Options for evaluating context-based adaptations of an adaptive collaboration environment

Joerg M. Haake
Fakultät für Mathematik und Informatik
FernUniversität in Hagen
58084 Hagen
Germany
joerg.haake@fernuni-hagen.de

1. Introduction
In modern organizations work is to a large extent collaborative. Especially knowledge work is increasingly performed by distributed teams cooperating across large, often global distances. While collaboration has become ubiquitous, several challenges need to be addressed by cooperation support systems for supporting it effectively. These challenges arise from a variety of features that are characteristic for collaboration. Among other aspects, collaborative tasks are often ill-structured, emerge in the course of the collaborative process, and need to respond flexibly to changing goals or situations. Users participating in a collaborative project may find themselves in different physical environments or settings and may use a variety of different devices. Also, users are often involved in more than one project at a time, raising the need for frequent task or tool switches and for rapid cognitive adjustments to the work at hand.

One approach to deal with the above problem is context-based adaptation of the collaboration environment (Haake et al., 2010; Veiel, Haake & Lukosch, 2009). Here, a context model capturing the current state of the collaboration is maintained and used to select and execute adaptation rules appropriate to improve collaboration in the current situation. Current approaches to evaluation of collaborative systems are mostly based on empirical studies (e.g. experimental studies, field studies, ethnographic studies) and using qualitative data. Such approaches have some problems in this case: Firstly, these methods cannot be applied at runtime to assess the quality of adaptation, identify deficiencies and overcome those by improving deficient adaptation rules. Secondly, these methods require extensive human work and cannot easily be automated. Thirdly, it is an open issue which measures can or should be used to assess the quality of adaptations.

In this paper, several options for evaluating context-based adaptations are discussed.

---

1 This paper was presented at the Workshop “Evaluation Adaptiver Systeme” held at the Mensch & Computer Conference 2010, Universität Duisburg-Essen, Duisburg, September 12-15, 2010.
2. Objects of evaluation

A context-based adaptive collaboration environment (Haake et al., 2010; Veiel, Haake & Lukosch, 2009; Hussain & Haake, 2010) consists of the following components:

- **Process**: defines how members of a group/organization collaboratively deal with understanding adaptations, defining adaptations (i.e. define rules based on change requests, annotations), modification of adaptations (i.e. improve the rule base based on change requests, annotations), and handling exceptions (i.e. undo-redo) (cf. Hussain & Haake, 2010);

- **Runtime adaptation environment**: supports both, the execution of the adaptation cycle (monitor state – evaluate rules – apply rules), and the execution of the above process model;

- **Adaptable applications**: provide an adaptation and state sensing interface to the runtime adaptation environment so that context can be constructed and maintained, and adaptations can be performed;

- **Rule base**: contains the current set of adaptation rules; and

- **Stakeholders**: End-users, Developers (e.g., rule designers).

Figure 1: Adaptive collaboration environment

The aim of a context-based adaptive system is ultimately to support better work of its users (i.e. better task performance). It reaches this goal by

- providing “better” interaction means/affordances matching the needs of the collaboration situation at hand (i.e. adaptation removes obstacles or provides affordances);

- collaboration policies, encoded in adaptation rules, express the organization’s agreement how “better” interaction means/affordances matching the needs of the collaboration situation at hand can be achieved;

- applying suitable collaboration policies/adaptation rules better interaction is facilitated or enabled as both (1) collaborators following such behavior meet their colleagues
expectation and support more fluid collaboration, and (2) the system providing matching support and interaction capabilities for such behavior and interaction;

- leading to better support for the work/tasks of its users.

As a means for assessing whether an adaptive system helps to reduce the gap between situational needs and system behavior, one may use the concept of “expectation conformance”. Expectation conformance measures to what degree users understand and anticipate system behavior. A high degree of expectation conformance means that the mental model of the users match actual system behavior. A low degree means that users expect the system to work differently, thus leading to surprises, anger, and suboptimal performance. One may improve this situation either by correcting the user’s mental model, or by adjusting system behavior.

In a collaborative system, where users collaborate via computer-mediated communication, users are confronted with system behavior caused either entirely by the system's programming or by other users' activities. Thus, a mismatch indicated by low expectation conformance may also be due to unexpected collaborator activities indicating a social conflict or insufficient application of the team’s collaboration policies. Due to the nature of CMC such alternatives are difficult to discriminate. We may thus assume that any activity indicating low expectation conformity or a mismatch is caused by either system deficits or insufficient policies or inappropriate adherence to policies, all to be addressed by the team.

According to Engelbart (1992) we can categorize activities of users as

- A-level activities: activities that are concerned with reaching the tasks or goals of an organization;
- B-level activities: activities that are concerned with improving the way of reaching the tasks or goals of an organization with A-level activities. These activities describe how working in the organization is improved; and
- C-level activities: activities that are concerned with improving the way of improvement with B-level activities (i.e. improving the way to improve the way work is performed).

For assessing context-based adaptive systems, a distinction between A-level und B-level seems sufficient. On the one hand, we consider operations on artifacts related to the task domain of the users as A-level activities. E.g., software engineers create, modify, delete, discuss, test etc. software artifacts (e.g. classes, methods) and the respective objects and operations of the context-adaptive system are considered A-level objects and operations. On the other hand, we consider operations on artifacts related to context, adaptation rules, and their application as B-level activities, as they aim at improving the collaboration between the users. E.g., software engineers may undo adaptations, redo them, create change requests for adaptation behavior and may be involved in rule discussion and revision, and the respective objects and operations of the context-adaptive system are considered B-level objects and operations. It is these B-level activities that can serve as indicators for mismatch respectively low expectation conformity.
3. Alternative approaches for evaluating adaptations

Following the above discussion, successful adaptations can, in principle, be measured by

- assessing the quality of the interaction respectively work performance before and after an adaptation; or
- assessing the quality of adaptations by user cognitive models; or
- analyzing the occurrence of B-level activities; or
- assessing the acceptance of an adaptation in its user community (assuming that beneficial adaptations are accepted while adaptations with negative consequences are rejected); or
- assessing the quality of the process used to deal with adaptations and adaptation improvement.

3.1 Assessing the quality of the interaction before and after an adaptation

The first alternative provides an objective measure for improvement. Improvement can stem from better interaction leading to better product quality or decreased effort, thus increasing efficiency. Unfortunately, assessing the quality of interaction requires a normative interaction model representing “good” or “optimal” interaction against which a real interaction log can be compared. A few examples show that there seem to be no global indicators or process models providing “good” or “optimal” interaction in all cases. E.g., while in requirements engineering a low number of conflicts may be seen as negative (indication for a dead community), in consensus-based decision making a low number of conflicts may be seen as positive.

3.2 Assessing the quality of adaptations by user cognitive models

Goals, operators, methods and selection rules (GOAMS) is a method to predict human behavior using the Human Processor Model (Card, Moran & Newell, 1983). López-Jaquero et al. proposed an approach to assess the user interface adaptations for single user interactive systems (López-Jaquero, Montero & González, 2009) using migration cost and adaptation benefit. Migration cost is defined by the work discontinuity and cognitive load to the user caused by the adaptation. Adaptation benefit is defined in terms of match to the user preferences, frequency of adaptation and user feedback of the adaptation.

In a context-based adaptive collaboration environment, a similar GOAMS analysis can be done considering the team characteristics (e.g., cumulative cognitive load, match to team preferences etc.) along with aforementioned user characteristics.

3.2 Analyzing the occurrence of B-level activities

B-level activities in an adaptive collaboration environment are concerned with improving the way of performing work in terms of sequences of A-level activities. In an adaptive environment, such improvements stem either from adaptations (i.e. the system improves the way work is performed by applying changes encoded in adaptation rules) or from improving adaptations (i.e. users improving adaptation rules or undoing negative adaptations). Thus, the extent of B-level activities can be seen as an indication of how good the adaptations match the expectations of the users, and how much potential for improvement is presently indicated by the users feeling that their B-level activities are needed.
Since every operation takes effort, and users have only limited resources in terms of time and attention, every B-level activity they perform reduces the extent to which they can perform A-level operations needed to achieve their organizational tasks and goals. Thus, we have a zero sum situation, where A- and B-level activities compete. While A-level activities can be seen as “productive” (in the sense of contributing to perform the domain work), B-level activities contribute to improving collaboration and thus may improve productivity. Thus, one cannot simply say that an organization should “optimally” spend 100% on A-level activities to be “optimally” productive (since it may miss improvement opportunities). Likewise, one cannot simply say that an organization should “optimally” spend 100% on B-level activities to be “optimally” productive (since though it may improve their way of work they would, in the meantime, get no domain task done). In reality, one would like to see “needed” B-level activities to be performed while investing the majority of work in A-level activities.

Figure 2: Sample plot of A- and B-level activities of a user over a time interval

Figure 2 shows an example of a possible distribution of activities of a user over a time interval. B-level activities include undo, redo, change requests on rules, and participation in rule discussion and revision. In the example shown in figure 2, the number of A-level activities is in each period higher than the number of B-level activities. In fact, not in every period did the user perform any B-level activities at all. Aggregating such plots for all users leads to a cumulative graph showing the absolute relationship between A-level and B-level activities. Phases with high number of B-level activities indicate that here users work on improvement of adaptation rules while phases with A-level activities only indicate phases of sole production work (without disturbances, e.g. caused by the adaptive system or by external events).

By adapting artifacts, such as adaptation rules and tool configurations, we aim at changing the practice of teams through the change of affordances resulting from changing tool configuration and through negotiation respectively joint construction of rules encoding team policies. The cognitive hysteresis model of DeLone & McLean (1992) describes the uptake of new practices by users as requiring a high gain before users switch to new, profitable practices (and staying with them even when switching to the older way would be better). We argue that automatic adaptation may help team to switch faster, as adaptation is performed without user initiative, while still leaving users in control of defining and improving their policies.
This overall view on A- and B-level activities ignores relationships between B-level operations. Semantically, B-level activities conform to the state transition diagram shown in Figure 3. Using these dependencies, episodes of B-level activities belonging to the same adaptation respectively adaptation rule execution can be defined on the activity log. Whenever an adaptation activity is found in the log, one can search for subsequent activities conforming to the state transitions defined in figure 3.

![State transitions due to adaptations and following B-level activities following the process defined in (Hussain & Haake, 2010)](image)

Figure 3: State transitions due to adaptations and following B-level activities following the process defined in (Hussain & Haake, 2010)

This observation leads us to the definition of different types of B-level episodes:

- **open adaptation episodes**: indicated by a sequence of “start state – adapted state” (adapt operation sequence, respectively). These indicate situations, where an adaptation was performed and nobody (yet) objected.

- **closed adaptation episodes**: conform to open adaptation episodes where nobody objected during the timeout interval. Through the timeout, the timeout state was reached and, possibly, positive rule feedback was added to the rule before reaching the end state concluding this episode.

- **open exception episodes**: indicated by a sequence of “start state – adapted state – exception state” (adapt – undo operation sequence, respectively). These indicate situations, where an adaptation was undone and nobody (yet) objected.

- **closed exception episodes**: conform to open exception episodes where nobody objects during the timeout interval. Through the timeout, the timeout state was reached and, possibly, rule feedback was added to the rule before reaching the end state concluding this episode.

- **open change request exist episode**: some change request on this rule application exist (i.e., the state sequence ends with change request exists state), but rule revision did not yet start. When the revision process starts, this episode becomes an “open under revision episode” (see below).
• **open under revision episode**: rule revision started (i.e., the state sequence ends with under revision state), but rule revision did not yet end. More operations are to follow in this episode in this future.

• **closed under revision episode**: rule revision was closed before reaching the end state concluding this episode.

These types of episodes may help us in limiting the number of activities to be analyzed in the activity log and in recognizing positive or negative situations:

- Closed episodes are, by definition, not subject to further activities. Thus, the episode recognition component can safely ignore the activities already assigned to closed episodes, and does not need to examine closed episodes as candidates for assigning further activities to them. The reason of using a timeout mechanism is exactly to limit the number of open episodes, where users do not perform further activities. Thus, the number of open episodes and unassigned activities is rather limited.

- Productive use of adaptation may be characterized by having only a limited number of open episodes, closing episodes relatively fast (limiting time devoted to B-Level episodes) and having large numbers of positively evaluated adaptations (represented by open and closed adaptation episodes).

- Unproductive use of adaptation may be characterized by having large numbers of open episodes, especially open change request exist episodes and open under revision episodes, as here users explicitly asked for change and need to agree on future policy. Large numbers of such episodes indicate that adaptation rules/policies are deficient, the users have not yet reached a common understanding and agreement on what such policies should look like, or that the environment is changing and modification of adaptation rules/policies is needed. Alternatively, the revision process or resources dedicated to its execution may be insufficient.

In terms of a learning curve, one would expect the following three phases to appear:

1. Initially, or after new team members join, or after some change happened, rules are more often regarded as not meeting the needs of the users. Thus, they are more often undone, annotated with feedback, and changed. Consequently, more open episodes exist, especially “open exception episodes”, “open change request exist episodes” and “open under revision episodes”. Due to the new users or system, users need more time in learning and applying system features, leading to longer episodes and possibly longer undo wars (toggling between adapted state and exception state).

2. After a while, improved or added rules should lead to higher acceptance. Still, exceptions lead to some undo, but the number of changes and negative feedback decreases. However, some negative feedback indicating further improvements remain. This leads to less open episodes, episodes become in average shorter, and fewer “open change request exist episodes” and “open under revision episodes” are created.

3. Eventually, the team/group has agreed on proven policies and performs work accordingly. Few undo operations point to rare exceptions that are not worthwhile to address in changes. The team works smoothly with a suitable system. Here, we see much less exception, change request and revision episodes, and such episodes should become shorter in average (compared to the previous phases).
One problem of any optimization is whether or how local optima are overcome? In our approach, one could inject policies from other teams at appropriate times (e.g., when work runs "too" smoothly?). Alternatively, one could argue that the environment is constantly changing, and thus sufficient pressure for continuing improvement is pertained to the system. Finally, one may also argue that change induces cost, and thus sometimes a local optimum may be advantageous compared to the cost-benefit ratio of further change.

3.3 Assessing the acceptance of an adaptation in its user community

This alternative examines neither output nor B-level sequences, but only interprets certain communication acts as indicators for acceptance or rejection of adaptations. Here, the following indicators seem useful to indicate acceptance of an adaptation:

- **individual level**
  - **acceptance quotient** $\text{accept}(\text{user, rule, t1, t2})$: number of undo +1 divided by number of executions +1 per rule in a given time interval
    
    $\text{accept}(\text{user, rule, t1, t2}) > 0$ means that the user rejected the majority of the adaptation occurring in the time interval
  - **feedback quotient** $\text{feedback}(\text{user, rule, t1, t2})$: number of negative feedback +1 divided by number of positive feedback +1 per rule in a given time interval
    
    $\text{feedback}(\text{user, rule, t1, t2}) > 0$ means that the user rated the majority of the adaptation occurring in the time interval positively
  - **change function** $\text{delta_change}(\text{user, rule, t1, t2})$: the function showing the cumulative number of actions in a revision cycle following an undo of the rule during the time interval
    
    $\text{delta_change}(\text{user, rule, t1, t2}) = 0$ means that no revision cycle is performed (either indicating that it is not yet started or that it is not needed)
    
    $\text{delta_change}(\text{user, rule, t1, t2}) > 0$ means that a revision cycle is performed
  - **exception function** $\text{delta_exception}(\text{user, rule, t1, t2})$: the function showing the number of undo of this rule during the time interval, which are not (yet) followed by a revision cycle
  - **user feedback** $\text{user_feedback}(\text{user, rule, t1, t2})$: the set of feedback forms on rule provided by the user during the time interval.
    
    $\text{size}(\text{user_feedback}(\text{user, rule, t1, t2})) > 0$ means that some feedback form was filled by the user on this rule during the given time interval

- **team or group level**
  - **acceptance quotient** $\text{accept}(\text{team/group, rule, t1, t2})$: number of undo +1 divided by number of executions +1 per rule in a given time interval.
    
    $\text{accept}(\text{team/group, rule, t1, t2}) > 0$ means that the users in the team/group rejected the majority of the adaptation occurring in the time interval
  - **feedback quotient** $\text{feedback}(\text{team/group, rule, t1, t2})$: number of negative feedback +1 divided by number of positive feedback +1 per rule in a given time interval.
    
    $\text{feedback}(\text{team/group, rule, t1, t2}) > 0$ means that the users in the team/group rated the majority of the adaptation occurring in the time interval positively
- **change function** \( \text{delta_change}(\text{team/group, rule, t1, t2}) \): the function showing the cumulative number of actions in a revision cycle following an undo of the rule during the time interval.

\[ \text{delta_change}(\text{team/group, rule, t1, t2}) = 0 \] means that no revision cycle is performed (either indicating that it is not yet started or that it is not needed).

\[ \text{delta_change}(\text{team/group, rule, t1, t2}) > 0 \] means that a revision cycle is performed.

- **exception function** \( \text{delta_exception}(\text{team/group, rule, t1, t2}) \): the function showing the number of undo of this rule during the time interval, which are not (yet) followed by a revision cycle.

- **user feedback** \( \text{user_feedback}(\text{team/group, rule, t1, t2}) \): the set of feedback forms on rule provided by the users from the team/group during the time interval.

\[ \text{size}(\text{user_feedback}(\text{team/group, rule, t1, t2})) > 0 \] means that some feedback form was filled on this rule during the given time interval.

### session level

- the above indicators defined for sessions

- open issue: definition of session
  - one tool with its users beginning when 2\(^{nd}\) user joins and ending when 2\(^{nd}\) last user leaves?
  - Could be defined within an adaptation rule? Could include a start new session tag, but how to detect the end? Which tools and their users would be included?

In terms of a learning curve, one would expect the following three phases to appear:

1. **Initial**: Initially, or after new team members joined, or after some change happened, rules are more often regarded as not meeting the needs of the users. Thus, they are more often undone, annotated with feedback, and changed.

2. **Intermediate**: After a while, improved or added rules should lead to higher acceptance. Still, exceptions lead to some undo, but the number of changes and negative feedback decreases. However, some negative feedback indicating further improvements remain.

3. **Flow**: Eventually, the team/group has agreed on proven policies and performs work accordingly. Few undo operations point to rare exceptions that are not worthwhile to address in changes. The team works smoothly with a suitable system.

Note, that after external change requiring change of collaboration policies, we would expect the team/group to start with another cycle on a lower level (suggesting that not all proven policies need to be changed, but some change may be required).

Success in terms of acceptance could then be indicated by reaching indicator values appropriate to the phase of the team/group. Identification of such thresholds could be performed on the basis of interaction analysis of real collaboration cases.
<table>
<thead>
<tr>
<th>Phase</th>
<th>Individual</th>
<th>Team/group</th>
</tr>
</thead>
</table>
| initial| lower acceptance quotient  
lower feedback quotient  
increasing change_function (as users learn how to use it)  
users make increasingly use of exceptions  
users make increasingly use of feedback forms | lower acceptance quotient  
lower feedback quotient  
increasing change_function  
high number of exceptions  
increasing number of feedback forms |
| intermediate | increasing acceptance quotient  
increasing feedback quotient  
moderately decreasing change_function  
slowly decreasing number of exceptions  
negative feedback forms decrease | increasing acceptance quotient  
increasing feedback quotient  
moderately decreasing change_function  
slowly decreasing number of exceptions  
negative feedback forms decrease |
| flow   | positive acceptance quotient  
positive feedback quotient  
low change_function  
low number of exceptions  
low number of negative feedback forms | positive acceptance quotient  
positive feedback quotient  
low change_function  
low number of exceptions  
low number of negative feedback forms |

Note, that after external change requiring change of collaboration policies, we would expect the team/group to start with another cycle on a lower level (suggesting that not all proven policies need to be changed, but some change may be required).

Success in terms of acceptance could then be indicated by reaching indicator values appropriate to the phase of the team/group. Identification of such thresholds could be performed on the basis of interaction analysis of real collaboration cases.

3.4 Evaluation of the process
The process model used to deal with adaptation and improving them could be validated by performing walk-through with test users. During such a walk-through, users are led through a predefined scenario illustrating the proposed process. By semi-structured interviews user feedback and assessment on the proposed process is elicited.

If a mock-up is used in this process, some feedback on usability issues can be collected, too.

Note, this is a validation based on user assessment, not based on showing that the proposed process in reality leads to better outcome or efficiency. If objective measures are to be used, a controlled experimental procedure could be used, where an experimental condition is compared to a control condition. However, while such procedures have higher internal validity they suffer from potentially lower external validity (i.e., one cannot be sure that the effect is also showing in the field, where other factors neglected in the lab study may turn out to overpower the lab effects).
4. Evaluation of the runtime adaptation environment

The success of adaptations is also subject to influences from the runtime adaptation environment used to perform adaptations. The runtime adaptation environment can be evaluated by:

- functional tests showing that the system can actually support the use cases or scenarios used in the evaluation of the process model;
- usability tests showing that the system is usable; and
- field studies with acceptance tests (see above) where:
  - interaction logs of real usage are collected and analyzed regarding the above interaction measures. If the hypotheses are backed by the data, one could draw conclusions regarding the learning curve and phase model.
  - subjective feedback is collected indicating acceptance.

5. Conclusions

In this paper, the issue of how to evaluate context-based adaptations in collaboration environments was discussed. An analysis of adaptive collaboration environments led to four approaches:

- assessing the quality of the interaction respectively work performance before and after an adaptation based on global performance criteria for group work or collaboration behavior was deemed to be inapplicable;
- analyzing the occurrence and episodes of B-level activities based on a process specifying how adaptations are applied and improved led to the definition of three phases during adaptation of collaboration environments for teams, and the definition of corresponding predicates useful to identify problem areas;
- assessing the acceptance of an adaptation in its user community (assuming that beneficial adaptations are accepted while adaptations with negative consequences are rejected) based on indicators for individual and group acceptance of an adaptation. It is suggested that such indicators can be used to identify the state of collaboration (initial, intermediate, flow) and to help identify cases where more adaptation is needed;
- assessing the quality of the process used to deal with adaptations and adaptation improvement based on the walk-through method or using other types of empirical studies were discussed.

In addition, the evaluation of the core component of a context-adaptive system was discussed: the runtime adaptation environment could be evaluated using functional tests, usability tests, and field studies.

While the current state of this discussion is at the conceptual level, the next steps include the specification of concrete measures and thresholds based on case data, and subsequently implementation of the evaluation methods and its testing.
6. Acknowledgements

Thanks are due to the members of the Contici project at FernUniversität in Hagen, Dirk Veiel, Sajid Hussain and Stephan Lukosch, for many inspiring discussions. The Contici project was funded by the DFG under grant number HA 3130/4-2.

7. References


