Towards Traceability of Decision-making in Intelligent Context-based Adaptive System Environments

Mandy Goram
FernUniversität in Hagen
58084 Hagen, Germany
mandy.goram@fernuni-hagen.de

Dirk Veiel
FernUniversität in Hagen
58084 Hagen, Germany
dirk.veiel@fernuni-hagen.de

Abstract

Intelligent applications and Artificial Intelligence are based on complex procedures and calculations in order to make decisions in challenging situations. In order to explain decision making to users, new approaches are required. Context-aware systems can be used to support personalization with regard to the current situation of related users. The challenge of context-based systems is to support the many different situations with personalized explanations. In this paper, we address this challenge. We use three different collaboration situations to illustrate our approach. Therefore we describe specific collaboration situations motivated by ethical or legal aspects in related adaptation policies. To provide personalized explanations in these situations, we use the CONTact platform and extend the related adaptation runtime environment with an explanation building component. This component uses the context, i.e. the current collaboration situation, to create dedicated personalized explanations.

1 Introduction

Intelligent applications and Artificial Intelligence are based on complex procedures and calculations in order to make decisions in challenging situations and to be able to act adequately [4]. Thereby transparency and traceability for decision making gets lost. In fact, these aspects often take second place to the performance of the systems [4]. From the user's point of view, it is important to understand why and how something is based in order to build trust in an intelligent application. In addition to the legal aspects, such as data protection, ethical and moral aspects such as the equal treatment and fairness of a system are also important [4]. It is important to support the users in situations where a system acts autonomously with explanations. To support users in certain situations the system must be aware of the user's situation and the related socio-technical environment, i.e. the context. According to Dey [2], context includes all information that can be used to describe a situation and contains information about the user, the technical and physical environment, as well as space and time. We extend his definition and add the legal constraints that exist in the specific situation. Due to that, the situation must include a proper evaluation of the legal facts that are needed to explain and support in the situation at hand. Explanations are important for intelligent personalized systems to support user acceptance and user trust.

In: Rudolph S., Marreiros G. (eds.): Proceedings of STAIRS, Santiago de Compostela, Spain, august 2020, published at http://ceur-ws.org

According to [3], the privacy control is strongly related to intelligible explanations. Therefore, it is necessary to explain system processes and data usage, to help users to understand the current situation [1].

Context-aware systems are used to support personalization with regard to the current situation of related users. These kinds of systems can be used to provide personalized explanations too. But this is no easy task, because different situations require different explanations which should be made available to the users in the specific situation. This makes it comprehensible for the individual user to understand what has happened in the application and why certain actions are not carried out or lead to an unexpected result, e.g., a common workspace is not created if there are aversions between participants, or an agreement to the data processing is missing that declares an explanation with agreement function. Other situations where the individual characteristics and abilities of the users must be taken into account are may be differences in language style and level of difficulty (e.g., simple language). Also the usage of jargon in notes and help texts for the respective target group is important as well as the provision of data usage reports in accordance with § 15 of the General Data Protection Regulation (GDPR). The challenge of context-based systems is to support the many different situations with personalized explanations. In this paper, we address this challenge. For that, we explain our context-based adaptive system approach and our work in a nutshell. Based on this, we introduce sample scenarios in section 3 to illustrate our approach on how to handle different situations with suitable explanations. In section 4 we describe our explanation building components and process. After the presentation of related work in section 5, we discuss our approach (cf. section 6) and finish with conclusions and future work in section 7.

2 Background

We develop an extendable context-based adaptive system environment (eCBASE) as a basis system for the development and integration of domain-specific context-based adaptive applications. eCBASE is designed to support users of the platform and domain-specific applications in a situation-specific way. In addition to the support for the use of the application or the processing of artifacts, the environment should also highlight the legal regulations that apply when using the application and explain the consequences for its use.

2.1 Context-based Adaptive Approach

We define a context-based adaptive approach as the provision of an environment that is able to perceive a situation, recognize the need for action and make an adequate adaptation of the system. This should support users in specific situations. In doing so, we address the individual needs of the users through personalized adaptations. In a group or collaboration situation, the system has to find a consent that takes into account the situation and circumstances of the individual group members.

Our context-based adaptive system is based on a formal model that we model using an ontology. We developed a core model with all necessary concepts and relationships which represents objects and functionalities within every supported application domain. This model will be extended for different scenarios and application domains at so-called extension points. It provides the basis to represent the context at runtime. For context modelling we use the OWL 2 Web Ontology Language and the generic four-layer framework for modelling context in a collaboration environment and the related collaboration domain model presented in [8].

2.2 Extendable context-based adaptive system environment

eCBASE is a common platform of functions and technologies for providing context-based adaptive and personalized software applications that can be made available for different domains. In the so-called core system, we model domain-independent and domain-specific objects and relationships as well as fundamental legal aspects, which must be considered in all future domain-specific applications. The design of the legal requirements is passed on to the responsible legal professional of the operating organization or the provider of a domain-specific application. eCBASE provides the necessary formal concepts and basic functions for this which are represented in a legal domain model (cf. [5]).

The application domain is the real-world environment and provides the setting in which a software application is used, e.g., online learning platforms (cf. [6]), community applications (cf. [5]), software development tools. Each of these domains (e.g., education, healthcare, software development) has special characteristics that do not exist in other areas. However, they also share a common basis which is necessary for IT-based support, e.g., recording, processing and changing data of objects or persons. Due to that, we call any application domain environment or system and the related requirements as domain-specific application or requirement.

3 Explanations in eCBASE

We explain our approach to generating personalized explanations in eCBASE in the following sections. We start with the introduction of a collaboration situation in which users can be supported by personalized explanations.

3.1 Why do we need personalized explanations?

A domain-specific context-based adaptive application senses the current situation and decides on the basis of the current available and related instances of the context which adaptation rule should be executed. Due to the adaptations in the applications by the adaptation rules, situations can occur in which the expectations of the users are not fulfilled (e.g., known action-response patterns do not occur, changes cannot be traced or reasons of the behavior are unknown). Therefore, it is necessary to explain system processes and data usage, to help users understand the current situation [1]. "The dynamic aspect of context implies that it is not possible to plan in advance the whole explanatory dialogue" [1, p. 123]. Therefore, the explanations must be generated in the specific situation. With our context-based approach it is possible to personalize the explanations, because we can gather all information of the users about the current situation through the context model. The personalized explanations shall "serve to clarify and make something understandable" [7, p. 498] to the user in a specific situation.

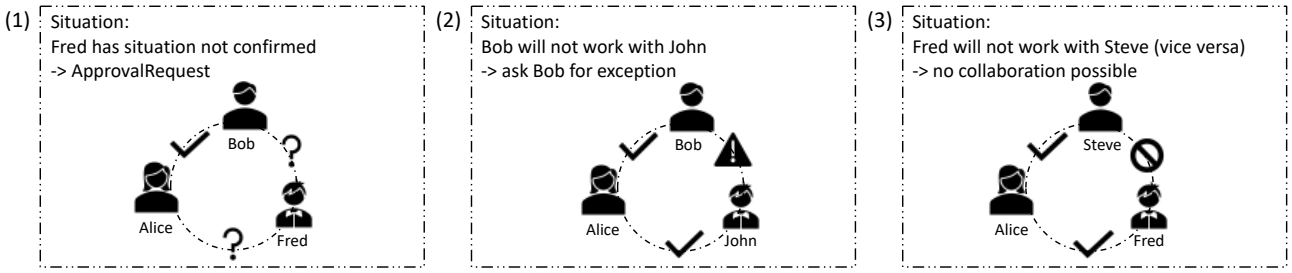

Figure 1: Scenarios

Our scenario takes place in an adaptive personalized learning environment (APLE) in which the instructor has specified that groups of three students each have to solve certain course tasks. In our scenario, we assume that a groupspace gets created as soon as three students access the same artifact (i.e. Task_A). Figure 1 illustrates three situations in which three users each access the same artifact, what could potentially lead to the creation of a groupspace. Each of these users has personal preferences as shown in Figure 2, which can concern, e.g.,, the type of collaboration (i.e. a shared groupspace; cf. Figure 2, *dm:ApplicationFunctionality*) and the collaboration partners (cf. Figure 2, *dm:GreyList*, *dm:BlackList*). These preferences should be taken into account for ethical and legal reasons when creating the groupspace.

Figure 2: Preferences

Before the groupspace gets created, the sensing engine gathers information about the current collaboration situation. The adaptation engine uses this information to create the context. It checks whether an adaption is possible or not. Therefore, it evaluates all realted adaptation policies (cf. Figure 3). An adaptation policy consists of a condition (*WHEN* to *DO*) and an action block (*DO* to *END*). As soon as the condition block evaluates to non null results, the action block gets executed. In [6] we have described how the use of the function for recommending suitable group members is determined by the data processing preferences of the users.

As shown in Figure 3 (1), we retrieve all users within the current collaboration situation, the context. We iterate over all users to apply the rule "Compliance by Design", i.e. we retrieve the application (*getApplicationInContext*) and the related requirement from the users context (*getRequirementInContext*) and request an approval for this specific setting (*requestApproval*). As soon as the user approves the request, we call *createOrUpdateAcceptedApproval* to store the decision in the context and notify the user about this operation.

The adaption policy (2) in Figure 3 takes the second scenario (cf. Figure 1 (2)) into account. Therefore we use the context, i.e. the current collaboration situation, to get relevant users (*getUsersInContext*), retrieve the related task they are working on (*getTaskInContext*) and retrieve the approvals and preferences. The function *getPreferences* enables us to retrieve specific context concepts from a users profile, in our sample *dm:GreyList*. The function *getTeam* uses the above context concepts and relations to check whether a team can be build or not. In the second scenario we are able to build a team. This leads to the execution of the action block, where related explanations will be created, before we check whether we need an additional confirmation or not (because of the grey list). When no confirmation is required we can open the shared groupspace for this team.

To support the third scenario (cf. Figure 1 (3)), we use the adaptation policy (3) in Figure 3. The condition block is similar to adaptation policy (2), despite the retrieval of the concepts *dm:BlackList* from users preferences. Instances of these concepts indicate that users do not want to collaborate with specific persons. Therefore we have to take care of it. The function *getTeam* checks whether there are users present in related black lists of other users that should form this team (cf. third scenario). If this is the case, the action block gets triggered. We do not create a team, but we explain the current collaboration situation and create related personalized explanations so that users get aware of it.

Scenario 1: The first scenario (cf. Figure 1 (1)) illustrates the situation when Alice and Bob already approved the usage of the *SharedGroupspace* (an application functionality) for *Task_A*. The third possible group member Fred has no approval for the usage of a *SharedGroupspace*, what leads to a situation that the adaptation rule "Compliance by Design" (cf. [5]) creates an approval request (cf. Figure 1, (1)) to notify Fred and to ask him for permission. If Fred allows APLE to use his data for a shared groupspace in this situation, a shared groupspace will be created, otherwise this group cannot collaborate together. The situation can be explained as illustrated in Figure 4, the requested approval and its explanation is on the left of Figure 4.

(1)
```
users = getUsersInContext("dm:User")
FOREACH user IN users DO
  # rule "Compliance by Design"
  WHEN
    app = getApplicationInContext(user,
            "dm:Application")
    req = getRequirementInContext(app,
            "dm:Requirement")
    appr = requestApproval(user, app, req)
  DO
    createOrUpdateAcceptedApproval(appr)
    notify(user, appr)
  END
END
```

(2)
```
WHEN
  users = getUsersInContext("dm:User")
  task = getTaskInContext(users)
  apprs = getApprovalsInContext(users,
            "dm:Approval")
  prefs = getPreferences(users, "dm:GreyList")
  team = getTeam(users, apprs, prefs, task)
DO
  expls = createExplanations(team)
  IF requiresConfirmation(team) THEN
    requestConfirmation(team, expls)
  ELSE
    openSharedGroupspace(team, expls)
  END
END
```

(3)
```
WHEN
  users = getUsersInContext("dm:User")
  task = getTaskInContext(users)
  apprs = getApprovalsInContext(users,
            "dm:Approval")
  prefs = getPreferences(users, "dm:BlackList")
  team = getTeam(users, apprs, prefs, task)
DO
  expls = createExplanations(team)
  notify(team, expls)
END
```

Figure 3: Adaptation Policies

Scenario 2: The second scenario illustrate the case, that the user preference of Bob prevents the collaboration between Alice, Bob and John. Bob had a bad experience with John and prefers to not work together with him again (cf. Figure 1, (2) exclamation mark). For that Bob has put John on his grey list (cf. Figure 2, *dm:GreyList*), which is used to indicate reservations about other users. But the grey list does not express a strict aversion. Therefore, Bob is told that a group could be identified for the group task, but John would be involved. Bob is asked if he would like to work with John to complete the group task this time (cf. Figure 4, center).

Scenario 3: This time the users Alice, Steve and Fred are possible group members, but Steve and Fred listed each other on their blacklist (cf. Figure 2, *dm:BlackList*). This expresses a strict aversion what makes it impossible to create shared groupspace for them (cf. Figure 1, (3)). Alice, Steve and Fred get a notification that currently no collaboration is possible (cf. Figure 4, right).

4 Explanation Builder Components

With the generic process presented in [5], we get the relevant information to create the explanations from the context.

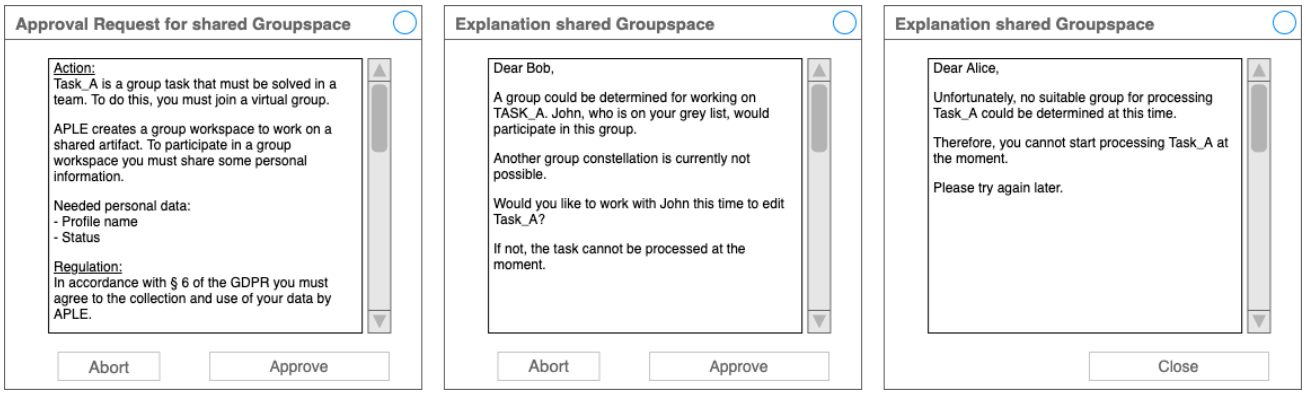

Figure 4: Samples of personalized explanations

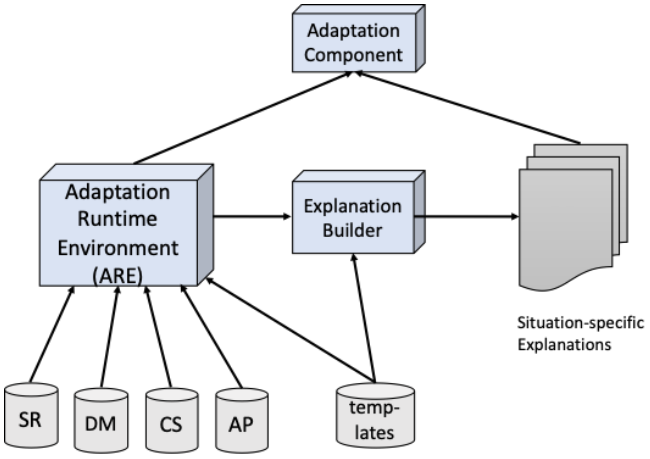

Figure 5: Explanation Builder

In Figure 5 we show the required components to generate situation-specific personalized explanations for the users. The *Adaptation Runtime Environment (ARE)* contains the sensing and adaptation engine to apply suitable adaptation policies (AP). Based on the domain model (DM), the sensing rules (SR) are used to generate the state. Applying the contextualization strategies (CS) to the state creates the contextualized state, i.e. the context of the current situation. The contextualized state is used to evaluation the condition blocks of adaptation policies. As soon as an action block can be executed, this is done through the Adaptation Component. To generate personalized explanations, the contextualized state and suitable predefined templates are given to the Explanation Builder. It generates situation-specific explanations which are transferred to the Adaptation Component to create and display a user interface artifact.

In Figure 6 we present a rough domain model which contains concepts and relationships to create a contextualized state according to the above introduced scenario. The prefix *dm* is used for common core concepts in eCBASE. For concepts and relationships of the learning management domain, we use the prefix *lms*. Does the concepts belong to legal regulations domain we use *lr*.

The important core concepts *dm:Requirement*, *dm:Condition* and *dm:Declaration* and their dependencies are used to support user control and intelligible explanations. Requirements (*dm:Requirement*) are conditions for applications and define what an application (*dm:Application*) or application functionality (*dm:ApplicationFunctionality*) must check and take into account during processing. The requirements are not stored as a fixed set of rules. Requirements can be related to technical conditions (*dm:Technical*), content definition (*dm:Content*) and/or legal regulations (*dm:Legal*).

Conditions (*dm:Condition*) determine what an application (*dm:Application*) or an application functionality (*dm:ApplicationFunctionality*) has to consider at runtime and how it should deal with certain situations. The

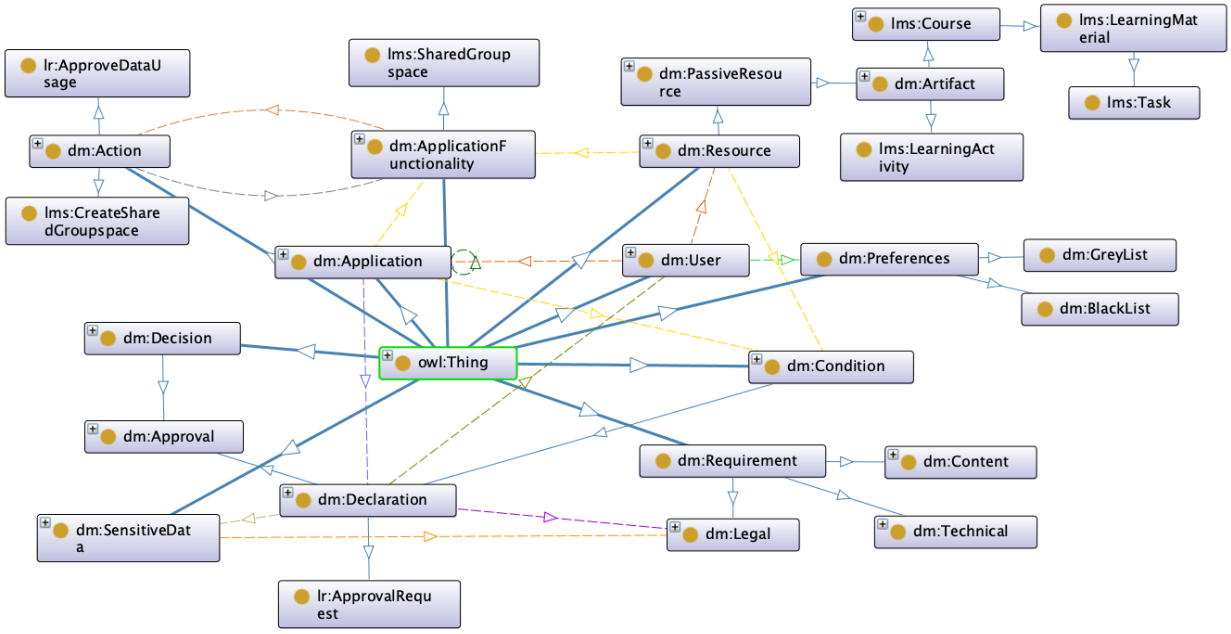

Figure 6: Domain Model

set of rules for the conditions is derived from the requirements, who are defined by the experts. Declarations (*dm:Declaration*) are the interfaces to users which can support comprehensibility and user control. In [6] we illustrated how legal professionals and domain experts can provide explanations for the declarations. Instances of the concept *dm:Declaration* will be created by the *Explanation Builder* which conducts the structure of the explanatory dialog and the related information of the specific situation. The purpose of these concepts is to explain *What happened?* (*dm:Requirement*), *Why did it happened?* (*dm:Condition*) and *What kind of explanation should be provided?* (*dm:Declaration*). During application runtime, all of the aforementioned concepts are instantiated for a specific situation, i.e. the context. Using the instances from the current context makes it possible to explain the situations to the users as illustrated in our sample scenario.

The extensions in Figure 6 belongs to our scenarios we described above. The students (instances of *dm:User*) are present in an online course (*lms:Course*) with learning material (*lms:LearningMaterial*). They do learning activities (*lms:LearningActivity*) and e.g., work on Task_A (instance of *lms:Taks*), which are artifacts (*dm:Artifact*) and belong to the passive resources (*dm:PassiveResource* subclass of *dm:Resource*) within the application APLE. For that, they use a shared groupspace (*lms:SharedGroupspace*) that must be created (*lms:CreateSharedGroupspace*) through an action (*dm:Action*). The shared groupspace uses personal data (*dm:Sensitive*) of the students (instances of *dm:User*). Due to that, they have to decide (*dm:Decision*) and approve (*dm:Approval*) the data usage before accessing the collaboration situation. If a situation requires an approval, as illustrated with Fred, the *lr:ApproveDataUsage* action is triggered and provides an explanation as *lr:ApprovalRequest*. The preferences (*dm:Preferences*) of the students for the assembling of workgroups are represented as black list (*dm:BlackList*) and grey list (*dm:GreyList*).

5 Related Work

Supporting intelligibility of complex context-aware systems is the approach of [10]. They point out that intelligibility must be accompanied by a control function for the user. Their work focus on an extension of the Context Toolkit. "The Context Toolkit aims at facilitating the development and deployment of context-aware applications."[1] The extension supports developers and designers who use the Context Toolkit to integrate intelligible explanations and user control while building a context-aware application. For that, they integrate meaningful explanations in the application "Situation" by exposing the internal processing of context-aware applications.

[1] http://contexttoolkit.sourceforge.net

Enhancements to the explanation component in the Context Toolkit are presented in [11]. They generate explanations of the behavior of more popular machine learning techniques and enriched explanations for user control [10, 11]. It is not known to us that the Context Toolkit supports context-based collaborative environments as well as legal regulations.

[8] present a generic four-layer framework for modelling context in a collaboration environment, a generic adaptation process, and a collaboration domain model for describing collaboration environments and collaboration situations. [12] implements the framework, using an extended domain model and the related adaptation process. The resulting CONTact platform is able to sense and formalize users' interaction with the system at runtime, and to adapt according to the user's current collaboration situation. The adaptation process may confuse users. Therefore, [9] enhanced the platform with context enriched explanations to help them understand the adaptation behavior. The explanations are applied on executed adaptation rules to support the user's understanding on the intention, terminology and the consequences of the related adaptation rule. For that, [9] use bound variables which are integrated in static explanation blocks of the adaptation rule to add situation-specific information (e.g., time and user) from the context to the explanations. The explanations are available on demand by pushing a button. For further information about the situation possible communication partners are identified and recommended in a buddy list.

The provided explanations are tailored to a situation when the adaptation rule gets executed and explain more general aspects of the situation. The explanations are neither personalized to the users circumstances nor support different kinds of explanations. Additionally, the platform does neither support legal regulation nor its integration or explanations about legal conditions. So far, there are no known context-based collaborative systems that support intelligibility for users and legal compliance as our presented approach.

6 Discussion

We presented an approach on how to support users with personalized explanations within a context-based adaptive system environment. Our sample scenario used a context-based adaptive learning environment APLE to illustrate how we deal with different collaboratin situations. These situations demand adaptation. We showed related adaptation policies that are capable of recognizing specific collaboration situations motivated by ethical or legal aspects. In our sample scenario, we used different preferences of the involved users and the legal requirements to illustrate it. These constraints can delay or prevent collaboration. Depending on the context, an appropriate strategy for handling the situation through the context-based application must be applied. In the presented approach, we address this challenge with generic adaptation policies and the Explanation Builder. Based on CONTact, our starting platform, eCBASE allows us to handle different collaboration situations, which has already been demonstrated by [12]. The Explanation Builder will use context information to personalize explanations. By integrating legal, content-related and technical requirements into our domain model, we can respond to these requirements (e.g., § 15 GDPR). Descriptions contained in the requirements can be used to create personalized explanations (cf. [5]).

7 Conclusion and Future Work

In this paper we presented an approach for collaboration support and personalized explanations in context-based adaptive systems based on the CONTact platform (cf. [12]). For this purpose, we extended the existing domain model of the CONTact platform with concepts and relationships which are needed to support a collaboration situation in APLE. Additionally, we briefly described an explanation building process and the related components to the Explanation Builder. Thereby we haven taken into account the legal requirements and integrated them into the adaptation rules and explanations of eCBASE.

To address the introduced challenge about the support of many different situations with personalized explanations, we combine the adaptation runtime environment of CONTact with the Explanation Builder to support personalization with regard to the current situation of related users. We illustrated the usage and creation of personalized explanations in our sample scenarios and the related adaptation rules.

We described how to create personalized explanations, but we also need to provide intelligible explanations what is a challenging task. To address the intelligibility of explanations a series of user studies needs to be done for each application domain which should be supported by eCBASE. Additionally, we have to find out, when and what is the right situation to provide explanations. Due to that, future work will focus on user interaction design and the way in which texts are formulated. Both must be integrated in eCBASE.

Acknowledgements

This work was supported by the Research Cluster "Digitalization, Diversity and Lifelong Learning. Consequences for Higher Education" (D^2L^2) of the FernUniversität in Hagen funded by the Ministry of Culture and Science of the German State of North Rhine-Westphalia.

References

[1] Patrick Brezillon, 'Contextualized explanations', in *Proceedings of International Conference on Expert Systems for Development*, pp. 119–124. IEEE, (1994).

[2] Anind K Dey, 'Understanding and using context', *Personal and ubiquitous computing*, **5**(1), 4–7, (2001).

[3] Anind K Dey and Alan Newberger, 'Support for context-aware intelligibility and control', in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 859–868, (2009).

[4] Virginia Dignum, Matteo Baldoni, Cristina Baroglio, Maurizio Caon, Raja Chatila, Louise Dennis, Gonzalo Génova, Galit Haim, Malte S Kließ, Maite Lopez-Sanchez, et al., 'Ethics by design: necessity or curse?', in *Proceedings of the 2018 AAAI/ACM Conference on AI, Ethics, and Society*, pp. 60–66, (2018).

[5] Mandy Goram and Dirk Veiel, 'Supporting privacy control and personalized data usage explanations in a context-based adaptive collaboration environment', in *11th International and Interdisciplinary Conference, CONTEXT 2019, Trento, Italy, November 20–22, 2019, Proceedings*, eds., Gábor Bella and Paolo Bouquet, volume 11939. Springer International Publishing, (2019).

[6] Mandy Goram and Dirk Veiel, 'Linking legal and domain-specific requirements in a context-based adaptive personalized learning environment', *Procedia Computer Science*, **170**, 995 – 1002, (2020). The 11th International Conference on Ambient Systems, Networks and Technologies (ANT) / The 3rd International Conference on Emerging Data and Industry 4.0 (EDI40) / Affiliated Workshops.

[7] Shirley Gregor and Izak Benbasat, 'Explanations from intelligent systems: Theoretical foundations and implications for practice', *MIS quarterly*, 497–530, (1999).

[8] Joerg M. Haake, Tim Hussein, Björn Joop, Stephan Lukosch, Dirk Veiel, and Jürgen Ziegler, 'Modeling and exploiting context for adaptive collaboration', *International Journal of Cooperative Information Systems (IJCIS)*, **19**(1-2), 71 – 120, (2010).

[9] Syed Sajid Hussain, Dirk Veiel, Joerg M. Haake, and Stephan Lukosch, 'Facilitating understanding team-based adaptation policies', in *The 6th International Conference on Collaborative Computing: Networking, Applications and Worksharing, CollaborateCom 2010, Chicago, IL, USA, 9-12 October 2010*, pp. 1–8. IEEE, (2010).

[10] Brian Y Lim and Anind K Dey, 'Toolkit to support intelligibility in context-aware applications', in *Proceedings of the 12th ACM international conference on Ubiquitous computing*, pp. 13–22, (2010).

[11] Brian Y Lim and Anind K Dey, 'Design of an intelligible mobile context-aware application', in *Proceedings of the 13th international conference on human computer interaction with mobile devices and services*, pp. 157–166, (2011).

[12] Dirk Veiel, Joerg M. Haake, Stephan Lukosch, and Gwendolyn Kolfschoten, 'On the acceptance of automatic facilitation in a context-adaptive group support system', in *46th Hawaii International Conference on System Sciences (HICSS)*, pp. 509–518. IEEE Computer Society, (1 2013).