation of evaluation context and is largely inspired by classical taxonomy and ontology construction methods such as Bachimont in [12].

Obviously this taxonomy isn't exhaustive; it does not intend to address every methodology with all their details in this presented form. However, it can be simply *extended* to support any new kind of method.

4 Strategies for Evaluation

Evaluation of CSCW project is, of course, very important from a managerial point of view. In the first place, evaluation before a project is started is a key element to decide whether the project is worthwhile. Afterwards, evaluation is also useful as a basis for rewarding participants, to justify financing similar projects, or to justify a second phase of the project [10].

All evaluations have common features: in all cases there is an object being evaluated, a process through which one or more attributes are judged and valued, and all evaluations have a purpose**.**

As we mentioned in the previous section [9] propose an interesting approach to CSCW evaluation by proposing a three-phased strategy. This strategy relies on the principle that you don't need the same method at each step of your development:

1. *Formative lab-based methods* (perform some pre-evaluation to avoid main errors).
2. *Field methods* (with the participation of users associated context).
3. *Qualitative methods in real work settings* (evaluate if it really works).

This work is one of the few we found to propose a real strategy of evaluation. It is even more valuable as it builds a frame for evaluation, meaning that instead of telling which method to use, is only give a more general category. Then you're free to use the best method for your system, picked-up in the right category.

Relying on this good idea, we decided to go deeper in the definition of strategies refining the description of evaluation methods according to the development strategy, processes and steps. Naturally this refinement takes also advantage of our previously presented taxonomy.

Another point has to be noticed before diving into the strategies. We think that the good evaluation of a CSCW system has to be organized in three phases:

1. Evaluate the collaborative aspect;
2. Evaluate the business aspect;
3. Evaluate the combination of collaborative and business aspects.

This separation is particularly relevant as it allows identifying quickly and efficiently lacks in collaborative and business aspects of the system. Thus is can also help finding problems emerging when you integrate collaboration into the business domain.

4.1 Building a strategy

Choosing a specific method instead of another is a critical need. It determines if the evaluation you'll lead is relevant or not for your system.

Deciding and choosing between these methods is puzzling. The methods have their own weaknesses, and trade-offs, they can be complementary or exclusive. Because the methods found overlapping problems, we expect that they can be used in tandem benefiting from each other, e.g., applying the discount methods prior to a field study, with the expectation that the system deployed in the more expansive field study has a better chance of doing well because some pertinent usability problems will have already been addressed.

To be quite exhaustive our model lets you two possibilities: picking up an existing strategy related to an evaluation close to your own; or building your own strategy "from scratch". The first opportunity is only interesting in some rare cases where you really are in the hurry and every hour or even every minute count. We'll get back on this first approach in the discussion, for the moment we focus on the second one.

So, how can we efficiently choose a strategy to evaluate a given system? The main idea is to find the best matching between the definition of your current system and the definition of the context in which strategies take place. For instance, the strategy will not be the same if you are at the beginning or at the end of your development lifecycle. This process is done according to five steps:

1. Describe the context of the evaluation.
2. Define the different phases of your development.
3. Extract the Evaluation Strategy Outline from the development phases.

4. Refine Evaluation Strategy Outline's methods' description.
5. Select Strategy's methods.

The first step to build a strategy is describing the context of the evaluation. In order to do so efficiently and exhaustively we propose to take a top-down approach. By this we intend to start by defining high level categories of the taxonomy and then going deeper and deeper. For instance, one of your first elements to describe is the development process type: do you use a traditional iterative process, a waterfall one or do you prefer the Extreme Programming methodology. Obviously, describing the development process is not sufficient, you also need to specify other system's relative aspects, business specific and collaboration's ones. Indeed, as the evaluation process is split in three different phases: business, collaboration and business + collaboration, we need to describe them sufficiently to find appropriate methods for each of them. Moreover, we believe in the proposition made in [9] to separate the evaluation in three phases: short lab experiment to detect main problems, field method with users' context to evaluate deeper and real settings evaluation to gain qualitative feedbacks.

Once the first description phase is done the second step consist in the description of the different phases of your development. For this step you have to define the different steps needed in your process and define the order in which they appear. For instance you should define the order in which you develop the different features of your system and of what types they are, business specific, collaboration specific or else. This part of the specification is really important as it is the base of the evaluation's outline generation.

Third step is the extraction of the evaluation's outline. As we've just said, this step relies on the description of development phases. Thus, to build the outline, or skeleton, of the evaluation strategy we have to consider each phase of the development process and establish if it requires an evaluation phase, moreover, we have not only to consider the development phase alone but we need to consider it and its position in the whole process of development in order to refine the evaluation methods and correctly establish if additional evaluation phases are needed. For instance, if the precedent phase of development was to create storage module and the current one focus on the development of an event logging feature, we not only have to test the new feature, but also the interaction between the storage and logging parts; for example to see if events are correctly represented in the repository.

The fourth step to build the evaluation strategy consists in refining the evaluation methods types selected to form strategy's skeleton. This part of the building process relies on the skeleton and on the previously description made through the taxonomy. Hence, the previous step gave us a set of evaluation methods types described with some broad criteria related to the development process phases. To complete this we use the description of the evaluation context made in the first step. Thus, we only have to "complete" the description of each method with the evaluation context and then access to a refined description of the evaluation context of each evaluation phase of the skeleton, giving us a refined skeleton.

The last step to build the strategy is the final selection of adequate methods. The main principle of strategy building is to find a matching between the evaluation context and the context in which a method takes place. As the evaluation context is described, we just have to find the corresponding methods. The natural way to perform it is by describing such methods. Hence the "matching" we propose rely on

the “comparison” of system’s context against methods’ intended situation. Thus the description of evaluation methods through the taxonomy is crucial. But it’s also a heavy task requiring a long study of each method. Still, that’s not an unfeasible work as we think the definition of methods can be enriched by all users, closing the loop of collaboration. Besides, the taxonomy approach of our work enables users to only describe some general aspects of methods, resulting in a broader range of selected methods of system evaluation but saving large resources in exchange. By doing so, you load your burden with an extra task: choosing between a set of methods.

Finally, you’ve got an evaluation process consisting in an ordered sequence of evaluation methods: an Evaluation Strategy. Figure 1 sums up how you can build your evaluation strategy.

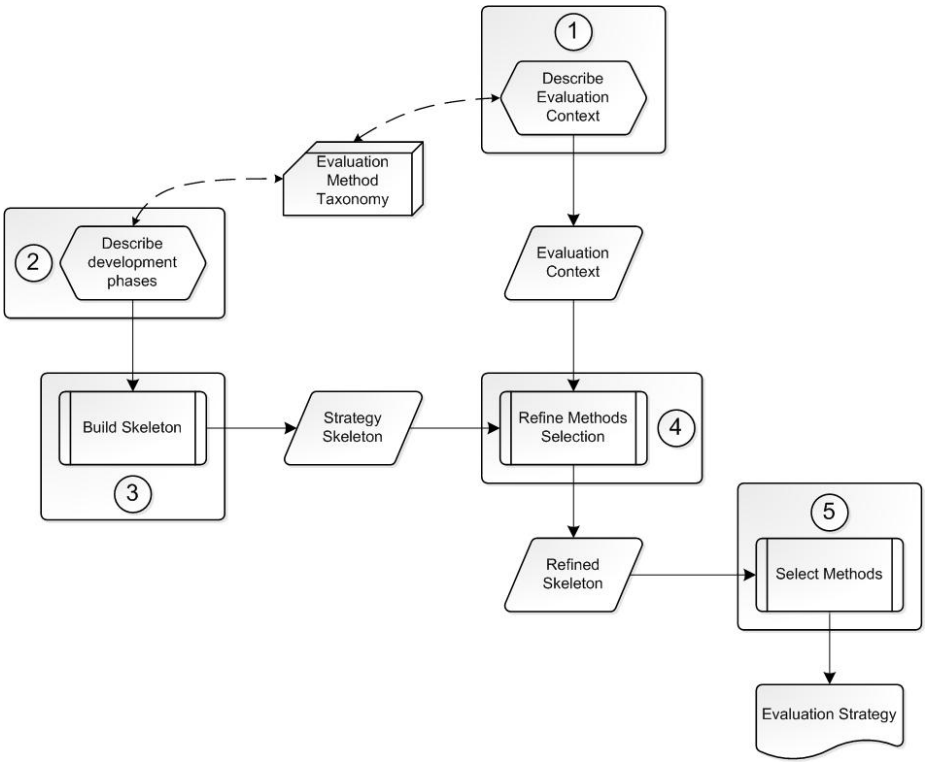

Figure 1 How to build an Evaluation Strategy

To conclude this section we’d like to consider a critical point in the evaluation process which is often a source of conflicts: the lacks of adaptability of evaluation method to the evolution of development process. That is to say, if your development process suddenly speeds up, the heavy evaluation strategy you have chosen may not be able to make it. Thus, we think it is essential for a strategy to be able to be adapted to the changing context. In this perspective, the taxonomy we propose is central. Indeed, it provides a simple tool to find what methods have to be removed and which have to be used instead in your strategy to fit the new evaluation context.

4.2 Example of Evaluation Strategy

To illustrate the use of our approach, we'll now take an example of CSCW system evaluation. Let's consider a service we developed in a previous work [11]. This service is quite simple: it provides the capacity to automatically publish a questionnaire on a dedicated forum-like website. Thus it allows a team of users to efficiently communicate and collaborate by giving them to possibility to send questionnaires, answer to them and have a synthetic view of the responses even if they only have a low-resources device.

• Business aspects:	• Collaboration
- Questionnaire ➢ Editing ➢ Sending ➢ Publishing ➢ Viewing ➢ Commenting ➢ Voting ➢ Synthesising	- Questionnaire ➢ Voting ➢ Commenting - Messaging ➢ Notification of publication ➢ Sending - Role management

Tab 4 Evaluation use case - Business and collaboration aspects

As we'll obviously not write the full specification of the system we'll only focus on main aspects of this development.

Following the five steps we have defined, the first one is the description of evaluation context. In a first time we have to identify business specific aspects and collaboration ones as shown on Tab 4. To complete the description of the context in which this development takes place, let's make a short description of the resources:

• Human	• Time	• Hardware
o 20 Man-day o 4 Peoples	o 2 weeks (firm)	o All necessary

Tab 5 Evaluation use case - Resources

On the previous table (Tab 5) we can see that the evaluation process have to be completed within two weeks. The hardware is not really a problem as required resources are relatively limited. Finally, the team has freed the equivalent of 20man/day to lead the evaluation to be distributed among 4 peoples.

Sticking with the taxonomy we can make the following assumptions:

- The development process relies on a fast iterative method;
- The goal of this process is to create a new feature for an existing system;
- End-users are accustomed to use communication means and web browsers;
- The evaluation have to explore the system in addition to validate the new features and check if it doesn't interfere with the normal behaviour of the system;
- Evaluators of the system need to be end-users for the final part of the evaluation;

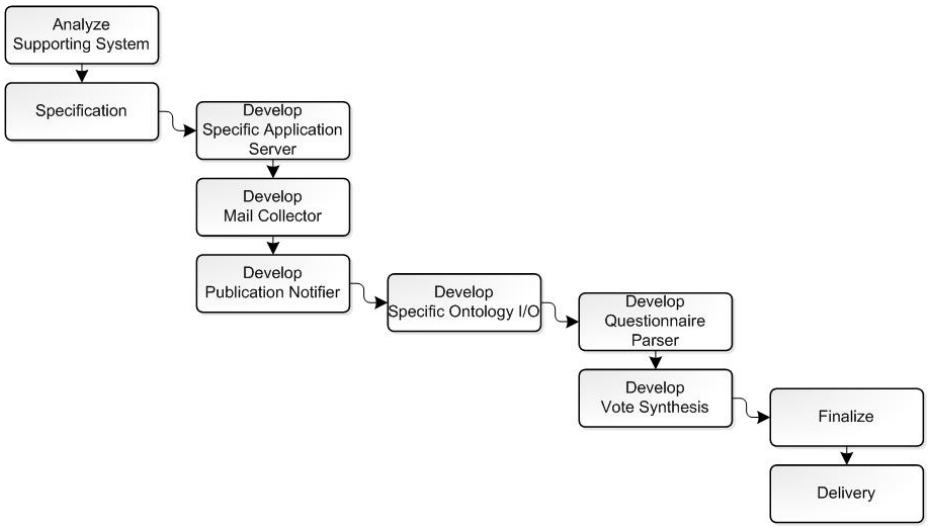

Figure 2 Development Life Cycle

Considering the limited time granted for evaluation but the rather large amount of human and hardware resources, the proposed methods have to be adaptable for larger groups with intensive evaluation instead of small groups or loose evaluation sessions. Moreover, to be loyal with Herskovic's proposal ([9]) each evaluation step is subdivided into three phases, but in the case of iterative development process and even more when the development has to be fast, each loop on a same feature reduces the evaluation time and especially the time for lab experiment.

Secondly, we had to define the development lifecycle we use. Figure 2 shows how this development was organized: Analysis of supporting system in order to know if it was able to correctly support the new features, specification of the new features, multiple development phases, finalization and then delivery.

Based on this development life cycle we can extract the outline of the evaluation strategy. As we can see on Figure 3, to extract this skeleton we start by considering each phase that has been described earlier, for each one of them we figure out if it requires one or several evaluation stages. For instance in our example the "Analyze Supporting System" phase requires to evaluate if the system is suitable for the desired evolutions. As a direct consequence we deduce that we need three "Integration Feasibility Methods", one to evaluate if the system can handle the collaborative aspect, one to know if it correctly sustains the business part of the new features and finally a method to evaluate how the combination of these two aspects interacts with the existing system.

Fourth step of strategy building we have to refine the description made in the previous step with the help of the evaluation context defined in the first step. As it would be a little too long to describe all the methods evoked in Figure 3, we'll only refine one of the stipulated method. In order to have a relevant example, we consider the last evaluation step: "Full System Evaluation Method".

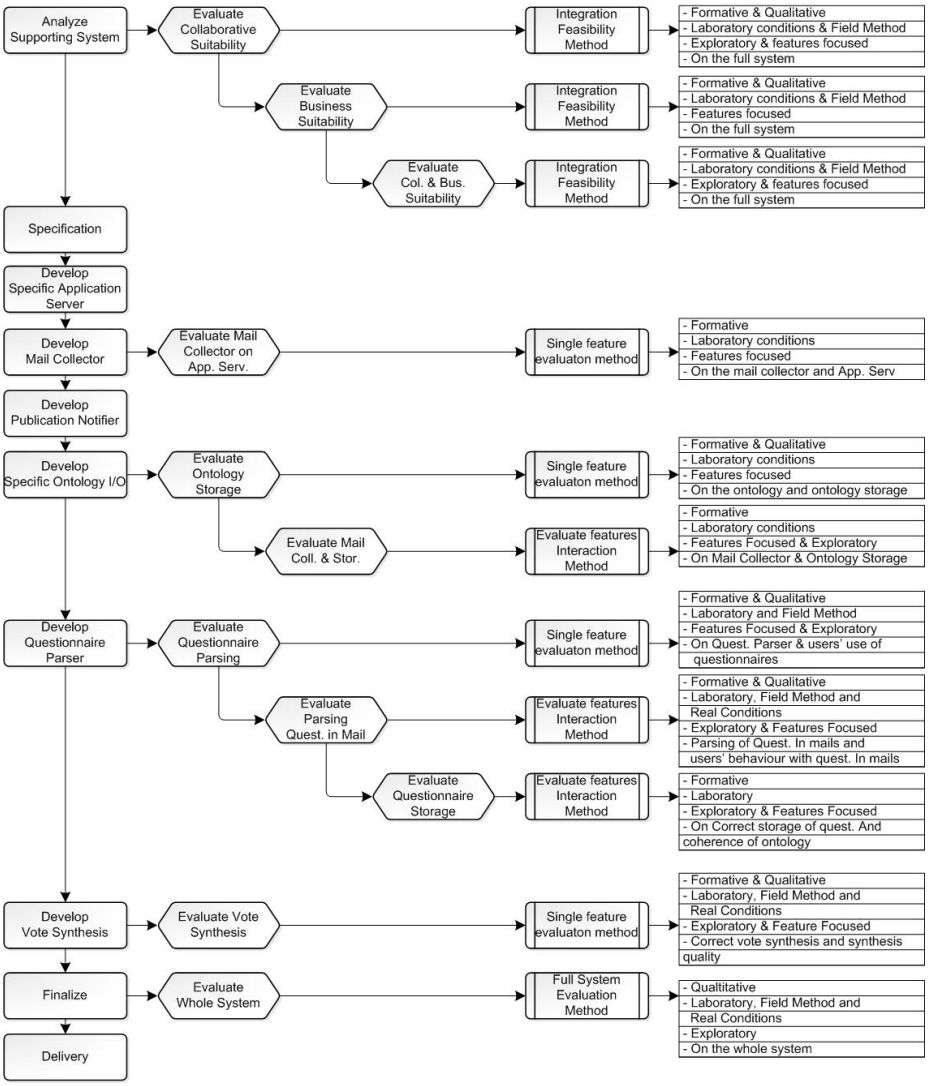

Figure 3 Use case - Strategy Outline

Looking at the associated description we are able to see that this step has the following requirements: *Qualitative evaluation*, *Triple-phased* (laboratory, field study and Real Conditions)*, Exploratory* (as all features has been tested previously we only have to figure out the overall quality of the system in its wholeness and if some unexpected challenges are leveraged*)* and obviously the evaluation concerns the *Full System*. Considering the evaluation context we know the selected method have to be rather fast than exhaustive as the evaluation time is short. Besides, we need to make end-users participate to the evaluation in order to have a real qualitative feedback but also to be able to efficiently explore the system with users' habits. Considering these requirements we can refine the method to the following ones: 1- Users' Exploration

(let users use the new features without guidelines, just with the instructions to know how to use it and let them explore the system); 2- Scenario Based Evaluation (write scenarios to guide users in there walkthrough); and 3- Scenarios Refining Method (starting from some pre-written scenarios, users have to collaborate with peoples in charge or the development to refine scenarios through their own experience and desire). As we design it, to complete this step of strategy building you have to refine all the methods of your strategy skeleton.

The last step of this process is the final selection of methods. It has to be said that this step is not mandatory for all the evaluation steps of the skeleton. Indeed, for some of these steps you can have found only one method matching their evaluation context. In that case you obviously don't have to make a choice. Nevertheless, given the refinement level of an evaluation step (have it to desired or not), it can be matched by several methods. Finally, with all the required selections done, we've got our Evaluation Strategy.

5 Discussion

In this paper we tried to consider the barely dusted field of CSCW evaluation building. As we have seen in the past sections there have been several attempts to construct and propose a taxonomy that could help representing evaluation methods. If we refer to Randall [6] we can clearly say that even if this classification is relevant, it cannot efficiently help a user choose one method or another, it can surely give a trail, but no more. With more advanced taxonomies such as the one proposed by Pinelle and Gutwin in [8] users can find better ways to consider the evaluation they have to perform. In a different perspective, Herkovic in [9] has extended the representation of evaluation to several aspects leading to a finer representation. In addition they propose to organize evaluation in three steps. These papers are interesting and our work has roots in them, however there are some shortcomings: the granularity of the representation of evaluation is not sufficient to find a determine a precise method; another lack is the point of view of some taxonomies: most of them don't consider evaluation from the system point of view but only from "evaluation process" point of view. Our thought is that to consider evaluation properly you should see it from the system point of view, what you want to evaluate but also from the evaluation method point of view, how you have to evaluate.

We proposed a taxonomy whose goal is to provide a base to represent CSCW systems evaluation context. By a fine comparison of evaluation methods representation against evaluation context we can select relevant methods to be used. Moreover, following the development process of the system and keeping in mind the final result we are able to propose a complete evaluation strategy, indicating which method have to be used at what time of which step.

As we mentioned earlier, our approach offers the possibility for users to inspire themselves on existing strategies. This capacity is especially relevant in the case when people don't have enough resources to build their own strategy or when they need a strategy for a brand new system that doesn't match existing ones. In this last case, despite the fact that our approach should be able to propose a least a set of strategies,

it can be interesting and relevant for the people in charge of the evaluation of this system to have a look at evaluation strategies related to closest already evaluated systems and maybe use one of them. In this kind of cases, where the taxonomy isn't sufficient because some specific parts of the system are not yet correctly represented, it is important that users can be allowed to complete the taxonomy by describing the missing parts. Besides this description, the feedbacks on the chosen evaluation strategy for this new system are even more significant and valuable.

Another point on which we want to insist is the possibility for users to choose an evaluation strategy and use some extra evaluation methods they think relevant. Once again, given the feedbacks of this extension, strategies can be updated and improved.

From the last points evoked we have to point out an emerging necessity of our system: the need for reasoning over the strategies. Thus, even if "Taxonomy" and "Ontology" are closely related parents, our model tends to become more "intelligent" and then our taxonomy tends to evolve to an ontology. Given that, we estimate than in a close future we'll be led to consider the design of inference rules, allowing us to naturally handle changes in our former taxonomy.

Continuing in the same prospect, our model will doubtlessly grow into some kind of expert system dedicated to the recommendation and knowledge representation and capitalization of CSCW systems evaluation. With further researches this systems should be able take into account feedbacks of evaluations to enhance and refine strategies.

Beyond these considerations, the very nature of our approach is based on the specification of the evaluated system. This particularity implies, once your specifications are finished, that you should have both your development design and your evaluation strategy. Furthermore, as evaluation can take place even at the beginning of the specification, the taxonomy we propose can help you lead some "pre-specification" evaluation.

Following the same perspective, the inherent flexibility of our model ensures you to be able to adapt your strategy according to the fluctuating requirements of the development and available resources.

Last but not least, the flexibility of our model provides an unusual but helpful advantage: the scalability in term of user's evaluation experience. Thus, our approach is suited for users with few or no experience in evaluation; in that case they can simply take one of the proposed strategies and apply it. But the model can also be used by expert in evaluation who will be able to rely on the obtained strategies and customize them the way they want. Such experts, via the feedbacks they can provide to our system, will confer it a part of their skills.

This paper doesn't pretend to solve all problems raised by the evaluation of CSCW systems. However, we think our contribution to this field of research can open a new perspective in the understanding and the way we consider the evaluation of complex systems. By rationalizing the evaluation and taking into account the limitations of available resources we tend to bring the evaluation processes more attractive and valuable for users. Besides, the foreseen automation of strategies evolution and recommendation, made through an expert system, will facilitate the job of many systems builders. Finally, defining evaluation strategies from the earliest steps of the development and by paradoxically keeping the possibility to make it evolve until the end of this process provides the flexibility that lacks in most of classical methods.

References

1. Hughes, J., King, V., Rodden, T., Andersen, H.: Moving out from Control Room: Ethnography in System Design. In: CSCW'94, pp. 429-439, ACM Press, New York. (1994)
2. Van der Veer, G., Van Wellie, M.: Task Based Groupware Design: Putting Theory into Practice. In: Proceedings of the conference on Designing Interactive Systems, pp.326-337. (2000)
3. Pinelle, D., Gutwin, C., Greenberg, A.: Task Analysis for Groupware Usability Evaluation: Modelling Shared-Workspace Tasks with the Mechanics of Collaboration. In: Transactions on Computer Human Interaction, pp.281-311. (2003)
4. Baker, K., Greenberg, S., Gutwin, C.: Empirical Development of a Heuristic Evaluation Methodology for Shared Workspace Groupware. In: CSCW'02, pp.96-105. (2002)
5. Neale, D. C., Carroll, J.M., Rosson, M. B.: Evaluating Computer-Supported Cooperative Work: Models and Frameworks. In: CSCW'04, Chicago, Illinois, USA. (2004)
6. Randall, D., Twidale, M., Bentley, R. In: Dealing with Uncertainly-Perspectives on the Evaluation Process. In: Thomas, P.J. (ed), CSCW Requirements and Evaluation, Springer. (1996)
7. Ramage, M.: How to Evaluate Cooperative Systems. In: Computing Department, University of Lancaster, United Kingdom, Ph.D Thesis. (1999)
8. Pinelle, D., Gutwin, C.: A Review of Groupware Evaluations. In: Proceedings of Ninth IEEE WETICE'00 Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises, Gaithersburg, Maryland. (2000)
9. Herskovic, V., Pino, J.A., Ochoa, S.F., Antunes, P.: Evaluation Methods for Groupware Systems. In: J.M. Haake, S.F. Ochoa, and A. Cechich (Eds.): CRIWG 2007, LNCS 4715, pp. 328–336. © Springer-Verlag Berlin Heidelberg. (2007)
10. Baeza-Yates, R., Pino, J.: Towards Formal Evaluation of Collaborative Work and its Application to Information Retrieval. (2006)
11. Hamadache, K., Manson, P., Lancieri, L.: Pervasive services, Brainstorming in situation of Mobility. Alexandria, Egypt. (2008)
12. Bachimont, B.: Engagement sémantique et engagement ontologique : conception et réalisation d'ontologies en Ingénierie des connaissances. In J. Charlet, M. Zacklad, G. Kassel & D. Bourigault (Eds.), Ingénierie des connaissances, évolutions récentes et nouveaux défis. Paris: Eyrolles. (French) (2000).