Towards effective collaborative design and engineering

Stephan Lukosch
Delft University of Technology
Faculty of Technology, Policy, and Management
Jaffalaan 5, 2628 BX Delft, The Netherlands
s.g.lukosch@tudelft.nl

Gwendolyn Kolfschoten
Delft University of Technology
Faculty of Technology, Policy, and Management
Jaffalaan 5, 2628 BX Delft, The Netherlands
g.l.kolfschoten@tudelft.nl

ABSTRACT

Effective collaborative design and engineering has to deal with various challenges. It is essential to create a shared understanding and facilitate interaction in such a way that effective collaboration becomes possible. Free riding, group think or hidden agendas need to be addressed by rarely available process facilitators. Available tools are not regularly used, are not intuitive and often are difficult to adapt to the changing group needs. In order to tackle the above issues, we want to enable effective collaborative design and engineering by offering intelligent collaboration support that supports facilitators of collaboration processes when monitoring collaboration processes and planning process interventions or tool adaptations.

Categories and Subject Descriptors

H.4.1 [**Office Automation**]: Groupware; H.5.3 [**Group and Organization Interfaces**]: Computer-supported cooperative work; K.4.3 [**Organizational Impacts**]: Computer-supported collaborative work

General Terms

Design, Human Factors

Keywords

Collaboration support systems, intelligent collaboration support, facilitation, group support systems

1. INTRODUCTION

Collaboration has become a critical skill as products and services are becoming increasingly complex, no individual has the skills to design, develop and deliver these alone. Collaboration is however, not without challenges. On a group level, it is essential to create a shared understanding, define rules for decision-making and facilitate interaction in such a way that effective collaboration becomes possible [13]. On a process level, free riding, dominance, group think, hidden agendas, are but a few phenomena in group work that make it a non straight-forward effort [16].

Groups might not be able to overcome the challenges of collaboration by themselves [16]. Even if groups are able to accomplish their goals, they can often collaborate more efficiently and effectively using collaboration support [6]. Collaboration support can be comprised by tools, processes and services that support groups in their joint effort. In knowledge oriented organizations, there is often a need or demand for collaboration support. However, tools and technology for group support exist in a variety of shapes from complex computer systems, as e.g. Group Support Systems (GSS), to simple boxes with cards and pencils. Each of these tools can be used by the group to be more successful in sharing ideas and indicating relations and preferences, but current challenges emerge from the fact that available tools are not regularly used, are not intuitive and often are difficult to adapt to the changing group needs [7]. This makes it difficult for organizations to provide their teams with a suitable and adaptable collaboration support that help them accomplish their goals efficiently and effectively.

As discussed in [7], current collaboration support systems focus adaptations with a limited scope. They are either restricted to specific domains or to specific aspects of collaborative work, often focusing on awareness or knowledge management. Compared to this, we aim to create intelligent collaboration support that creates a shared understanding, facilitates collaborative actions across various geographic, temporal, disciplinary, and cultural boundaries and provides intuitive and adaptive tool support. This will allow us to offer collaboration support for a variety of collaborative tasks in a way that groups can use it for themselves without the need for extensive training or a professional facilitator. In this paper, we will as a first step propose a conceptual framework towards intelligent collaboration support. We modeled collaboration processes and identified factors suggesting process changes as well as adaptations. These factors are the basis for this framework of intelligent collaboration support, which will offer us a first step in monitoring groups and predicting the need for facilitation interventions.

In the next section we will explain in detail how facilitators guide collaboration processes. This will lead to a conceptual framework of collaboration support interventions, presented in section 3. Next we will present how this framework can be used to identify specific collaboration situations to create intelligent collaboration support. We will then reflect on this design and end with conclusions and a research agenda.

2. FACILITATING COLLABORATION PROCESSES

One of way of supporting groups in achieving their goals more efficiently and effectively is to support the group by structuring and guiding their activities. This skill and profession is called facilitation. The facilitation task is described extensively in GSS literature [1, 8, 14]. The task of a facilitator requires both experience and extensive knowledge of group dynamics and facilitation methods. This tasks involves for instance management of the activities the group is performing, quality of their deliverables, relations between the participants and the use of resources and time [9]. This type of process guidance is often offered by someone external to the group, to ensure impartiality and objectivity.

In an effort to reduce the need for professional facilitators, researchers have been coding facilitation practices to enable the separation of the design task of a facilitator and the execution task [11]. In this way a master facilitator called collaboration engineer, can design and transfer a collaborative work practice to practitioners to execute it for them selves based on a short training. This approach is called Collaboration Engineering [3]. To ensure the predictability and transferability of the collaborative work practice, they are designed with design patterns called thinkLets [4].

To realize an intention by means of intervention, two types of interventions are required [2]. First, there are static interventions in which one or more commands are given to initiate the key activities of a process. We will refer to this kind of communication as an instruction intervention. Second, there are dynamic interventions intended to adjust the actions performed by the group to resolve a discrepancy between the facilitator's intentions and the groups' actions. These interventions depend on emergent conditions. We will call these messages adjustment interventions.

The conceptual design of a thinkLet exists of a set of instruction and adjustment interventions described as rules [4]. These rules are similar to rules mimicking human behavior in avatars [2]. Each rule describes for a role an action that needs to be performed using a capability under some set of constraints to restrict those actions. Further, some thinkLets include conditional rules for frequently-required adjustment interventions because specific discrepancies manifest predictably during the execution of an activity based on the thinkLet.

An example of a set of rules are captured in the LEAFHOPPER thinkLet [4]:

1. Allow participants to add in parallel any number of contributions to any category.
2. Allow participants to add only contributions that are relevant to the categories in which they are placed.
3. Allow participants to add only contributions that match to the contribution specification.
4. Let participants shift focus from category to category as interest and inspiration dictate.
5. Ensure that participants read the contributions of others for inspiration.

3. CONCEPTUAL FRAMEWORK

In order to provide intelligent collaboration support, we first need to identify the key goals that guide facilitation interventions. When facilitators intervene to initiate activity they can offer these at different levels [15]:

1. **Collaboration process design**: Interventions to guide collaborators in choosing appropriate tools and techniques to support the collaboration process.
2. **Collaboration process execution**: guidance to move from one activity to a next activity, changing the collaboration support environment to transfer between activities, while taking documents and decisions along to a next phase.
3. **Collaboration process guidance**: Activities need to be initiated and guarded to execute the collaborative activity.
4. **Collaborative behavior guidance**: guidance in determining and adjusting improves collaborative effectiveness.

In a face to face context, facilitators can make adjustment interventions based on behavior of group members, including communication with group members, quality of the output of the group, and progress versus planned time for the group task. Based on our experience and discussion with expert facilitators, the following list is a first attempt to identify factors used to determine the need to make an intervention:

- **Group**: behavior, emotions, communication, body language, address facilitator, gestures
- **Task**: amount of input, rate of input, quality of input, quality of output, shared understanding, fit in relations in output
- **Time**: progress, time left

Some of these aspects are non-digital and based partially on interpretations. This requires a *translation* to gain the same insights from the online interaction. For instance facilitators might monitor de-focus of participants as an indicator that the group is finished with the task. However, this might also be learned from a significant decrease in input rate. However, without technology support to monitor input rate, perhaps per participant, this would be difficult to detect for a facilitator. Also the interpretation of input rate requires some experience and understanding of the cognitive implications of tools and knowledge sharing. Therefore we need more than a *thermometer* to measure input rate, we need an intelligent collaboration support system that can monitor these factors, and use them to reason about the current collaboration process in order to support facilitators in making intervention decisions.

4. LEAFHOPPER FACILITATION INTERVENTIONS

In the textbox below we describe what a facilitator does after initiating the LEAFHOPPER thinkLet to brainstorm ideas in categories. Underlined are those indicators the facilitator uses to make decisions on interventions. Some of these indicators can directly be observed, others are an interpretation of the facilitator.

After initiating the Leafhopper the facilitator needs to maintain several rules. The contributions of the group need to meet the quality intended, they need to meet the contribution specification, the category in which they are placed. The contributions need to be made in a certain timeframe, and they need to cover a certain scope of information (completeness). Additionally the facilitator will need to maintain a safe and respectful atmosphere to ensure that people feel free and encouraged to participate. To ensure that the participants can share all relevant contributions, the facilitator can add

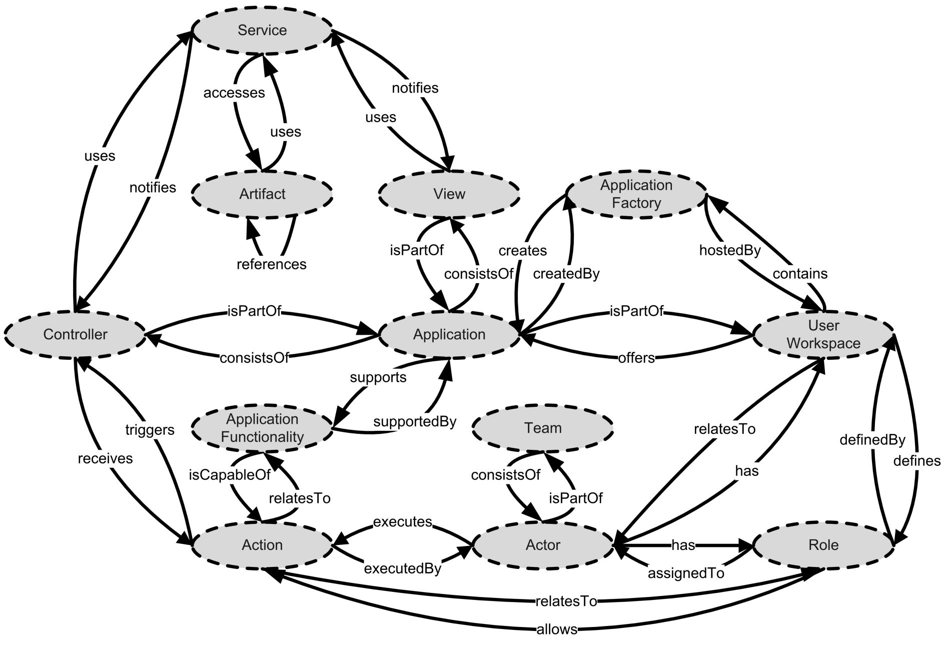

Figure 1: Domain model for collaboration in a shared workspace

a category 'other'. This category is monitored by the facilitator. When a pattern of contributions can be found in this category, the facilitator will add a new category to cover this topic.

The facilitator will monitor the input, mainly to detect if there are **small or insufficient quality contributions**. Later in the process the facilitator will monitor if the categories each contain a **sufficient number of contributions**. Also, the facilitator will monitor the 'other' category to **see if there is a persistent topic addressed**, and therefore, a need to add a category. The facilitator might intervene if some **categories are not filled**. Such intervention would be made **before the time for the task is passed**, to give participants time to add ideas in these categories, but not **too early**, when participants might not yet had a chance to contribute to all categories. The facilitator will also observe the group to see if **participants get distracted**, or **focus on other activities**, which indicate that they are (no longer) motivated for the task. The facilitator will also monitor behavior, communication and body language to see if any of the input causes an **emotional reaction**, which could indicate conflict or flaming, which would require intervention. Finally the facilitator will monitor the **input rate** and the **focus of participants** to detect when there is no more inspiration and the task can be ended. If the group is still very **active** and **focused** when **time is running out**, the facilitator might encourage the group to speed up or to focus on more important contributions in order to ensure that **sufficient progress** is made when the task should be finished. In some cases this can also be a reason to give the group more time for the task.

5. DESIGNING INTELLIGENT COLLABORATION SUPPORT

In order to create a collaboration support system that can suggest facilitators to make interventions, we use an explicit context model to describe the current collaboration situation. A collaboration situation can be characterized by the configuration of the collaboration environment as well as the state of interaction of the users with the system (e.g., based on interaction history) and the organizational setting (e.g., team structure, roles, tasks). Dey et al. [5]define context as any information used to characterize a situation of an entity where an entity may be any object, person or place providing information about the interaction between a user and an application. With this definition, any information may help characterizing the situation of the interaction's participants because it is part of the context itself. For our purposes, we can narrow this definition so that 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. This context information is necessary to recognize situations which demand a facilitator's intervention (and thus help minimizing the effort needed for adaptation).

We use a collaboration domain model for describing collaboration environments and collaboration situations [7]. Figure 1 summarizes this domain model and shows the basic classes and their relations that can be used to describe collaboration context in a global collaboration space. The domain model intends to capture the basic concepts of collaborative workspaces. It focuses on the technological support for collaborative interaction and does not distinguish different artifact types or task domains. If applied to a certain collaboration environment, it must be extended with concepts matching the specific properties.

The model in Figure 1 distinguishes different concepts that describe collaboration in a collaboration environment and relations between these concepts. We start exploring and explaining the model in Figure 1 with the concept of an *Actor* (see lower part of Figure 1). The domain model assumes that *Actors* are member of a *Team* and have a *Role* defined by the *User Workspace*, as *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 an example consider a chat application which should offer at least two action types: *OpenChat* and *SendMsg*. These two actions would allow users to communicate with each other by opening a chat tool and send messages to each other. Other forms of collaboration such as within a collaborative diagram editor would require to add additional action types in order to specify the application functionality.

As *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 received by the corresponding *Controller* components of the *Application*. An *Application* implements the model-view-controller (MVC) paradigm [12] and consists of *Views* and *Controllers* components. *Views* and *Controllers* use *Services* to access the *Artifacts*. *Artifacts* use *Services* to notify *Views* and *Controllers* about changes. Each *Application* is part of a *User Workspace* and is created by an *Application Factory* which specifies what *Applications* are available within a workspace and how these can be initialized. Finally, the class *Application Functionality* specifies the functionality an *Application* offers, e.g. in relation to communication, shared editing, or awareness.

All above classes are useful to model and store the configuration of a collaboration environment and to capture the current context at runtime. Based on such context information, a collaboration environment is enabled to recognize situations, which demand a facilitator's attention and intervention.

The domain model is abstract and not related to a specific application domain. When considering our example on the LEAFHOPPER thinkLet, we need to extend the model as shown in Figure 2. In order to incorporate the LEAFHOPPER thinkLet, the *Artifact* class, the *Action* class and the *Role* class were extended. Based on this extension, we can now distinguish between participants and the facilitator as well as identify contributions within a category.

Based on the extended domain model, we can suggest process interventions or tool adaptations in order to improve collaborative interaction. One process intervention within our LEAFHOPPER example is triggered when the category 'other' exceeds a specified threshold. The following rule consists of a condition and an action block. The condition block retrieves all contributions within the context model that belong to the category 'other' and then evaluates whether the number of contribution has exceeded a specified threshold. If this is the case, the action block opens an alert view for the facilitator. The following pseudo code shows how such a rule can be specified:

```
rule "create new category"
when
  $contributions: Contribution(category ==
    'other')
  eval($contributions.size() >= 20)
 then
  openForFacilitator(Alert, "Number of
                     contributions in
                     category 'other' has
                     exceeded specified
                     limit. Check whether
                     new category is
                     necessary.")
end
```

Another example for a rule that monitors whether there are empty categories and in case again alerts the facilitator can specified as follows:

```
rule "empty categories"
when
  $category: Category(size == 0)
then
  forall $c in $category
    openForFacilitator(Alert, "Category "+
                       $c.name()+" is empty.
                       Focus the attention of
                       the participants on the
                       empty category.")
end
```

As final example, the following rule checks the focus of the participants in order to alert the facilitator when half of the participants do not focus on the activity of creating contributions. For that purpose, the rule retrieves for focus of each participant by identifying the active *View* in the *User Workspace*. Based on the basic collaboration model (cf. Figure 1), this information can be inferred via the *User Workspace* and the opened *Applications* within the workspace. The following example rule assumes that the participants should focus on a view with the name 'contribution input' and if they do not do so alerts the facilitator:

```
rule "participants distracted"
when
  $participants: Participant(focus !=
                 "contribution input")
  $threshold: $participants[0].team().
              size()/2
  eval($participants.size() >= $threshold)
then
  openForFacilitator(Alert, "More than 50%
                     of the participants do
                     not focus on creating
                     contributions.")
end
```

6. DISCUSSION AND CONCLUSION

Collaboration has become a critical success factor for many organizations, as products and services are becoming increasingly complex and cannot be designed individually. However, collaboration has several challenges. It is essential to create a shared understanding and facilitate interaction in such a way that effective collaboration becomes possible. Free riding, group think or hidden agendas need to be addressed by rarely available process facilitators. Available tools are not regularly used, are not intuitive and often are difficult to adapt to the changing group needs. In order to tackle the above issues, we want to enable effective collaborative design and engineering by offering intelligent collaboration support that supports facilitators of collaboration processes when monitoring collaboration processes and planning process interventions or tool adaptations.

In this article, we identified several factors that are observed by professional facilitators before changing and adapting an ongoing collaboration process. We further introduced an abstract context model which can be used to model collaboration within a shared workspace. We extended this model to include concepts and classes of the LEAFHOPPER thinkLet. Based on the experiences of a professional facilitator, we used this extended context model to define rules which can assist a facilitator.

Based on the proposed rules, a context-adaptive and intelligent collaboration support environment, such as [17], can alert a facilita-

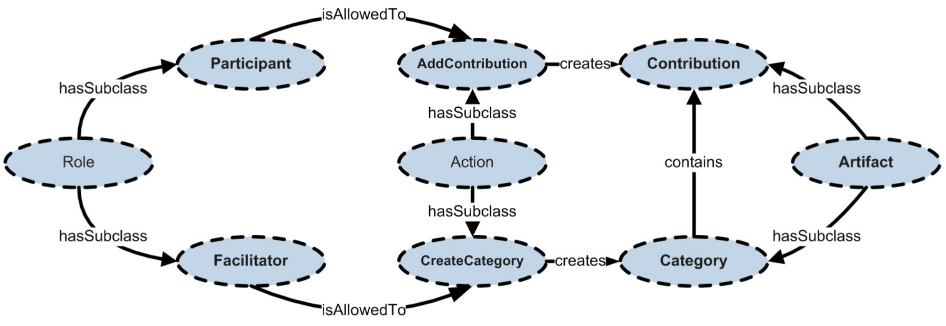

Figure 2: Extended domain model for the LEAFHOPPER thinkLet

tor when an intervention might become necessary and reduce the facilitator's overhead. In future work, we will go a step further and model entire collaboration processes based on thinkLets [10]. We then will study factors that determine and influence collaboration performance, e.g. cognitive load or shared understanding, and that inform facilitation interventions. Once identified, we will integrate these factors in our context model and think about possibilities to measure soft factors via additional application functionality and without obstructing or distracting the group work, e.g. by offering means for self reporting. We will further identify and specify rules that recognize situations that require process interventions by recording facilitation interventions and the performance indicators at the point of intervention to gain more fine-grained rules for process intervention.

The path to intelligent collaboration support sketched above is long, but small steps might already improve collaborative design and engineering today. As facilitators need to monitor many factors and indicators of progress, basic suggestions for process intervention as outlined above might already reduce some of the cognitive load of the facilitation task.

7. REFERENCES

[1] F. Ackermann. Participants perceptions on the role of facilitators using group decision support systems. *Group Decision and Negotiation*, 5:93–519, 1996.

[2] N. Badler, R. Bindiganavale, J. Bourne, M. Palmer, J. Shi, and W. Schuler. A parameterized action representation for virtual human agents. In *Workshop on Embodied Conversational Characters*, 1998.

[3] R. Briggs, G. de Vreede, and J. Nunamaker. Collaboration engineering with thinklets to pursue sustained success with group support systems. *Journal of Management Information Systems*, 19:31–63, 2003.

[4] R. Briggs and G.-J. de Vreede. *ThinkLets: Building Blocks for Concerted Collaboration*. Delft University of Technology, Delft, The Netherlands, 2001.

[5] A. K. Dey, G. D. Abowd, and D. Salber. A conceptual framework and a toolkit for supporting the rapid prototyping of context-aware applications. *Human-Computer Interaction*, 16(2, 3, & 4):97–166, 2001.

[6] J. Fjermestad and S. Hiltz. A descriptive evaluation of group support systems case and field studies. *ournal of Management Information Systems*, 17:115–159, 2001.

[7] J. M. Haake, T. Hussein, B. Joop, S. Lukosch, D. Veiel, and J. Ziegler. Modeling and exploiting context for adaptive collaboration. *International Journal for Cooperative Information Systems (IJCIS)*, 19(1-2):71–120, 2010.

[8] S. Hayne. The facilitator's perspective on meetings and implications for group support systems design. *Database*, 30(3-4):72–91, 1999.

[9] G. Kolfschoten and G. de Vreede. A design approach for collaboration processes: A multi-method design science study in collaboration engineering. *Journal of Management Information Systems*, 26:225–256, 2009.

[10] G. Kolfschoten, S. Lukosch, and M. Seck. Modeling collaboration processes to understand and predict group performance. In A. Dix, T. Hussein, S. Lukosch, and J. Ziegler, editors, *Proceedings of the IUI workshop on Semantic Models for Adaptive Interactive Systems (SEMAIS) 2010*, 2010.

[11] G. Kolfschoten, F. Niederman, G. de Vreede, and R. Briggs. Roles in collaboration support and the effect on sustained collaboration support. In *Hawaii International Conference on System Science (HICSS-41)*, 2008.

[12] G. E. Krasner and S. T. Pope. A cookbook for using the model-view-controller user interface paradigm in Smalltalk-80. *Journal of Object-Oriented Programming*, 1(3):26–49, Aug. 1988.

[13] S.-Y. Lu, W. Elmaraghy, G. Schuh, and R. Wilhelm. A scientific foundation of collaborative engineering. *CIRP Annals - Manufacturing Technology*, 56(2):605 – 634, 2007.

[14] F. Niederman, C. Beise, and P. Beranek. Issues and concerns about computer-supported meetings: The facilitator's perspective. *Management Information Systems Quarterly*, 20(1):1–22, 1996.

[15] F. Niederman, G. de Vreede, R. Briggs, and G. Kolfschoten. Extending the contextual and organizational elements of adaptive structuration theory in GSS research. *Journal of the Association for Information Systems*, 9(10), 2008.

[16] J. J. Nunamaker, R. Briggs, D. Mittleman, D. Vogel, and P. Balthazard. Lessons from a dozen years of group support systems research: A discussion of lab and field findings. *Journal of Management Information Systems*, 13:163–207, 1997.

[17] D. Veiel, J. M. Haake, and S. Lukosch. Facilitating team-based adaptation of shared workspaces. In *International Symposium on Collaborative Technologies and Systems (CTS 2010)*, pages 275–284. IEEE, 2010.