Improving the testbed development process in collaboratories

de Moor, A.

Published in:
Proceedings of the 12th International Conference on Conceptual Structures

Publication date:
2004

Improving the Testbed Development Process in Collaboratories

Aldo de Moor

Infolab, Dept. of Information Systems and Management
Tilburg University, the Netherlands
ademoor@uvt.nl**

Abstract. Collaboratories are increasingly important as instruments for distributed work. They are highly complex socio-technical systems, in which often advanced ICTs need to be carefully tailored to subtle work practices and organizational structures. However, despite their importance and potential impact, not many successful examples of collaboratories exist. One key obstacle is the complexity of the testbed development process in which the collaboratory is to evolve. In this paper, we propose a method for collaboratory improvement. We show how conceptual graph theory can be used to help improve the testbed development process.

1 Introduction

The scientific research community is one of the creators and oldest users of the Internet. Already when the first computer networks became operational at the end of the sixties, researchers started using their potential for collaboration. Although originally intended for the mere exchange of files, enthusiastic users immediately invented e-mail, one of the current killer applications. Internet-based technologies have developed at a phenomenal rate, both in reach and range. The privileged happy few from the early days have become a worldwide population of hundreds of millions of active users. Moreover, the primitive file exchange technologies from the beginning have evolved into a huge toolbox of functionalities. Much attention has been paid to the needs of individual users: e-mail, information retrieval applications like search engines, office tools, and so on. However, collaborative applications are still underdeveloped. Despite the billions of dollars poured into research into computer supported cooperative work, most collaborators still use powerful but primitive technologies such as mailing lists. Although these tools have proven to be very successful in bringing people together into virtual communities, they have many limitations such as information overload, navigation problems, primitive workflow management capabilities, and lack of customization. This negatively affects motivation and the accomplishment of joint objectives.

One major cause of problems is that the usefulness of knowledge tools is not rigorously evaluated [20]. The scientific community being global and therefore to a large

** Proc. of the 12th International Conference on Conceptual Structures: Conceptual Structures at Work (ICCS 2004), Huntsville, Alabama, USA July 19-23, 2004, Lecture Notes in Computer Science, Vol. 3127, Springer, Berlin, pp.261-274

extent virtual in nature, and having a strong drive to collaborate, is among the first to have become aware of the need for systematic support for the evolution of its socio-technical systems. It is no longer enough to just offer a toolbox with many hammers and nails, and then to wait for a virtual house in which to collaborate to construct itself. Instead, perspectives, methods, and techniques need to be developed in which such socio-technical systems are developed more effectively and efficiently. This takes place in collaboratories.

A collaboratory consists of ”various tools and technologies ... integrated to provide an environment that enables scientists to make more efficient use of resources wherever they are located [19]”. Since research collaboration is highly complex, constantly changing, and in need of many sophisticated ICTs, such integration of tools into a complete environment is not trivial. In collaboratories, a structured and fine-tuned testbed development process should therefore be adopted in which guided experiments with various technologies efficiently lead to more effective community information systems. However, what properties such an evolutionary process should have, let alone how it is itself to be supported by information systems is still unclear. In earlier work, we outlined parts of the solution for collaboratory process improvement. In [6], we described how Conceptual Graphs could be used to improve the pragmatic inquiry process needed for more focused testbed development. In [5], we explained how formally modelling tool contexts could be helpful to this purpose as well. Using this previous work as a starting point, our purpose in this paper is twofold: (1) to explicitly model the collaboratory testbed development process using conceptual graphs and (2) to use this formalization to improve this process. Conceptual graphs are the formalism of choice since they are very useful in pattern matching, which we will show to be essential in this process.

In Sect. 2, we show how collaboratories can be viewed as evolving socio-technical systems. Sect. 3 studies some knowledge structures useful for the modelling of collaboratory improvement. In Sect. 4, we present our method for collaboratory improvement. We end the paper with conclusions.

2 Collaboratories: Evolving Socio-Technical Systems

Already from the beginning, collaboratory research has focused on socio-technical systems. For example, the definition adopted by the National Science Foundation in the early years of collaboratory research was: ” [A collaboratory consists of] various tools and technologies [...] integrated to provide an environment that enables scientists to make more efficient use of resources wherever they are located” [19]. However, the insight is gaining ground that collaboratories are especially entities that support rich and recurring interaction around a common research focus, the critical element for their success thus being the opportunities they allow for encounters, discussions, and the sharing of ideas [10]. A very important question therefore is ”how to support communication that permits human cooperation even when the evolutionary social mechanisms that depend on proximity are absent [15].”

2.1 The Testbed Development Process

Despite their great potential, collaboratories have not grown as much as originally envisioned. This is not so much a failure of the original vision, but a consequence of the great difficulty of supporting complex group work in virtual settings [10]. Successful collaboratory development requires (1) a system architecture and integration to explore ways that people and machines can use component technologies most effectively, (2) a research program to study the conditions required for collaboration, and (3) user-oriented rapid-prototyping testbeds, to understand the impact of the technologies used [19]. Already in the very first report on collaboratories, the essential role of testbed development for collaboratory construction in various scientific disciplines was stressed [17]. Testbeds must be established in actual working contexts, where the required tools for data access and communication can be designed, developed, and tested within a program of prototyping, testing, and evaluation that can support continuous development. Mary Keeler's PORT (**P**eirce **O**nline **R**esource **T**estbeds) project is an important initiative, which aims to produce an integrated context view of such development, using a collaboratory on the interpretation of Charles Peirce's manuscripts as an example [13].

To conceptualize an approach for formally modelling and supporting the testbed improvement process, we build on two streams of thought: Douglas Engelbart's ideas on improving the process of co-evolution of the social and the technical systems, and socio-technical interaction networks as a way of formally modelling improvement patterns.

Engelbart has spent his life studying how to improve systems development by long-term, pragmatically guided whole-system evolution. With today's dazzling rate of technological progress, tool functionalities are developing faster than man's ability to use them to the fullest. Engelbart's mission is to find ways to accelerate our intellectual development, so that we can keep up with our tools and improve their co-evolution with work practices [12]. In his view, one is not just to focus on the co-evolution of Human and Tool-systems, but also to continuously improve the design process itself [8]. In his CODIAK (**CO**ncurrent **D**evelopment, **I**ntegration and **A**pplication of **K**nowledge) process, he describes a vision for increasing the capabilities of organizations to "improve their improvement" [9]. It consists of intelligence collection (to identify problems, needs, and opportunities), dialog records (to conduct and coordinate improvement dialogs), and knowledge products (that capture relevant improvement project status and plans). However, these artefacts are not an end, but only a means to improve improvement capabilities and processes. This is in line with modern views on knowledge management, which claim that tacit and explicit knowledge both have specific, inter-related roles to play in optimizing knowledge creation processes [18].

A good way to capture the structure and behaviour of collaboratories, is to see them through the lens of socio-technical interaction network (STIN) theory, which is rooted in actor-network theory [14]. Using the theory to do careful empirical analyses of work situations, it was found that almost identical technologies are often configured very differently in practice. The theory acknowledges that collaboratories can be seen as layered systems of technical and social components. However, it takes issue with the fact that in traditional socio-technical systems analysis the technology often predominates. In contrast, STIN analysis has a more integrated view of the interaction between humans and technologies. Technologies are not a given, to which the social system has

to adapt. Rather, there are many social characteristics that help shape the technologies, but are also being formed by them. STIN models take into account such issues as actor relations, content control, resource dependencies, work to make the system useful and sustainable, translations to mobilize resources, and business model and governance structures. The models can be used to analyze the complex interactions between people, between people and technologies, and between technologies themselves. Social analysis is required in all stages of the collaboratory lifecycle, including planning, development, configuration, use, and evolution. These forms of analysis acknowledge that 'users' are too shallow a construct for obtaining useful insight in collaboratory improvement. Instead, the models provide socially richer characterizations of people working and communicating in complex, overlapping socio-technical networks.

Combining Engelbart's views on testbed development improvement with the basic ideas behind socio-technical interaction networks seems a feasible approach toward modelling testbed development improvement. However, both approaches are still qualitative, informal and disjoint. Collaboratory improvement, with all its inherent operational and evolutionary complexities, could benefit greatly from a - partially - formal approach. In this way, more systematic analysis methods and supporting tools can be developed. In the remainder of this article, we will outline the foundations of how such an approach could look like, using conceptual graphs as the knowledge representation and reasoning formalism. First, we introduce some relevant knowledge structures.

3 Knowledge Structures for Collaboratory Improvement

Based on Engelbart's views, we distinguish several systems providing contexts of interpretation of tools. Next, we acknowledge the need for improvement patterns, and analyze one promising direction: a socio-technical pattern language. We then explain why conceptual graphs are our knowledge formalism of choice.

3.1 Tool Contexts in Testbed Development

As a starting point for our approach, we use Engelbart's insights into co-evolution improvement to create a layered tool context model. Focal constructs in these contexts are *processes*, since obtaining better quality testbed dynamics is at the core of what is needed. Based on Engelbart, we distinguish four layers of testbed processes:

- *Information/Communication (I/C) processes*: the processes enabled by the tools.
- *Workflows*: the processes in which tasks are executed and coordinated so that the goals of the community can be accomplished.
- *Design processes*: the processes in which the users reflect on their work and I/C processes and propose modifications to the design of their socio-technical system so that workflows can become more effective and efficient.
- *Improvement processes*: the processes in which the design processes themselves are made more effective and efficient.

These processes are organized in four nested systems: the information system, work system, design system, and improvement system respectively. Each higher-order system

provides a context for the system it embeds. For example, the information system is embedded by the work system, which thus defines the context of *use* of the information system. Similarly, the design system provides the context of *change* of the work system, and the improvement system the context of *optimization* of the design system. Collaboratory improvement can only occur systematically if all systems and their interfaces are clearly defined at the level of detail required by the particular community. The advantage of modelling collaboratory improvement as a set of embedded systems is that it allows for the abstracting of details less relevant to a particular improvement purpose. For instance, sometimes the improvement may focus on the information system, when a new tool needs to be evaluated. In other cases, the community may not be interested so much in the particular details of its current socio-technical system, but want to focus on how it can improve its evolutionary practices. Using this contextual framework, testbed development methods can be more systematically created, interpreted, and changed.

3.2 Improvement Patterns

Using the tool context model allows us to conceptually distinguish between the layered elements of improvement knowledge. However, in order to reason about properties of this knowledge, such as about *knowledge gaps*, we need to use modelling approaches that are well suited to the particulars of this kind of knowledge. As the theory and findings on socio-technical interaction networks show, socio-technical systems knowledge evolves in *fragmented*, partial ways. Furthermore, testbed development knowledge often has different degrees of *specificity*: at the beginning of a project, some aspects, such as which particular workflows to support by what tools can remain undefined, while some other issues, as who to involve in a particular design process may need to be very precisely defined. How to combine this fragmented and diffuse knowledge evolution process with the systemic view proposed by Engelbart?

Patterns are good ways to capture such knowledge and place it in a system context. A pattern is something designed or used as a model for making things (Merriam-Webster). The following statements more precisely indicate the power of patterns:

> A pattern is a careful description of a perennial solution to a recurring problem within a building context, describing one of the configurations which brings life to a building.
>
> A pattern language is a network of patterns that call upon one another. Patterns help us remember insights and knowledge about design and can be used in combination to create solutions. (both quotes from Alexander et al, 1977, in [22]).

Humans are very good at using patterns to order the world and make sense of things in complex situations [16]. In the information systems literature, patterns are gaining prominence as a way to deal with the complexity and dynamics of the real world. For example, workflow patterns can be used to develop ideas on how to implement certain business requirements given that a workflow server already exists. Others focus on developing pattern languages that capture communication knowledge of large, distributed

communities [22]. In collaboratory evolution, a structured representation facility for efficiently reporting, tracking, and mapping advancements within projects is essential, making it possible to compare patterns of development and trace similarities and differences among project technical requirements [13].

Thomas et al. give a good description of the role that patterns can play in socio-technical systems improvement, and propose a socio-technical pattern language to make the software development cycle more effective and efficient [24]. In outlining the elements and use of this language, they pose three important questions: (1) how to guide pattern authors to help produce clear, understandable and helpful patterns? (2) how to support users to explore a design space using patterns? (3) what is the relation between a pattern and software components?

Important as an informal pattern language may be, it is not enough to ensure effective and efficient pattern use. Improvement patterns outline dependencies between events in the collaboratory and actions to be taken, by the system or the users. Often, human beings will need to be the agents to signal the need for change, but computers can help monitoring and managing the complex chain of change actions required. Even in small examples like the case mentioned next, people are already at a loss of doing so, as many of the case participants have reported. Thus, the informal pattern language helps to identify what needs to be done when, but in addition some (semi)-formal knowledge representation and reasoning method can be helpful to activate that knowledge. For this purpose, conceptual graphs are our formalism of choice.

3.3 Using Conceptual Graphs

Collaboratories are complex and dynamic networks. Important properties of networked societies are that they are loosely organized, their boundaries are permeable, they are often imperfectly integrated, have (reconfigurable) nodes that may be part of other networks, and have flat and recursive hierarchies [14, 25]. Semantic networks are useful forms of network knowledge representation, as they use links to both record facts and to provide associative access paths, by which facts can be accessed from each other. These paths are then the basis for efficient reasoning algorithms [27]. In fact, the ability to represent and use these links is essential in defining the knowledge of a collaborative community [2].

Conceptual graphs are a flexible and extensible method for knowledge representation. Conceptual graphs are particularly useful forms of semantic networks, as they also include generalization hierarchies (of types, relations, and complete graphs), with a set of powerful operations that make use of properties of the hierarchies of graphs and their components. Conceptual graphs not only allow the description of complex domain knowledge, but also to validate that knowledge against meta-knowledge about the domain. Thus, they are very well suited to represent and reason about the context of pattern knowledge, which, as we argue, is a prerequisite in collaboratory modelling.

4 Towards a Method for Collaboratory Improvement

The previous section outlined the main knowledge structures needed to conceptualize collaboratory improvement, and presented a context- and pattern-based view on the

testbed development process. The current section describes the method that formalizes the *process* in which these knowledge structures are put to use. The method uses conceptual graphs as a formal backbone to initialize focused conversations for specification. To illustrate the method, we first introduce a case. We then examine the role of conversations for specification in collaboratories. To better support these conversations, a collaboratory improvement ontology is introduced, which forms the conceptual basis for our architecture of collaboratory improvement systems.

4.1 Case: the Tools-Yak Community

Blue Oxen Associates[1] aims to develop the art and science of collaboratories and in this way to contribute to the common good. One of its initiatives is to develop collaboratories on collaboratories. To this purpose, it hosts several mailing lists, one of them being the Tools-Yak list[2]. By discussing the ways tools could and should be used in collaboratories, and then implementing and testing proposed solutions, the community formed around this list aims to find principles and practices that contribute to better socio-technical systems for collaboratories. The list has been operational since December 2002. The list has attracted and sustained high-quality discussions, resulting in several interesting experiments and evaluations. As such, it seems a good candidate to illustrate some of the ideas proposed in this paper.

To get an idea of the complexity of collaboratory conversations, we show some illustrative data. For the example, we analyzed the mailing list archive of the first half year of the Tools-Yak community. As in this paper we cannot go into depth, we give an example of only one particular tool being examined in the community: Purple Numbers. One problem with Web pages is that it is hard to refer to specific items in a page. This makes linking often too coarse-grained. To increase the granularity of web links, software has been developed to automatically add 'purple numbers' to each paragraph in a web page. Instead of just linking to a page and then making cumbersome and possibly erroneous references to the "second section, first line", one can now include a link to the specific paragraph of a Web document. It is a simple (technical) idea, but turns out to have important socio-technical ramifications. This is illustrated by the following key indicators:

From December 2002 - May 2003, 95 mails were exchanged on the implementation or use of purple numbers. 17 people were involved in those discussions, with an average of 5.6 mails per person, and a standard deviation of 6.0. In this period, purple numbers were mentioned in no less than 23 threads. The average length of those parts of the threads in which purple numbers were mentioned, was 4.1, with a standard deviation of 6.6. There were several interesting outliers: one person, a coordinator of the community, contributed 24 messages. Furthermore, one thread, on the role of purple numbers in e-mail was extremely long: 31 messages.

Quantitative data on collaboratory improvement are still scarce in the literature. Although benchmarks are lacking to fully interpret their meaning, the presented indicators may help to appreciate the complexity of collaboratory development. They are typical

[1] http://www.blueoxen.org
[2] http://collab.blueoxen.net/forums/tools-yak/

of successful virtual communities, in which there is broad participation, some of it initiated by a facilitator, and in which a wide range of topics is discussed, sometimes leading to passionate debate. However, in Tools-Yak, many community members have voiced the concern that the quality of the collaboratory improvement *content* is excellent, but that many good ideas are lost because of their fragmentation across threads and lack of follow-up. Thus, much is to be gained by improving the quality of the collaboratory improvement *process*.

4.2 Conversations for Specification

The lifeblood of any community, collaboratories not excluded, are conversations. There are many types of work-related conversations, one of which is the conversation for action, in which the goal is to coordinate explicit cooperative action [26]. Another major class of conversations in collaboratories are *conversations for specification*. We define such a conversation as a self-contained unit of communication to accomplish certain specification objectives, like the specification of an experiment for testing a tool in its context of use. Evidence for the effectiveness of predefined conversation models is ambiguous, however [1]. We therefore require a conversation to be only partially structured in the sense that only main specification process entities need to be defined. However, the format of the utterance acts in which these definitions are made can remain free, as is the case in e-mail conversations. We propose that such partially-formalized conversations are crucial in providing adequate support for collaboratory improvement.

There is great value in free-form e-mail message exchange, as it does not constrain users in artificial formats which may have little meaning to them. However, e-mail does not support productive conversations per se [21]. Its free form is not only its strength, but also its weakness. Most explicit dependencies in e-mail are chronological: somebody sends an e-mail with a question or an idea, one or more people respond, those replies themselves attract new replies, and so on. Mostly, e-mail discussions on a particular topic are prolonged, divergent, and repetitive. Such problems lead to many process inefficiencies, as has been studied extensively in, for example, the literature on the IBIS (issue-based information systems) paradigm [3]. We conjecture that such conversation process inefficiencies, not lack of motivation, may be one of the most important reasons that successful collaboratories are so few and far between.

The question now is: can we reduce conversation inefficiencies using some form of (conceptual graph-based) formalization, without losing the strengths of informal (e-mail-based) conversation? In other words, can formalization, used wisely, contribute to collaboratory improvement? The answer is to be found in defining a practical form of *incremental formalization*. This enables users to choose when and how to add finer-grained, computer-readable codification to informal content [23]. In this process, boundary objects are defined: objects which are both plastic enough to adapt to local needs and the constraints of several parties employing them, yet robust enough to to maintain a common identity across sites (Star and Greisemer in [23]). These formal boundary objects form a process space that *circum*scribes rather than exactly prescribes all aspects of collaborative work [11]. As such, this way of thinking fits very well with the pattern-oriented design philosophy.

4.3 A Collaboratory Improvement Ontology

The core of the formal part of our architecture is a collaboratory improvement ontology. The current ontology is by no means complete, but forms a sufficient root hierarchy to be further refined and extended in future collaboratory research efforts.

At the heart of collaboratory improvement is the process of pragmatic inquiry, as described in [6]. Here, the improvement process is seen as a continuous process of hypothesis testing on the role that tools should play in the collaboratory. Proposed hypotheses on, for example, which tools to use to support a particular workflow, can be implemented and tested, and their usage evaluated based on certain community-defined criteria, such as security and userfriendliness. After evaluation, a hypothesis is labelled either as failed or succeeded. The socio-technical system itself can be defined in many different constructs, three of which are named here: *element*-definitions describe parts of the various (information, work, design, and improvement) subsystems making up the total socio-technical system. *Mappings* connect elements from these subsystems. One example of a mapping is a support definition, which links tools in the information system to workflow definitions in the work system. *Socio-technical (system) patterns*, finally, can be constructed of any (cross-sections) of the other elements. For example, "Who Speaks for Wolf" is a powerful socio-technical pattern aimed at engaging all the stakeholders in a design discussion, also those who are absent at a meeting[24]. The focus of this particular pattern is on roles and processes in the design system: make sure to involve all end users in changes to their socio-technical system, etc. However, it also optimizes this process in the improvement system, specifying, for example that *bad* designs need to be weeded out, that stakeholders should be involved in the design process *early*, etc. Such *qualifications* of the design process should be modelled in the highest-level improvement system. The type hierarchy of the ontology is given next.

```
T >            T >              T >
  Criterion >         Process >          System >
  Definition >          Design_Proc        Des_Sys
    Element             I/C_Proc           Impr_Sys
    Mapping >           Impr_Proc >        Info_Sys
       Support            Propose_Hyp      Work_Sys
    STS_Pattern           Test_Hyp       Tool >
  Hypothesis >          Workflow >         Mailing_List
    Prop_Hyp            Archive            Wiki
    Tested_Hyp >        Discuss            ...
      Failed_Hyp        ...
      Succ_Hyp
```

Note that many of these concepts have proper type definition graphs. These can be used to enforce required semantics, but have been omitted for lack of space. Furthermore, this ontology contains some domain-specific concept subtypes, such as Mailing List and Wiki-Tools, and Discuss and Archive-Workflows. In other cases, these specific types could be different.

4.4 An Architecture for Collaboratory Improvement Systems

Fig. 1 outlines the architecture of the system we propose to be used in support of the collaboratory improvement process. It is based on Dan Corbett's important vision of

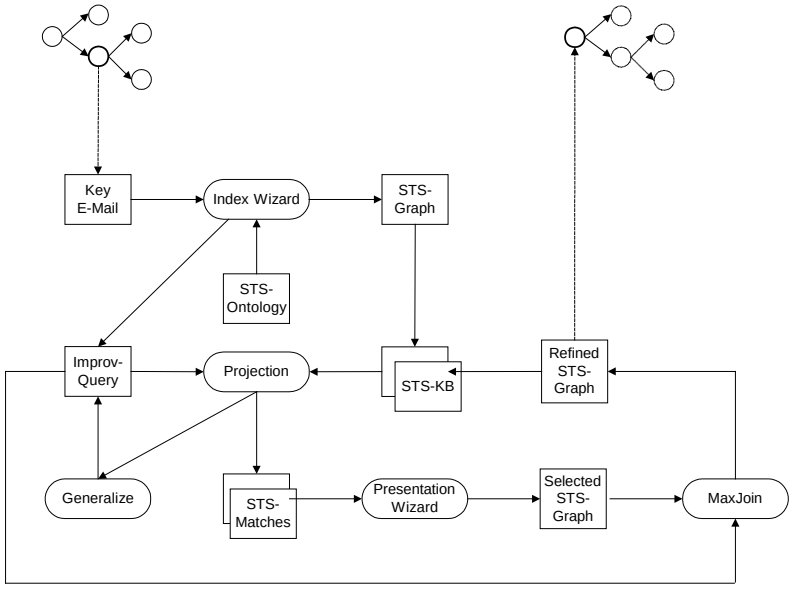

Fig. 1. An Architecture for Collaboratory Improvement Support

a Knowledge Conjunction Toolbox. This vision allows knowledge to be continuously refined and extended, crucial in any evolving design situation. In Corbett's approach, user-drawn graphs are checked for canonicity by the system. This canonical graph can then be used as an index to search the knowledge base for matching graphs. If a match is found, the user can extend his graph by unifying it with a retrieved graph. If unification fails, one or more concepts in the query can be generalized, and a new attempt at unification can be made [4].

The outline of our architecture is as follows. As e-mail conversations for specification take place, users can identify *key mails*. These are e-mails in which, in the user's opinion, important collaboratory improvement suggestions are made. For example, the author of the very first e-mail to Tools-Yak, Eugene, indicated that two tools had been installed: Mailing Lists and Wikis. The mailing list was to support the discussion and list archiving-workflows, whereas the Wiki was to be used for management of the knowledge obtained in the collaboratory discussion.

As soon as a key mail has been recognized, the Index Wizard is invoked. In a simple (pseudo-natural language) dialogue, the user is quizzed by the system. To do so, it can use (an extended version of) the ontology given above. Many advanced querying techniques have been developed in the CG community over the years. A very simple scenario of how a key mail could be indexed is the following.

– After Eugene indicates that the current mail is a key mail, the system presents Eugene with list of indexing options:
  (a) Describe tool functionalities;
  (b) Describe workfbw properties;
  (c) Describe workfbw support

– Eugene selects option (c)
– The system retrieves all subtypes of this Tool-concept, including Blog, Mailing_List and Wiki; the same goes for the subtypes of Workfbw. The system presents these concepts as a simple HTML-pulldown menu, from which Eugene has to select which workfbw is supported by which tool. Furthermore, he is asked whether the discussion is about the results of an experiment already conducted, and whether this experiment was successful or not, or whether it is about a new experiment. The system would store this result by adding a relation to the STS-graph with a Succ_Hyp, Failed_Hyp, or Prop_Hyp concept, respectively.

The result of this human-machine dialogue is an STS (socio-technical system)-graph, which is stored in the STS-knowledge base. In case of the first mail, this graph obtained from the dialogue could look like this:

```
[STS_Pattern] -
   (Part) -> [Prop_Hyp]
   (Part) -> [Support] -
               (Inst) -> [Mailing_List]
               (Obj) -> [Discuss]
   (Part) -> [Support] -
               (Inst) -> [Mailing_List]
               (Obj) -> [Archive]
   (Part) -> [Support] -
               (Inst) -> [Wiki]
               (Obj) -> [Knowledge_Management]
```

Note that the graph represents a proposed hypothesis: the coordinator *expects* the tools to be used this way, but as the intensity of the discussion in the following half year has shown, many modifications were proposed. Had our system been available, many additional graphs could thus have been added to the knowledge base.

Now, assume that Mary, the coordinator of the PORT collaboratory, would like to find out about possibly useful experiences and contacts related to the following question she has: are there actually any successful user experiences with tools for knowledge management in collaboratories? Her interaction with the index-wizard leads to the following formal representation of her query:

```
[STS_Pattern] -
   (Part) -> [Succ_Hyp]
   (Part) -> [Support] -
               (Inst) -> [Tool]
               (Obj) -> [Knowledge_Management]
```

To find relevant conversations, the improvement query is projected on the STS-KB. In this case, the STS graph representing Eugene's mail is not retrieved, as it is not a specialization of the query. Assuming this is the only graph in the KB, the query fails. In a subsequent dialogue with the wizard, Mary decides to generalize her query: she is not just interested in successful experiments, but in *any* experiment. The first Part-relation of her query is therefore dropped. Querying the knowledge base again this time returns Eugene's graph. The Presentation Wizard uses this graph to initiate a dialogue with Mary on whether this is what she wants. The conversation looks interesting, and she wants to know more. The system then creates a maximal join of her query with Eugene's graph, stores this new graph in the KB, and initiates a new conversation to which both Mary and Eugene are invited. The link to the initial node of this conversation is stored with the graph in the knowledge base. The participants can then use the normal

discussion tools like e-mail and mailing lists to exchange tips and tricks. If in the future somebody has a query like Mary's initial one, this person will now immediately be guided to the log of the new query, which is much more specific than the thread started by Eugene's initial mail.

The system has been partially implemented using Adil Kabbaj's PROLOG+CG[3]. Of course, many technical extensions are conceivable. Once pointed to relevant discussions, the users could also use the more traditional discussion tool navigation functionality, for example search-options in a mailing list-archive, to further expand the context of interpretation of a particular improvement proposals. Additionally, content-based automatic mail-indexing tools, such as developed in the FCA community could also be added. However, such enriched content analysis is not the focus of this paper: our contribution has been to find a subtle balance between the strength of human collaborators (interpretation of rich e-mail content) and the power of machine systems (automatic inferencing of complex pattern knowledge), so that current bariers to socio-technical system evolution can be reduced.

In sum, the contribution of this approach is that collaboratory improvement is framed as a problem of socio-technical system evolution; that these abstract ideas are operationalized in a practical use-situation ; and that a fine-balanced mix of human natural language interaction is coupled with powerful graph matching to find links to rich human conversations, to be interpreted by people. The innovation of this approach is that the indexing by people is very simple, and hardly disruptive, while simultaneously being semantically very rich, as the index graphs are framed in terms of a socio-technical system improvement ontology. This meta-level reasoning, which makes good use of generalization hierarchies of the index graphs, helps people find relevant mails more easily than possible with current keyword searches. Such an approach will become truly powerful when multiple collaboratories start using this or a similar ontology: cross-community learning can then take place between communities that do not even know each other.

5 Conclusions

Improvement patterns are an essential element of collaboratory evolution. They capture collective wisdom and can be used in various ways, for example, in guiding discussions or designing tools that are customized to the complex needs of a particular community. Although improvement pattern *content* is quickly maturing, many *process* inefficiencies of how to effectively use these many rich and - by definition - partial patterns remain.

In this paper, we proposed the outline of a semi-formal method to help collaboratories more efficiently establish contacts and tune in to relevant informal collaboratory improvement discussions, across cases and communities. Our model is grounded in Engelbart's context-based philosophy of the socio-technical system improvement process in combination with a pattern-based view to deal with the partiality and specificity of this process. We formalized this model using conceptual graphs.

The basis of the formal model is an ontology of collaboratory improvement. We showed how the collaboratory improvement process can be modelled as a search for a

[3] http://www.insea.ac.ma/CGTools/PROLOG+CG.htm

match between an improvement query graph and the graph definitions representing the existing socio-technical system. Queries themselves become the basis for new socio-technical system graphs. All graphs are linked to specific conversations in the collaboratory. The main use of the retrieved graphs is to act as indices to previous community discussion, and as initiators of new discussion.

The ultimate goal in collaboratory improvement is to build active knowledge systems: systems that have the capability to solve practical and complex problems. Crucial to such systems is that they can inter*act* and not just inter*face* with the real world [7]. A collaboratory improvement system that would make use of the approach described in this paper, combined with, for instance, actor-based checks for improvement opportunities and automatic notification of key users, could be a major step forward on the way to more powerful and tailored collaboratory development.

Another implication of this work, is that it may give conceptual graph theory a class of practical and theoretical problems that do justice to its power and elegance. It has often been said that this theory was ahead of its time, in a way a solution looking for a matching problem. The area of collaboratory improvement, with its high significance in for instance the research and business domains could prove to be one of the 'killer problems' our field has been waiting for.

By combining the representational and reasoning power of conceptual graphs with the - fortunately - unique capabilities of human beings to interpret and frame improvement problems and their solutions, we have presented one operationalization of the essence of Engelbart's vision on using computer technology to *augment* collaborative communities. We hope that others will extend this preliminary work and use it to develop the collaborative methods and systems that are essential in an ever more complex and dynamic society.

References

1. E. Auramäki and K. Lyytinen. On the success of speech acts and negotiating commitments. In *Proceedings of the First International Workshop on Communication Modelling, the Language/Action Perspective (LAP'96), Oisterwijk, The Netherlands, July 1-2, 1996*, pages 1–12, 1996.
2. M. Bieber et al. Towards knowledge-sharing and learning in virtual professional communities. In *Proc. of the 35th Hawaii International Conference on System Sciences, Hawaii, January 5-7*, 2002.
3. J. Conklin, A. Selvin, S. Buckingham Shum, and M. Sierhuis. Facilitated hypertext for collective sensemaking: 15 years on from gIBIS. In *Proc. of the 8th International Working Conference on the Language/Action Perspective on Communication Modelling, Tilburg, the Netherlands, July 1-2*, 2003.
4. D. Corbett. *Reasoning and Unifi cation over Conceptual Graphs*. Kluwer Academic, New York, 2003.
5. A. de Moor. Making Doug's dream come true: Collaboratories in context. In *Proc. of the PORT Pragmatic Web Workshop, Borovets, Bulgaria, July 15*, 2002.
6. A. de Moor, M. Keeler, and G. Richmond. Towards a pragmatic web. In *Proc. of the 10th International Conference on Conceptual Structures, (ICCS 2002), Borovets, Bulgaria, July 15-19*, Lecture Notes in Artifi cial Intelligence. Springer-Verlag, 2002.

7. H. Delugach. Towards building active knowledge systems with conceptual graphs. In *Proc. of the 11th International Conference on Conceptual Structures (ICCS 2003), Dresden, July 2003*, pages 296–308, 2003.
8. D. Engelbart. Coordinated information services for a discipline- or mission-oriented community. In *Proc. of the 2nd Annual Computer Communications Conference, San Jose, California, January 24*, 1973.
9. D. Engelbart. Toward high-performance organizations: A strategic role for groupware. Technical report, Bootstrap Institute, 1992.
10. T.A. Finholt. Collaboratories as a new form of scientific organization. *Economics of Innovation and New Technology*, 12(1):5–25, 2003.
11. G. Fitzpatrick and J. Welsh. Process support: Inflexible imposition or chaotic composition? *Interacting with Computers*, 7(2):167–180, 1995.
12. J. Gillies and R. Cailliau. *How the Web Was Born*. Oxford University Press, 2000.
13. M. Keeler. Collaboratories: Improving theory and method. In *Workshop on Innovations in Digital Asset Management, Fraunhofer / IPSI, Darmstadt, Germany, October 6-8*, 2003.
14. R. Kling, G. McKim, J. Fortuna, and A. King. Scientific collaboratories as socio-technical interaction networks: A theoretical approach. In *Proceedings of AMCIS 2000, August 10-13, Long Beach, CA*, 2000.
15. R.T. Kouzes, J.D. Myers, and W. Wulf. Collaboratories: Doing science on the Internet. *IEEE Computer*, 29(8):40–46, 1996.
16. C.F. Kurtz and D.J. Snowden. The new dynamics of strategy: Sense-making in a complex and complicated world. *IBM Systems Journal*, 42(3):462–483, 2003.
17. J. Lederberg and K. Uncapher. Towards a national collaboratory: Report of an invitational workshop at the Rockefeller University, New York City, march 17-18. Technical report, National Science Foundation, 1989.
18. I. Nonaka, R. Toyama, and N. Konno. SECI, ba and leadership: A unified model of dynamic knowledge creation. *Long Range Planning*, 33:5–34, 2000.
19. NRC. National collaboratories: Applying information technology for scientific research. Technical report, National Research Council, Committee Toward a National Collaboratory: Establishing the User-Developer Partnership, Washington, D.C., 1993.
20. T. Renkema and E. Berghout. Methodologies for information system investment evaluation at the proposal stage: A comparative view. *Information and Software Technology*, 39(1):1–13, 1997.
21. D. Sanderson. Collaborative and cooperative mediated research. In T.M. Harrison and T. Stephen, editors, *Computer Networking and Scholarly Communication in the Twenty-First Century University*, pages 95–114. State University of New York Press, 1996.
22. D. Schuler. A pattern language for living communication. In *Participatory Design Conference (PDC'02), Malmo, Sweden, June*, 2002.
23. S.B. Shum and A.M. Selvin. Structuring discourse for collective interpretation. In *Proc. of Distributed Collective Practices 2000: Conference on Collective Cognition and Memory Practices, Paris, September 19-20*, 2000.
24. J. Thomas, C. Danis, and S. Greene. Socio-technical pattern language proposal. In *Pattern Language Workshop*, 2002.
25. B. Wellman. Computer networks as social networks. *Science*, 293:2031–2034, 2001.
26. T. Winograd. A language/action perspective on the design of cooperative work, report no.CSLI-87-98. Technical report, Center for the Study of Language and Information, Stanford University, May 1987.
27. W.A. Woods. Understanding subsumption and taxonomy: A framework for progress. In J.F. Sowa, editor, *Principles of Semantic Networks: Explorations in the Representation of Knowledge*, pages 45–95. Morgan Kaufmann, San Mateo, CA, 1991.