Facilitating Understanding of Team-based Adaptation Policies

Syed Sajid Hussain, Dirk Veiel and Joerg M. Haake
Faculty of Mathematics and Computer Science
FernUniversität in Hagen
58084 Hagen, Germany
Email: {sajid.shah, dirk.veiel, joerg.haake}@fernuni-hagen.de

Stephan Lukosch
Faculty of Technology, Policy, and Management
Delft University of Technology
PO box 5015, 2600 GA Delft, the Netherlands
Email: s.g.lukosch@tudelft.nl

Abstract—Knowledge work in modern organizations is increasingly performed collaboratively by distributed teams. Such collaboration involves dynamically changing situations. Making the collaboration environment fit the needs of these situations is supported by system-initiated adaptation based on the user and team context. Adaptation in the collaboration environment may confuse the users, because they cannot remember all adaptation policies. In this paper we propose an approach to present context enriched explanations to help them understand the adaptation behavior. Also, we describe a social network analysis strategy to identify socially related and current situation relevant communication partners. Integrated on demand communication is facilitated among these partners for discussion and understanding the adaptation policies.

Index Terms—Context, adaptation, collaboration environment, context enriched explanation, socially related communication partner, process support and the meta-design framework

I. INTRODUCTION

Collaboration in modern organizations has become ubiquitous. Distributed teams solve complex problems using multidisciplinary knowledge and skills of team members. Such collaboration poses enormous challenges: it occurs on the spur and in-situ, is often ill-structured, and involves a variety of dynamically changing situations [1]. Collaboration Environments (CE) provide a variety of tools and services to support communication, coordination and collaboration among team members. However, these tools and services are not coherently integrated to cope with these challenges insofar as users have to manually select tools and services, configure and adjust them as they are required.

The context-based adaptation approach aims to bridge the gap between the actual needs of collaborating end-users and the functionality provided by their CE. Context-based Adaptation and Collaboration Technology (CONTact) [2, 3, 4] is a service oriented context-adaptive shared workspace CE. It uses a generic four layered framework to maintain and exploit user and team context for adaptive collaboration [5]. 1) The knowledge layer captures the knowledge about collaboration and task domains. 2) The state layer maintains the concrete collaboration situation. 3) The contextualization layer filters the most relevant parts of the state with respect to a given set of focus objects. 4) The adaptation layer maintains the adaptation policies and the resulting adaptation state.

CONTact provides a framework and a runtime system allowing context-adaptive applications to register to it, and uses context information to adapt. Adaptations involve reconfiguring application affordances and interaction behavior to improve collaborative work among end-users. Such reconfiguration (governed by adaptation policies) may cause modification to UI, to the application logic, to the services, and/or to the shared model of the CE [6].

End-users facing system-initiated adaptation may be confused because they (especially less experienced users) cannot remember all adaptation policies. Therefore, as prescribed by the meta-design framework [7], integrated and situated system support is needed to help them understand the adaptation behavior. In [8] we have proposed a theoretical process model guiding the interaction among end-users when dealing with context-based adaptations in CONTact CE. This process model guides us as designers of CE about the in-use state of adaptations, thereby providing requirements for supporting end-users consuming and improving adaptations.

In this paper, we present an approach to implement context enriched explanations to help avoid confusion and to facilitate understanding adaptation policies in CONTact CE using the above process model. Also, as suggested by the process model, the level of understanding varies among users involved in a team-based adaptation. Therefore, we describe an approach of social network analysis to identify situation relevant and socially related communication partners. Explanations include hyperlinks to such communication partners to help address further need for on demand discussion, clarification and learning from one another.

The rest of this paper is organized as follows: In section II, we analyze the problem to identify requirements and section III reports the related work. Section IV presents our approach of extending the domain model to present context enriched explanations and means of communication with relevant partners. In section V, we discuss our approach and finally section VI concludes this paper summarizing the contributions.
II. Problem Analysis

In the following, we use a sample scenario to describe how users can get confused after a team-based adaptation has been applied to the CE. We use the process model to identify the requirements to avoid such confusions and to facilitate understanding adaptations.

A. Scenario

Figure 2 shows the organization chart of our Sample Company. Alice, Bob, and Dave are members of our Project Team. Alice is the head of the Development Department and manages the project. Bob is responsible for the more technical part of the project and, because of that, he is the personal point of contact for Dave who is an expert in managing networks and their infrastructure. The Project Team has to prepare a product specification and presentation for the customer.

Now assume, few days before a presentation for the customer, Dave updates his part regarding the network topology and the corresponding constraints. Next morning, while driving to work, he recognizes that one important fact is missing. At the office, he opens the shared workspace, where the presentation is stored and notices that it is locked. He gets confused because someone has removed his access rights. Consider there may be multiple adaptation policies applicable to this collaboration state (e.g., shared editing, versioning, read only view, and locking). Also, the presentation may be locked by a user for editing, as well as by the system because of an expired deadline. Therefore, despite knowing that the system can adapt itself to the current collaboration state by applying context-based adaptations, Dave is not able to figure out what adaptation caused the locking of the presentation and why. What Dave does not know is that Alice is presenting the current version of the presentation to Carl of the Sales Department, to get valuable feedback from him. This sample conflict leads to the question, how a system can help avoiding or getting out of this confusion.

B. The Process Model

We have proposed a theoretical process model guiding the interaction among end-users when dealing with context-based adaptations (as shown in Figure 1) [8]. The process model provides end-user support requirements while dealing with adaptations.

An adaptation policy has many stakeholders namely: the users 1) who are affected by this policy and like it, 2) who are affected by this policy, disliked it and added exceptions to it, 3) who defined or revised it, and 4) who initiated or negotiated an amendment request to it. We assume that CONTact is running with some initial adaptation policy set, and users are performing their domain activities.

The process model considers all states of affected stakeholders individually and as a team (shown in Figure 1 as Affected Stakeholder vertical panes and Team horizontal block, respectively) while dealing with an adaptation. Alice
and Dave are effected stakeholders of the adaptation in our scenario. The reader should note that each adaptation performed in the CE will start another instance of the process. This means that at every point in time, users may either perform domain operations or act within one of their running processes dealing with previous adaptations.

When an adaptation occurs and the affected stakeholder (Dave in our scenario) is busy with some domain activity, he needs to be made aware of it. If the stakeholder is aware of an adaptation, he needs support to understand it (to avoid confusion). A stakeholder well experienced with a specific type of adaptation may not require any support for understanding it. If the stakeholder understands a change, and has no objections, it is assumed in our model that he liked it. Moreover, if he didn’t notice it is assumed that he liked it, too.

Support for adaptation understanding is required to help Dave understand the rationale of the changed system configuration (in our sample scenario the removal of access rights). Current situation needs to be considered for generating such explanation.

**Requirement 1:** The CE has to provide situation aware explanations to help understand the rationale of the adaptation.

If the stakeholder thinks it to be only an exception, he should be supported to undo the adaptation, e.g., via an undo button that is available at the UI of all affected stakeholders. Another affected stakeholder, when informed of an undo, may argue that the adaptation was right, therefore he needs to be supported in redoing the adaptation, e.g., via a redo button available at the UI of all affected stakeholders. It follows that in case of an adaptation policy undo/redo, the relevant stakeholders need to be notified. An undo/redo of an adaptation is essentially also an adaptation itself. An undo/redo pattern shows social conflicts among a group of stakeholders. This conflict needs to be identified and socially removed by communication between conflicting stakeholders. They may then be satisfied to end this adaptation process or may decide to initiate an amendment request to revise the adaptation policy. If the stakeholder thinks that instead of making an exception a permanent policy amendment is required, he should be supported to initiate an amendment request. Since an amendment to a policy potentially affects all stakeholders, a consensus on the amendment is needed. This request may then be implemented with the help of policy designers. There are two types of amendment requests namely: reactive and proactive amendment request. The former is an amendment request to an inappropriate policy while the latter is a request of a new policy. Both types of amendment need to be supported differently.

As shown in the Figure 1, when an adaptation process reaches the team block (by a dislike), it then requires involvement of all affected stakeholders insofar as involving all the stakeholders, therefore a confusion of a stakeholder leading to dislike is vitally important here. Also, the affected stakeholders may at a specific time be in different states while dealing with the same adaptation (see the vertical effected stakeholder panes in Figure 1). This shows a varying level of understanding of the adaptation. Therefore, if the situation relevant stakeholders can communicate with each other, they can improve their understanding. Moreover, it is a normal human behavior to discuss freely with socially related communication partners (having social acquaintance). Hence, further discussion can be facilitated by identifying communication partners who are situation relevant and socially related. These communication partners may explain the adaptation policies and in some cases the vocabulary used in the formulation of the situation-aware explanation.

**Requirement 2:** The CE has to identify communication partners for further explanation and discussion. These partners should be a) situation relevant and b) socially related.

### III. Related Work

Grudin in his seminal work argued that UI design has to be adapted to end-users’ domain work context and UI consistency is relative [9]. Empirical studies have shown that users are somehow always confused when facing some adaptation [10]. Explaining the rationale of such adaptation (Requirement 1) increases understanding; builds trust and eventually leads to adaptation acceptance [11]. In Intelligent Tutoring Systems (ITS), explanations are presented to complement learning material [12]. In Knowledge Base Systems (KBS) [13], explanations are presented to justify the system behavior. But these systems normally focus adaptations involving single users. Explanations in Information Recommendation Systems (IRS) tell the users why they might like the recommended information [14]. In ITS, explanations are generated using the learner model, while IRS systems use the community model as well. Morch and Mehandjiev propose the idea of application units and multiple representations to capture and preserve design rationale of system components [15]. Although IRS consider adaptations involving multiple users, but their focus is limited to information recommendation. Hence there is a need to provide explanations of system adaptation behavior using the rationale of system components, user model and current collaboration situation.

Although explanations help understand the adaptation behavior, but the need for discussion and common grounding [16] still remains because of confusing statements or terminology. Such discussions can be meaningful and result oriented if the communication partners are a) situation relevant and b) socially related (Requirement 2).

Synchronous communication channels like Chat and Instant Messaging (IM) are increasingly used in workplace for interactions, broadcasting of information or questions, and negotiations [17]. Pipke calls for integrated discourse-based communication between situation relevant partners [18].

Social relationships ease the communication between people. Enterprises can benefit from the social networking of
their employees [19]. Therefore social network analysis is needed to find situation relevant and socially related communication partners and situated and integrated communication as guided by the Meta-design framework.

IV. APPROACH

In the following, first we briefly introduce the CONTact CE, and identify additional requirements for CONTact addressing Requirement 1 and Requirement 2. Next we present our solution consisting of presenting contextualized explanations and integrated communication to conform to the identified requirements. We extend our domain model and integrate information about the social relations between users (cf. Requirement 6) and their communication details (cf. Requirement 7).

A. CONTact Architecture

In CONTact CE every user has his workspace. Collaborative applications register themselves to the workspace. The applications specify the adaptations they support, and the context information they can provide. Adaptation policies (implemented as rules) specify under which conditions (based on properties of the context) what adaptations are to be performed. The runtime system captures the user and team context by collecting respective information from the applications, and if the condition of a specific adaptation policy is met, it is executed causing, e.g., the applications to adapt.

CONTact uses four-layered framework for context-based adaptations consisting of the knowledge layer, the state layer, the contextualization layer, and the adaptation layer as described in [5]. The knowledge layer contains all relevant conceptual and factual knowledge about the domain. We use a collaboration domain model for describing CE and collaborative situations to manage the context and handle context-based adaptations as described in [20]. The state layer contains information about the current situation including information about the physical and computing environment, resources and the user model. In the contextualization layer, contextualization rules define which subset of the state is currently relevant. Upon this, adaptation policies defined in the adaptation layer are selected. From the set of policies, the relevant policies are identified using the contextualized state.

Requirement 1 can be supported by presenting an explanation that takes the current context into account. To implement this at the CONTact CE, we have to extend the current prototype leading to further requirements:

Requirement 3: The adaptation rule has to support an explanation block.

Requirement 4: The explanation block has to support mechanisms to add valuable information about the current situation to the explanation at execution time.

Requirement 5: CONTact Client has to present the situation-aware explanation on demand.

Requirement 2 leads to the following requirements:

Requirement 6: CONTact has to find possible socially related and relevant partners that may explain the current situation to the confused user.

Requirement 7: CONTact has to offer means of communication with socially related and relevant partners.

B. Extending the Domain Model

We extend the domain model for collaboration by adding the ontology vCard [21]. We use the vCard ontology to address Requirement 7 because it is standardized and supported by different tools (e.g., Microsoft Outlook). An excerpt of Bob’s information in vCard is shown in Figure 3. Additionally, we support a basic implementation of the Friend of a Friend (FOAF) [22] pattern using the user’s buddy list to address Requirement 6. An excerpt of Bob’s FOAF profile (presented in Figure 4) shows that Bob knows Alice and Dave.

Fig. 3. An Excerpt of Bob’s VCard

These two contributions enable us to describe the current collaboration situation of the users, their corresponding contact information. The social relations between them can be calculated by their buddy lists. Figure 5 shows the current contextualized state of the Project Team currently available at the system (i.e. they are online). The knows relation between Alice and Bob, and Bob and Dave are established by using the aforementioned FOAF information. The corresponding vCard information is not shown to preserve the readability. The arrows/relations in light grey will be removed after applying the adaptation rule shown in Figure 6.
C. Context Enriched Explanations

Now, the context representation contains information that enables us to address Requirement 1, i.e. to enrich explanations of applied adaption rules to make user better understand the intentions, terminology, and consequences of the corresponding adaption rule and the system. Currently, an adaption rule consists of a block of conditions to be fulfilled, before the corresponding action block will be executed. The two system variables used in the following adaptation rule (${focus}$ and ${time}$) are initialized before executing it. To support situation-aware explanations we propose to add an explanation block (cf. Requirement 3) that is able to use the bound variables from the conditions block to add situation specific information to the explanation (cf. Requirement 4). The adaption rule shown in Figure 6 is the source of the confusion of Dave in our sample scenario (cf. section 2). After this adaption rule is applied, only the presenter (Alice) has rights to execute actions on the corresponding presentation artifact.

In our scenario, the above adaption rule is triggered by Alice (Actor: alice) executing the action SetPresentationMode:alice on the Artifact:Presentation. The system variable ${focus}$ is initialized with SetPresentationMode:alice before executing the adaption rule. The function getArtifactsInContext returns a set of artifacts which are in the context of the action SetPresentationMode:alice. To be able to handle only artifacts of type Presentation, we apply the function getArtifactsOfType to filter the found artifacts. By executing the function getActorOfAction, we retrieve the actor starting the presentation. The function getActorsInContext then calculates all actors which have access to the presentation artifacts in the context of SetPresentationMode:alice, i.e. in Figure 5 Actor:alice and Actor:dave. To avoid removing the access permissions of the presenter, we remove the presenter from the list of actors.

After all these functions were applied and the returned set of actors is not an empty, the corresponding action block is executed. The function revokeAccessRightsForActions removes the rights for the actors to execute the given actions on the artifact. Applying this to our sample scenario will remove Actor:dave’s permission to execute OpenPresentation and SetPresentationMode on the Artifact:Presentation.

```
RULE “disallow actions on presented artifact”
WHEN
// ${focus} = SetPresentationMode:alice
// ${time} = current timestamp
?artifact: getArtifactInContext(${focus})
?presentations: getArtifactsOfType(?artifacts, “Presentation”)
?presenter: getActorOfAction(${focus})
?actors: getActorsInContext(?presentations) - ?presenter
DO
revokeRightsForActions(?actors, ?presentations, “OpenPresentation”, “SetPresentationMode”)
EXPLANATION
Because ?presenter presents ?presentations at ${time} you are not allowed to open or present it.
END
```

Fig. 6. Adaptation Rule; Disallow Actions on Presented Artifact
The next step of executing the adaptation rule is to create the corresponding explanation using the bound variables from the conditions block. In our sample scenario the corresponding explanation looks like this: “Because Alice presents Presentation at 2010-04-01 09:15:00, you are not allowed to open or present it.” This explanation is presented to the user on demand, i.e. when he or she presses the explanation button of the CONTact client. Figure 7 illustrates the situation after Dave pressed the explanation button.

D. Communication Partners

We address Requirement 2 (The CE has to identify communication partners for further explanation and discussion. These partners should be a) situation relevant and b) socially related) by adding possible communication channels to socially related and in the current situation relevant partners to the explanation view (cf. Requirement 5) that is shown on demand. Creating this information takes the current context representation into account.

We assume that socially related persons add each other to their buddy list (cf. FOAF [22]). Adding someone to the buddy list leads to updates of the context representation by adding knows relations between the corresponding buddies (cf. Requirement 6). As shown in Figure 5, Alice knows Bob, and Dave knows Bob, and Bob knows both of them. In our assumption knows means that the two persons are socially related and are used to talk to each other. Hence, Dave is not directly connected to Alice, because Dave is used to talk to Bob, e.g., about the project, but not to Alice.

The applied adaptation rule triggered by Alice (cf. Figure 5) leads to the confusion of Dave. To help getting Dave out of this situation, we use the context representation to try to find possible socially related and relevant persons that may explain the current situation to him (cf. Requirement 6). Therefore, we try to find a path between the actor causing the confusion and the confused user. As shown in Figure 4, Alice and Dave are not directly connected, but they are related to the same person (Bob). So a path can be built (Alice – Bob – Dave). We suppose that Dave should ask Bob for further explanations because he is directly socially connected to him, and he is in the context of the current situation being in the same Project Team. In the case that Bob is not present Alice should be contacted, because they share the same “buddy”. In worst case, none of them is accessible at the moment, so we suppose that Dave contacts Mark, the adaptation policy designer. Applying these assumptions in the sample scenario the ordered list of persons to be asked looks like this: Bob, Alice, Mark.

Using this ordered list of persons, the explanation view creates and shows the following additional information: “For further explanation and discussion you can contact Bob, Alice, and/or Mark.” To enable Dave to contact these persons, the explanation view uses the context representation to retrieve possible communication channels to them (cf. Requirement 7). When Dave clicks on a person to communicate with, the corresponding available communication channels are presented to him. Applying these mechanisms lead to the explanation view shown in Figure 7 on the right hand side.
V. DISCUSSION

As suggested by the process model (cf. section II-B), we have implemented context enriched explanations to reduce confusion and to facilitate situated and embedded understanding about adaptation policies in the CONTact CE. We use the extended domain model and implemented the corresponding sensing capabilities in our current prototype to address **Requirement 1**. We have extended our adaptation policy set to address **Requirement 3**. Also, we have extended our CONTact runtime system to address **Requirement 2, Requirement 4, Requirement 6** and **Requirement 7**. The explanation view shown in Fig. 5 has been developed and integrated into CONTact CE as an Eclipse Plug-in to support **Requirement 5**. Functional tests demonstrated that the requirements are indeed met. A walkthrough through the sample scenario shows that the approach meets our claims.

A. Walkthrough through the Scenario

Next we present a walkthrough using our sample scenario (cf. section II-A) and the process model (cf. section II-B). The initial contextualized state of Project Team is shown in Figure 5 (with the light grey arrows). As Alice opens the artifact Presentation and sets it to PresentationMode, the adaptation rule “disallow actions on presented artifact” (cf. Figure 6) is executed, which changes the context to Figure 5 (without the light grey arrows). This adaptation (S0) starts an instance of the adaptation process. As Alice is passively affected by this adaptation, she doesn’t notice it, and consequently it is assumed she liked the change and the process model finishes for her following state transitions S0, Sf. The process instance of Alice is shown in Figure 8.

Now Dave tries to open the artifact. He figures out that it is locked; he doesn’t understand it and is at the confused state now following the state transitions S0, S1, S2. The process instance of Dave is shown in Figure 9. A screenshot showing this state is shown in Figure 7 (without the Explanation View). Now he presses the Explain icon. In response to this, the CONTact system computes context enriched explanation (cf. section IV-C) and further information to figure out socially relevant partners for communication (cf. section IV-D). CONTact presents this information to Dave (cf. Figure 7). He discusses with Bob to clarify the situation, he then understands and appreciates the adaptation (following state transitions S2, S3, Sf). As there are no more affected stakeholders, the process finishes.

Without the explanation and socially relevant partner information, Dave would have disliked the adaptation causing either an undo or an amendment request. The undo (being another change) might confuse Alice, possibly leading to a redo from her and a social conflict. If Dave would have initiated an amendment request, it would also add an agenda item in the next review session. Thus, the social filter in the process model saved this whole unnecessary effort. Hence we conclude that our approach helps Dave out of confusion and saves efforts for social conflict handling and saves an extra agenda item in the review session. Dave learns about this policy and this helps him in the future execution of the similar policies. Also, Dave may now reflect about improving this policy in other situations in the future.

B. The Meta-design Framework

In [8] we have reported the results of two pluralistic usability walkthroughs [23] to validate our process model using a mockup. Pluralistic usability walkthrough is a usability inspection method involving users, developers and usability experts. The participants in such walkthrough step through a scenario and discuss the usability issues related to the UI. In these walkthroughs, we have used a script to simulate stakeholder confusion following an adaptation of CONTact CE. We have provided hardcoded general explanations, without any context information. We have
presented explanations to the confused stakeholders facing the adaptation. In the discussions, they have assessed even such a general explanations as helpful. They did however, pointed out that the explanations were lengthy.

In this paper, we present an approach to actually generate these explanations in the CONTact CE. We have learned our previous experience to shorten the explanations. Also, we enriched these explanations with the current context. Our new approach presented above also identifies situation relevant and socially related communication partners for situated and integrated discussions instead of lengthy explanations. The meta-design framework [7] promises informed participation and a better system understanding for end-users. Our argument here is that our approach of presenting context enriched explanations and discussion, conforming to the meta-design framework, improve the end-user understanding about the adaptation behavior of CE [7]. In future, we will conduct more usability studies to actually validate our approach.

VI. CONCLUSION

In this paper we have presented an approach to implement context enriched explanations to help avoid confusion and to facilitate understanding adaptation policies in CONTact CE using the process model described in [8]. We have described a simple approach of social network analysis to identify situation relevant and socially related communication partners. Our approach conforms to the meta-design framework leading to better system understanding for end-users.

Our approach exceeds current approaches by using context information about the current collaboration situation. We support integrated and situated end-user communications to help them understand the adaptation behavior of CONTact CE.

Future work is needed on assessing the impact of our approach in experimental studies as well as in longer-term field studies. Also more work is needed to improve our social network analysis strategy.

REFERENCES


