
Dirk Veiel
FernUniversitaet Hagen
dirk.veiel@fernuni-hagen.de

Stephan Lukosch
Delft University of Technology
s.g.lukosch@tudelft.nl

Joerg M. Haake
FernUniversitaet Hagen
joerg.haake@fernuni-hagen.de

Gwendolyn Kolfschoten
Delft University of Technology
g.l.kolfschoten@tudelft.nl

Abstract

Problem solving in groups usually consists of different phases. The quality of the results highly depends on applying suitable methods (e.g., to improve creativity or to support shared understanding). This is usually the responsibility of a trained facilitator. However, good facilitators are a scarce resource. In this paper, we explore the acceptance of automatic facilitation in a context-adaptive Group Support System (GSS) implemented on the CONTact platform – our infrastructure for implementing context-adaptive collaboration environments. We show how good practices related to the process of creative problem solving can be used to define interventions and identify the situations, in which they are needed, to achieve a good problem solving process. Based on these characterizations we show how to develop group-based adaptation policies implementing automatic facilitation. A pilot study with distributed autonomous teams of 4-5 people using our prototype system provide insights on the acceptance of automatic facilitation.

1. Introduction

In knowledge oriented organizations, there is often a need or demand for collaboration support. However, group work is challenging [23], and groups have limited self-steering capacity [23]. It has been shown that groups can benefit from collaboration support to work more efficiently and effectively [8]. Such collaboration support can be comprised of tools, processes and services that support groups in their joint effort. While today collaboration tools exist in a large variety, it remains challenging to appropriate and adapt tools to effectively support collaboration [12]. This makes it difficult for organizations to provide their teams with suitable and adaptable collaboration support. This gap is often addressed by using a professional facilitator.

The profession of facilitation consists of means to support groups in achieving their goals more efficiently and effectively by structuring and guiding their activities. The task of a facilitator requires both experience and extensive knowledge of group dynamics and facilitation methods. Facilitation involves, e.g., management of group activities, quality, relations between participants, resources, and time [20]. However, good facilitators are a scarce resource. In addition, the knowledge about the techniques and tools used is often not maintained in organizations. This limits the number of facilitated meetings within an organization.

Current Group Support Systems (GSS) do not provide automatic facilitation support. Rather, they either ignore facilitation needs or they aim at supporting a human facilitator through awareness features and access to good practice knowledge. Recently, thinkLets (scripts to prescribe collaborative work practices) were developed to aid groups in adopting more effective processes without a facilitator [16]. However, current group support tools do not explicitly support the application of such scripts [2].

In this paper, we explore whether automatic facilitation in a context-adaptive group support system is accepted by groups performing a specific instance of a creative problem solving process, i.e. brainstorming. Such a process usually consists of different phases: idea creation, categorization of ideas, elimination of duplicate ideas, and ranking of ideas to select those suitable for further elaboration. The quality of the results highly depends on applying suitable methods. Thus, we propose to provide a GSS with automatic support for facilitating the application of good practices fitting the needs of the current meeting situation.

Based on CONTact – our infrastructure for implementing context-adaptive collaborative environments – we developed a GSS that supports automatic facilitation. We show how good practices related to the process of brainstorming can be used to define interventions and identify the situations in which they are needed. Based on such characterizations, we show how to develop adaptation policies that implement such facilitation and adapt the group process and interaction. To sense and formalize users’ interaction with the system at runtime, we use a
context model for collaborative interaction [12]. Using this formal context representation group adaptation policies are executed when the specified situation occurs. This mechanism allows us to perform (1) interventions within a collaboration phase, such as using creativity techniques within the idea creation phase, as well as (2) shifts between phases, such as moving from an unstructured idea generation phase to the idea categorization phase when sufficiently many ideas have been generated or the group runs out of ideas. We conducted a pilot study to evaluate the feasibility and acceptance of such interventions. An analysis of user assessment and of the respective interaction processes indicates acceptance by the teams of 4-5 people performing such a brainstorming.

The next section discusses findings on facilitated brainstorming and suggests an improved group process together with requirements for automatic facilitation support. Then, we briefly introduce the idea of context-based adaptive systems followed by the presentation of our approach to automatic facilitation of brainstorming through context-based adaptations. Finally, we present the results of our pilot study, summarize our results and discuss implications and limitations.

2. Facilitating brainstorming

Brainstorming is a term originally coined by Osborn [24] as a way to come up with creative ideas. Osborne sets a very specific process and a set of rules to stimulate creative ideas. A broader term to encompass different ways to do brainstorming is ideation [26] or generation [3]. While ideation focuses on creativity, generation is a bit broader term encompassing not only creativity, but also gathering contributions of other types. There are three types of generation: (1) new or creative ideas (solutions, problem solving); (2) collection of expertise, facts, and explanations (knowledge sharing); and (3) collection of feedback, reflections, and assessments.

Electronic brainstorming tools and Group Support Systems that offer brainstorming capabilities have been around for 3 decades now. These systems offer several advantages over oral or paper based brainstorming [1]:

- **Parallelism.** Participants can add contributions simultaneously. This means that the disadvantage of waiting for a speaking turn is avoided. The result is a saving on meeting time [30].
- **Anonymity.** The software does not identify the source of information. This means that each contribution is valued on content. Dominance in a meeting is thus avoided [23].
- **Electronic recording and representation.** Every contribution is stored. This delivers more accurate and objective minutes, and allows users to use the data processing capacity of computers to aggregate voting results or to analyze text [1].

The problem of cognitive overload in electronic brainstorming has been the topic of investigation in many studies [10],[15],[23]. To reduce cognitive load in brainstorming we can offer the group a structure to organize their ideas on multiple pages [4],[19]. However, both ‘normal’ brainstorming and brainstorming in a structure have advantages and disadvantages [4],[19], see Table 1.

Table 1 shows quite some advantages of structured brainstorming in categories, as it reduces information overload, offers a scope and reduces the need to scroll through the page. However, the advantage of setting a scope also has a downside, of limiting the ‘out-of-the-box’ thinking. While one can create a category with the label ‘other’ this does not completely eliminate the scoping effect. Comparisons between nominal brainstorming groups (without interaction) and interactive brainstorming groups also show, that at first, people need to focus on the ideas or contributions that first come to mind [6],[32], as ideas of others will distract them. However, more often many know more than one, and some effect of inspiration of ideas of others has been reported and might also depend on the evaluation criteria of ideas; inspiration seems to have a more positive effect on quality than on productivity [25]. Furthermore, when considering brainstorming as a step in a larger process, sharing ideas and building on each other’s ideas will help to build shared understanding.

<table>
<thead>
<tr>
<th>Unstructured brainstorming (one big list)</th>
<th>Structured brainstorming (in categories)</th>
</tr>
</thead>
<tbody>
<tr>
<td>+ ideas of others inspire</td>
<td>+ ideas of other inspire</td>
</tr>
<tr>
<td>+ interaction builds shared understanding</td>
<td>+ interaction builds shared understanding</td>
</tr>
<tr>
<td>- information overload, cognitive overload</td>
<td>+ structure reduces information overload</td>
</tr>
<tr>
<td>- contributions do not fit on the page, scrolling is needed</td>
<td>+ less need for scrolling</td>
</tr>
<tr>
<td>- no basis for convergence</td>
<td>+ structure can support convergence</td>
</tr>
<tr>
<td></td>
<td>+ different topics can be addressed in parallel</td>
</tr>
<tr>
<td></td>
<td>+ structure offers a scope</td>
</tr>
<tr>
<td></td>
<td>- structure limits ‘out-of-the-box’ thinking</td>
</tr>
<tr>
<td></td>
<td>- less overview of contributions</td>
</tr>
</tbody>
</table>

We therefore propose a process where we take advantage of both approaches. At first participants brainstorm without a structure until they reach a state in which information overload is likely, then they cluster the ideas in categories. Afterwards, they brainstorm further within the structure.

This process can be supported with the use of GSS in combination with facilitation. In several GSS there is a capability to brainstorm ideas, and then cluster them in categories [8],[22],[28]. To design an approach with automatic facilitation support we turn to Collaboration Engineering (CE). In CE, the objective is to support collaboration by transferring the complex work practices
that are nowadays usually supported by expert facilitators to practitioners [5]. To support this transition groups are supported with thinkLets: scripts to prescribe collaborative work practices [16],[17]. However, this transfer would become easier if the group support tools would offer more guidance (restriction) throughout the processing of the script [2].

Some earlier work has made an effort to offer tools that support clustering without facilitation. These tools let participants cluster ideas asynchronously [13],[14]. We, however, want to keep the efficiency of synchronous group interaction. For this purpose, we propose a context-adaptive collaboration support system that supports the transition from single list brainstorming to clustering and structured brainstorming automatically. The following steps would need to be supported by the facilitator and thus embedded in the system:

a. Instruct the group to brainstorm in parallel to identify solutions to the problem
b. Detect when information overload becomes likely or when the group runs out of initial ideas
c. Instruct the group to create clusters of the ideas they have, and label those clusters
d. Ensure that the amount of clusters is balanced (too few clusters does not offer sufficient structure, too many increases complexity)
e. Cluster all ideas in categories
f. Instruct the group to look at the categories to add new ideas to complete the set of solutions, also reflect on the scope of the categories to see if this matches the solution space
g. Stop the group when the set of solutions is complete and consistent

Adaptive systems have been developed in [7],[9],[31] to support adaptation based on the effort or performance of one user. The system that we envision in this paper needs to offer adaptations based on group effort and group performance. We therefore propose an adaptive system based on group context.

3. Context-based adaptations for groups

This section describes the idea of context-based adaptations for groups and the components required for implementing them [12]. We define context as: “context includes all information which is necessary or helpful to adapt a shared workspace to better fit the needs of a collaborating team. This implies that the context contains information about the team as well as about the current collaboration situation” [29]. Consequently, context-based adaptations affect the computer-mediated interaction of the group, and are based on group variables and outcomes, making their acceptance and effect highly critical and prone to conflict.

Typically, context-adaptive systems use an adaptation cycle similar to the one shown in Figure 1. Initially, a user performs actions in an Application (cf. ①), which informs the Sensing Engine (cf. ②) resp. the Sensing Engine monitors actions performed in an Application, and updates the Context Model. These updates can trigger an Adaptation Engine (cf. ③) that uses Adaptation Policies to keep track of situations requiring adaptations. An Adaptation Policy consists of a condition and an action block. The condition block uses concepts from the context model to specify a situation requiring an adaptation. The action block contains so-called adaptation actions that adapt the Application. When the Context Model has been updated the Adaptation Engine evaluates the condition blocks of all Adaptation Policies. The Adaptation Policies evaluating to true are selected for execution, and the bindings of the variables of the respective condition blocks are used to generate an Adaptation Log from the respective action blocks. The Adaptation Log is then sent to all affected clients. The Adaptation Component (cf. ②) uses the Adaptation Log to adapt the Application accordingly. As mentioned above, the Sensing Engine monitors these adaptations, and the cycle starts again. This approach creates and maintains a context model for the group by relating individual user’s interactions into a coherent group state. Thus, it captures the interaction and state of group collaboration. Such a group context model can then be exploited by adaptation policies to configure the group’s application environment to the needs of the group. In order to achieve this, an adaptation policy can cause different adaptations to individual users’ applications. This distinguishes our approach from work on single-user adaptive systems.

![Figure 1: Adaptation cycle and related components](image)

In [12],[21],[29] we present the related conceptual architecture that has been successfully implemented in our Context-based Adaptation and Collaboration Technology (CONTact) platform.

4. Automatic facilitation of brainstorming by context-based adaptations

As mentioned in section 2, we propose a process to support good practices in brainstorming, i.e. at first participants brainstorm without a structure until they reach a state in which information overload is likely, then they cluster the ideas in categories. Afterwards, they brainstorm further within the structure. In order to support this process
we have to analyze it in more detail from a user’s perspective:

1. The user starts the GSS and joins the brainstorming, initially without a structure.
2. For idea creation by brainstorming without a structure, the GSS should present a UI where the user can enter new ideas and see the topic of the brainstorming, some instructions (see a), and the ideas created by others.
3. When information overload is likely (see b) (e.g., when the total amount of ideas exceed the number of ideas that can be presented on the screen two times, i.e. users have to scroll a lot) the users should move to the categorization phase.
4. For categorization of ideas, the GSS should enable users to create, rename and delete categories (the total number of categories is limited to seven), and to move ideas from one category to another (see c.).
5. When all ideas have been moved to related categories (see e), the users should move to the brainstorming in a structure phase.
6. For brainstorming in a structure, the GSS should support users in creating new ideas in a category as well as manipulating the category structure (see categorization phase, step 4 and f).
7. When the user leaves the brainstorming the GSS should close related UIs.

When we compare the facilitation instructions in section 2 and the user actions above, the system supports tasks a, b, c and e, while a balanced cluster structure (d), reflection (f) and terminating the task (g) is left to the group themselves through natural leadership or consensus. The trickiest steps are to detect the transition from brainstorming to clustering and to detect the transition from clustering to structured brainstorming. Such transitions between phases require some assessment of the results, and the progress of the group, and adaptation of functionality and the related UI to support the kind of computer-mediated interaction desired in the respective phase. The description of situations indicating phase transitions can be seen as conditions that have to be fulfilled to trigger related interventions (e.g., moving to the next phase and changing the UI accordingly).

Using the above steps we can describe how the GSS must change in order to provide the interaction opportunities required in each phase or situation. When the situation described in step 3 occurs, the GSS should move from the idea creation phase to the categorization phase. Thus, a short explanation regarding the intervention (i.e. the phase transition) and the new instructions should be presented to the user, and the capability of creating new ideas has to be enabled.

Within step 4 (categorization phase) or 6 (structured brainstorming phase) there may be one additional change at the UI because of the limitation of the number of categories that can be created and used. As soon as seven categories have been created the related UI elements have to be removed or disabled and a short explanation should be presented to the user.

We now describe how we use our context-adaptive GSS to support the required adaptations. Figure 2 shows the process to be supported by our GSS. For reasons of simplicity, we assume an ‘in-meeting’ situation, i.e. one person has already prepared the brainstorming and created related documents. After saving the document, the system sends invitations to all participants when the brainstorming should take place. In this situation the process shown in Figure 2 is started.

![Figure 2: Facilitated brainstorming process](image-url)

As soon as the first user accepts the invitation (step 1), we execute the <init> step (cf. Figure 2). This sets up the brainstorming session, creates the so-called root folder where the uncategorized ideas are kept, and initializes related components. Then, the idea creation phase (step 2) is started and the corresponding UI is shown to all participants that accepted the invitation and therefore joined the brainstorming session. Within the idea creation phase, participants brainstorm without a structure. As soon as they reach a state in which information overload is likely (e.g., when users have to scroll a lot to go through all ideas; step 3), intervention 1 (cf. Figure 2) is applied. This causes the process to move to the categorization phase (step 4). Within this phase the participants can create new cluster folders (representing categories) and cluster the ideas into them. When a pre-defined number of cluster folders have been created (either in the categorization or the idea creation & categorization phase), intervention 2 (cf. Figure 2) is applied that disables the functionality to create new cluster folders. Intervention 3 (cf. Figure 2) is applied as soon as the participants have categorized all ideas from the root folder into other folders (step 5). This leads to the idea creation & categorization phase (step 6) where the participants brainstorm further within a structure (cf. Table 1 for benefits and drawbacks of this approach). As soon as
all participants left the brainstorming session (step 7) the results are stored and the given process ends (<end>).

Not explicitly shown in Figure 2 is the fact that we support so-called latecomers, i.e. persons who join the brainstorming after the idea creation phase. Latecomers will see all information of the brainstorming (i.e. ideas and cluster folders) and their UI gets adapted according to the last triggered intervention.

To support this process and related interventions, we need to extend the original domain model for collaborative interaction presented in [12]. Figure 3 summarizes the extended domain model. This model distinguishes different concepts that describe collaboration in an electronic brainstorming environment and relations between these concepts. Concepts are depicted as ovals whereas relations are depicted as directed arrows. We start exploring and explaining the model in Figure 3 with the concept of an Actor (see lower part of Figure 3). The domain model assumes that an Actor is a member of a Team (Group) and has a Role defined by the User Workspace. Applications are started from within the User Workspace and thus the workspace can ensure pre-defined Roles. Each Role allows an Actor to perform specific Actions. The available Actions are defined by the supported Application Functionality of an Application as described later in this section. Actors interact with the Application by performing Actions allowed by their Roles. Roles define interaction possibilities within an application, e.g., in a shared writing application an author might perform all edit actions whereas a reviewer can only comment existing text. The Actions are performed by the corresponding Application.

**Figure 3 Domain model for collaboration in a shared workspace for brainstorming**

An Application uses Services to access the Artifacts. When considering an electronic brainstorming we first have to add new concepts and relations to reflect the Artifacts of a brainstorming, i.e., Ideas and Cluster Folders, and corresponding relations. For clarity reasons we omitted the labels of relations showing the specialization between Artifact and Idea resp. Cluster Folder. Artifacts use Services to notify Applications about changes. Each Application is part of a User Workspace. Finally, the Application Functionality specifies the functionality an Application offers, e.g., with respect to communication, shared editing, or awareness. Most of these classes are derived from patterns for computer-mediated interaction [27], which describe best practices for designing tools for collaboration.

The facilitated brainstorming process (cf. Figure 2) focuses on the ‘in meeting’ situation. Thus, we have not presented the concepts and relations of the preparation phase in Figure 3. Figure 4 shows the Application Functionalities and Actions provided by them. An Electronic Brainstorming requires two Application Functionalities, Idea Creation and Categorization. All of these concepts are specializations of Application Functionality. The facilitated brainstorming process (cf. Figure 2) consists of the three phases Idea Creation, Categorization and Idea Creation & Categorization. For each phase we define which Application Functionalities, i.e. which actions, are permitted in them.

**Figure 4 Extensions of Application Functionality and Action**

An Electronic Brainstorming is capable of actions like Start, Stop, Next Phase and Show Intervention. The initiator of a brainstorming session (i.e. the first participant accepting the invitation) starts a related brainstorming session. This is done implicitly (i.e. without user participation) by executing the Action Start. After the initialization is done (<init>), the initiator can move to the Next Phase, i.e. Idea Creation. Within this phase users can Create Ideas. Triggering the Action Next Phase, the phase Categorization will be activated. This is done by intervention 1 when the pre-defined number of ideas has been created. This phase supports Actions like Create Folder and Categorize Idea. When a pre-defined number of folders have been created, the function to create further folders is deactivated (intervention 2). When all ideas have been moved into folders, the process switches into the idea creation & categorization phase (intervention 3). Here, more ideas can be created in folders, ideas can be moved between folders, and more folders can be created up to the maximum number of folders. When the Electronic Brainstorming is finished (i.e. all participants left the brainstorming session) the related Stop Action will be triggered (<end>). Note: we omitted the labels of specialization arrows starting at Application Functionality and Action.

We can now define adaptation policies that implement the facilitation interventions 1-3 (cf. Figure 2) by using
above concepts and relations to specify situations requiring interventions in the condition block of related adaptation policies. Within the action block we encode the related adaptation actions. As shown in Figure 4, one action supported by Electronic Brainstorming itself is Show Intervention. In our prototype this action presents an intervention text to all users of a brainstorming session.

In the following, we assume that the brainstorming is properly prepared and started; users have joined, and are currently brainstorming without structure (cf. idea creation phase in Figure 2). The first adaptation policy (intervention 1) tries to reduce the information overload by moving to the categorization phase as soon as a pre-defined amount of ideas has been created. The related adaptation policy looks like this:

```plaintext
When ClusterFolders.get("ROOT").Ideas.count >= MAX_IDEAS firstIteration == true Do firstIteration = false ShowIntervention("You reached the total number of ideas that require to move from idea creation to categorization phase. Please create 3-7 cluster folders and categorize the ideas!") NextPhase("Categorization") Disable("Create Idea") Hide("Create Idea")
End
```

The conditions are specified in the conditions block after the keyword “When”. In the above adaptation policy, we continuously check the total number of ideas created within this brainstorming session. When the value of “Ideas.count” of the root cluster folder reaches the predefined threshold MAX_IDEAS and we are in the first iteration (after starting a brainstorming session this is the case) information overload is likely. Therefore, the related action block should be applied at the client of each participant. In our experiment we set the value of MAX_IDEAS to twice the number of visible ideas in our client (i.e., 34). In our opinion this is the value where information overload is likely because the user has to scroll a lot. The first action in the action block sets the flag that the first iteration is finished. The first real action to be executed in all users’ clients will show the related intervention text. The next actions will affect the UIs of the clients, i.e., activate the phase Categorization, and disable and hide Create Idea functionality. The related actions are stored in the Adaptation Log, which gets transferred to and executed by the Adaptation Component at related clients.

Now, users work in the Categorization phase and can create up to 7 cluster folders (MAX_CFOLDERS) and start categorizing the ideas. As soon as they created 7 cluster folders and creating cluster folder is enabled (when starting the Categorization phase this is the case) the following adaptation policy (intervention 2) will be triggered:

```plaintext
When ClusterFolders.all.count >= MAX_CFOLDERS createFolders == true Do createFolders = false
```

The actions specified in the action block will, firstly, show the given intervention text that describes the current situation and points out the restricted functionality. Secondly, the functionality to create new cluster folders will be disabled and hidden.

Usually, users then start to categorize the given ideas and move them to related cluster folders. As soon as all of the initial ideas have been moved to other folders (i.e., the root folder is empty) and we run the first categorization cycle the following adaptation policy (intervention 3) is triggered:

```plaintext
When ClusterFolders.get("ROOT").Ideas.count == 0 firstCycle == true Do firstCycle = false ShowIntervention("Great! There are no ideas left for categorization. Now, you can create more ideas. You can easily add new ideas to existing cluster folders by selecting the cluster folder first.") Enable("Create Idea") Show("Create Idea")
End
```

Firstly, we set the flag that we have finished the initial categorization. Secondly, we show the intervention text to all participants of the brainstorming session and, thirdly, we enable and show the functionality to create new ideas. After executing these adaptations, users work in the Idea Creation & Categorization phase and can easily brainstorm further, e.g., by choosing a given cluster folder (the root folder is still available) and brainstorm new ideas related to it. Apart from idea creation, users can also categorize ideas, as mentioned above. With this functionality, users can brainstorm in a structure.

5. Results from Pilot Study

Using the CONTact platform with (1) an application supporting collaborative electronic brainstorming and (2) the adaptation policies described above we are able to support automatic facilitation of brainstorming sessions following the process shown in Figure 2. We set up an experiment to evaluate the feasibility of automatic intervention in the collaboration process and the acceptance of such automatic interventions.

5.1. Method

**Design:** Groups of four to five participants were instructed to use our GSS to perform a brainstorming and categorization of ideas on a predefined topic. The meeting
process and outcome was analyzed to determine whether interventions were triggered in the right situations, and whether such interventions were accepted and followed by the group.

**Setting:** Groups of four to five participants had access to a shared workspace composed of a Skype group conference for full-duplex audio communication and the context-adaptive brainstorming tool.

**Subjects:** 3 groups of 4-5 students from the University of Hagen (Germany), enrolled in B.Sc. or M.Sc. programs in Computer Science, and Delft University of Technology (The Netherlands), enrolled in Ph.D. programs, participated in the experiment. If possible, students who already had collaborated with each other were assigned to one group in order to lower the time needed to get fully functional groups. Otherwise, students were randomly assigned to groups.

**Table 2:** Questionnaire items

<table>
<thead>
<tr>
<th>No.</th>
<th>Question</th>
<th>Scale</th>
</tr>
</thead>
<tbody>
<tr>
<td>Q1</td>
<td>How did you experience the moment when the system suggested clustering the ideas in categories?</td>
<td>open text</td>
</tr>
<tr>
<td>Q2</td>
<td>How did your group decide to work after the system suggested you to cluster ideas?</td>
<td>open text</td>
</tr>
<tr>
<td>Q3</td>
<td>How would you assess your own productivity (quantity and quality)…before the clustering? …after the clustering?</td>
<td>open text open text</td>
</tr>
<tr>
<td>Q4</td>
<td>How would you assess your group’s productivity (quantity and quality)…before the clustering? …after the clustering?</td>
<td>open text open text</td>
</tr>
<tr>
<td>Q5</td>
<td>Did you appreciate the suggestion of the system to cluster ideas?</td>
<td>open text</td>
</tr>
<tr>
<td>Q6</td>
<td>Would you say that the suggestion of the system to cluster ideas contributed to the success of the brainstorm? If yes, to what extent: If yes, in what way did it improve the brainstorm?</td>
<td>limited–radically?ptn open text</td>
</tr>
<tr>
<td>Q7</td>
<td>Why did you think the system suggested to cluster the ideas?</td>
<td>open text</td>
</tr>
<tr>
<td>Q8</td>
<td>Would you appreciate other suggestions from the system to alter the way the group is working?</td>
<td>open text</td>
</tr>
<tr>
<td>Q9</td>
<td>Do you have ideas for improvement of these automatic system suggestions?</td>
<td>open text</td>
</tr>
</tbody>
</table>

**Task and Procedure:** The students were asked to brainstorm on ways to achieve a more ‘green’ ICT infrastructure at their university, or on ways to use ICT to reduce the carbon footprint of the university. This could involve anything from motivating people to switch off their PC to cloud computing to save server energy. The students got instructions on how to start the brainstorming application and Skype conference, and how to log in. Other than this, there was no facilitation or intervention by humans. The users were entirely guided by the system.

We did not allot a timeframe for the task, and rather let the group determine when they were finished with the task. To ensure that they would brainstorm sufficient ideas to go through the intervention process, we indicated in the assignment that they needed to offer a large set of common and innovative ideas for green ICT, and that they should brainstorm until they felt confident that they had enough ideas, and a complete overview to advise the board of the university on their green ICT policy.

**Evaluation infrastructure: A log file of all automatic interventions and all user operations was recorded for analysis. Furthermore, we captured a screencast and recorded the group communication during the brainstorming. After the experiment we asked each participant to fill out a questionnaire. The questionnaire contained the questions presented in Table 2.**

**5.2. Results**

We analyzed the results of the experiments based on the log of actions (automatic interventions, user operations) and the questionnaires per group. Using the log files and the questionnaires we first analyzed the process itself.

**Table 3:** Results on measures from logs and binary/numerical items from questionnaires

<table>
<thead>
<tr>
<th>Factor</th>
<th>E1</th>
<th>E2</th>
<th>E3</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: 1st intervention was triggered?</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>2: time needed for understanding 1st intervention: move to categorization phase (seconds)</td>
<td>40</td>
<td>64</td>
<td>49</td>
</tr>
<tr>
<td>3: 3rd intervention was triggered?</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
</tr>
<tr>
<td>4: time needed for understanding 3rd intervention: move to creation &amp; categorization phase (seconds)</td>
<td>117</td>
<td>32</td>
<td>49</td>
</tr>
<tr>
<td>5: phase 3 time spend on additional brainstorming</td>
<td>13:03</td>
<td>4:11</td>
<td>38:15</td>
</tr>
</tbody>
</table>

As shown in Table 3, the time it took the group to understand the first adaptation and to start clustering was different (row 2). All groups that worked through the intervention created 3-7 clusters, and clustered the 34 ideas they created before the intervention. After clustering, the system suggested the group to brainstorm further (phase 3). All groups brainstormed more ideas, but again the time to understand the adaptation (row 4) and the time on phase 3 (row 5) were different. In phase 3, two groups seemed to be inspired by the clustering exercise; one group only added a few additional ideas. Looking at how groups worked in phase 3 reveals that groups applied a mix of structured brainstorming (i.e. creating new ideas within a cluster) and unstructured brainstorming (creating ideas in the root folder, and moving them into clusters later). Groups who did not create the maximum number of clusters in phase 2 did actually create more clusters in phase 3. This indicates
that groups found the categorization task helpful and were able to apply it also in phase 3 without more interventions.

Figure 5 shows the number of ideas generated over time and the moments of intervention. Experiments 1 and 2 show that the intervention created a pause, after which the group brainstormed further. In experiment 3, we see a reduction in the ideation rate before the intervention and an increase after the intervention.

Based on this overview, we used the questionnaire (answers to questions Q1 to Q9, see Table 2) to zoom in on how the users experienced the interventions (adaptations). Users experienced the first intervention (adaptation moving to phase 2: categorization) differently (Q1). Some found it surprising; others were already looking for an opportunity to cluster. There were also some users that noted the change later than others, as it appeared at the top of the screen, while users were typing at the bottom of the screen. Another remark was that the change should have happened earlier.

Groups responded differently to the first adaptation (Q2). Some groups discussed what to do, or an emergent leader took charge of the clustering task, others just started identifying clusters and clustering ideas. From (Q3 & Q4) we learned that most participants found that productivity reduced after clustering, or stayed the same and that just before clustering, productivity was also lower.

From (Q5 &6) we learned that several users thought that the first intervention should be earlier or later, indicating that a more sensitive trigger for the adaptation is required. Some clearly recognized the benefits of the intervention such as that it structured the brainstorm, created overview, helped to identify similar ideas, triggered new ideas, supported reflection and prepared for evaluation of ideas. Others simply indicated that the adaptation was made because a certain number of ideas was created, indicating that they did not see the positive effect of the intervention (Q5, Q7). Participants from two groups indicated in the questionnaire that the intervention improved the brainstorm, members from one group reported positive and negative impact, also because of the choice for cluster labels (Q6). Also some interventions were perceived as an enforcement of process adaptation, while they would have liked the system to only suggest adaptation (Q5). All groups indicated that the effect of the interventions on quality and productivity is not clear to the group.

The above ambivalent observations can also be found in the recorded conversations of the different teams. After the first adaptation policy that asks the participants to start clustering ideas has been applied, several participants reacted surprised or confused, e.g.:
- “Oh no, my bar is gone, I cannot type anything new.”
- “Who removed my bar?”
- “You created the total number of ideas. What? I can’t create new ideas.”

Once the categorization phase was on its way one participant stated: “Sorting is fine, but why is there a limit of 34? This is a weird number.” Another one even stated: “Oh, the system works. I like the system.” While another participant complained: “This categorization is really annoying.”

When all ideas are categorized, our GSS asks the participants to continue their brainstorming within the categories (intervention 3). Here, most participants accepted the intervention immediately. One participant commented the intervention very positively: “Hey, the system sends you messages that’s cool.”

The reaction on the second intervention that asks to rename or generalize folders was again ambivalent. One participant stated: “You created the maximum number. Ok.” Another participant, however, stated: “Maximum number of ideas. Maximum number of folders. Limited creativity.”

Besides the adaptation acceptance, we also have some preliminary effects of the intervention on productivity. In a similar study in [18], an experiment was performed with
217 students in 42 groups of 4-6 students brainstorming solutions for a case study. Their assignment was to use a GSS tool and to brainstorm at least 30 ideas. On average the groups brainstormed 38 ideas, from those they created on average 5 clusters. Note that we removed 4 outlier groups from the data set. Table 4 shows the productivity of the three groups in our experiment. While results are very inconclusive, we see a significant amount of ideas brainstormed after the clustering task, increasing the total productivity well beyond the 38 ideas of the groups in the above experiment.

<table>
<thead>
<tr>
<th>Factor</th>
<th>E1</th>
<th>E2</th>
<th>E3</th>
</tr>
</thead>
<tbody>
<tr>
<td>No. of ideas created in phase 1</td>
<td>34</td>
<td>34</td>
<td>34</td>
</tr>
<tr>
<td>No. of clusters created in phase 2</td>
<td>3</td>
<td>3</td>
<td>6</td>
</tr>
<tr>
<td>No. of ideas moved to clusters in phase 2</td>
<td>34</td>
<td>34</td>
<td>34</td>
</tr>
<tr>
<td>No. of clusters created in phase 3</td>
<td>4</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>No. of ideas created in clusters (=_=root) in phase 3</td>
<td>8</td>
<td>3</td>
<td>5</td>
</tr>
<tr>
<td>No. of ideas created in root in phase 3</td>
<td>10</td>
<td>7</td>
<td>14</td>
</tr>
<tr>
<td>No. of ideas moved to clusters in phase 3</td>
<td>10</td>
<td>3</td>
<td>14</td>
</tr>
<tr>
<td>No. of ideas created after clustering</td>
<td>18</td>
<td>10</td>
<td>19</td>
</tr>
</tbody>
</table>

5.3. Discussion

Our experiments show that automatic intervention in the brainstorming process can be achieved through context-based adaptations. Our context-adaptive brainstorming system enabled groups to successfully perform a structured brainstorming process. In total, our results suggest that groups tend to accept interventions aiming at process improvements, though there are also critical voices. Some of the critical voices can be addressed with more sophisticated sensors that choose the moment of intervention in such a way that an ongoing interaction with the system is not stopped. The latter happened especially during the first intervention when participants – out of a sudden – could not enter new ideas.

The graph in Figure 5 shows us that we could detect a decline in the ideation rate. The decline could trigger the intervention; in this case the intervention came in a bit late for group 3 and slightly early for groups 1 and 2. In group 2 this was suggested, but there was also a participant who stated that the intervention came just as they were having fewer ideas. Similar, in group 1 one participant stated that the intervention came in time as they were starting to create similar ideas. Group 3 indeed stated that they were already looking for a way to structure their ideas when the intervention took place, suggesting it was rather late for them. It also seems that enforcing process improvement is not always accepted. Groups that learn to work through the different phases might later want to determine by themselves when to adapt the system for the next phase.

When asked about suggestions for other interventions (adaptations) we got the following ideas (Q8 and Q9): more visible notification; don't disrupt typing; support in clustering; create adaptation based on rules the group determines in advance; create adaptation based on semantic information; detect divergence in focus, then intervene; stimulate creativity when participants are idle for some time; offer ranking after the brainstorm is finished. These suggestions indicate that the participants are open to receive automatic facilitation support, if it is understandable and proves useful.

6. CONCLUSIONS

In this paper we introduced a method for deducing a specification of interventions, which aim at improving computer-mediated interaction in groups, from the user’s view of good collaborative problem-solving practices. It was shown how the method could be used to analyze brainstorming good practice, specify interventions, and implementing them as adaptation policies in a context-adaptive GSS. Furthermore, the results of a pilot study on the acceptance of such automatic adaptations by three teams performing a brainstorming process were presented. Results indicate first evidence that groups tend to accept interventions aiming at process improvements, though more thought should be given to whether process changes are enforced or just recommended. In addition, several groups articulated their interest in being able to define or modify interventions. Acceptance may be improved by letting users determine such rules by themselves.

Our results demonstrate that automatic facilitation approaches are possible, if good facilitation practice is known and can be expressed in machine executable form. Context-adaptive systems have proven useful for both formalization and automation of the detection of useful intervention situations as well as for formalization and automation of the execution of useful interventions at a group level. Several suggestions for improvements made by participants in our experiments can easily be expressed as additional adaptation policies in our system.

In our opinion, the proposed method can also be applied in other domains resp. problem-solving processes, where good practice knowledge exists. While the resulting interventions may be implemented as modifications of any GSS, e.g., at the source code level, the presented implementation as adaptation policies in an context-adaptive GSS presents several benefits: (1) the modification resp. improvement of adaptation policies may be simpler as recoding entire application suites; (2) different collaborative problem solving processes (such as brainstorming, ranking, decision making) may be supported.
in parallel by orthogonal adaptation policies; and (3) complex collaborative problem solving processes consisting of several steps resp. phases performing simpler collaborative problem solving processes may be orchestrated and supported through a set of related adaptation policies. Thus, our approach allows for more experimentation and flexibility in both, future research and application in practice.

Obviously, our study has a number of limitations: Firstly, with just three groups of four to five people it is not possible to generalize. However, though we saw a diversity of perceptions and behaviors, we still could see positive effects of our approach. In the next steps of this research more experiments are required for the generalization of results. More specifically, we want to examine the effect of enforcing adaptations or only suggesting a change of activity, and explore ways of visualizing adaptations effectively. Here, we benefit from a policy-based adaptive system since it becomes much easier to experiment with different policies and with different user interfaces.

7. References

[26] Santanen, E.L. Resolving Ideation Paradoxes: Seeing Apples as Oranges through the Clarity of ThinkLets. HICSS 2005.