End-User Controlled
Group Formation and Access Rights Management
in a Shared Workspace System
End-User Controlled Group Formation and Access Rights Management in a Shared Workspace System

Joerg M. Haake, Anja Haake, Till Schümmer, Mohamed Bourimi, Britta Landgraf
FernUniversität in Hagen
Informatikzentrum, Universitätsstrasse 1, 58084 Hagen, Germany
Telephone +49 2331 987 327
joerg.haake@fernuni-hagen.de

ABSTRACT
Group formation and access rights management become crucial issues when shared workspaces are used to support flexible, emerging group work. End-Users should be able to form groups and adapt access rights for changing groups and workspaces. Current shared workspace systems do not support this sufficiently. Our approach combines a room metaphor-based shared workspace with the key-metaphor for facilitating both, end-user controlled flexible group formation and access rights management. An evaluation of this approach during four month of use has indicated that end-users can form groups and manage the access rights of their shared spaces.

Categories and Subject Descriptors
H.4.1 [Office Automation]: Groupware
K.3.1 [Computer Uses in Education]: Collaborative Learning

General Terms
Design.

Keywords
Groupware, group formation, access rights management, access control, shared workspaces, CSCL, CURE.

1. INTRODUCTION
Many forms of collaboration, such as peer-learning groups or innovation teams in virtual organizations, require dynamic group formation. In such teams, e.g., availability of group members or the needed expertise may change. Thus, team membership will change over time. We argue that in these situations only the users themselves can define and adapt their groups properly. Thus, end-user controlled group formation becomes crucial.

Emerging forms of group formation pose another problem to a shared workspace system that provides each group with a dedicated shared space: the problem of adapting access permissions to shared spaces with changing membership. Already in 1991, Ellis, Gibbs and Rein [5] argued: “Since access information changes frequently, there must be lightweight access control mechanisms that allow end-users to easily specify changes”. We are convinced that only the users of a shared space themselves can properly define the access permissions. Given a large number of groups in such a system, no central administrator may serve all change requests in due time. Rather, the users themselves must understand and control the access permissions.

In summary, we found that end-users in order to conduct cooperative work in a shared workspace system have to be able to (1) create groups dynamically without prior planning of a system administrator, (2) employ different forms of group formation, and (3) control access rights to their workspaces. Current approaches do not sufficiently support these requirements from an end-users perspective (cf. section 2).

To facilitate end-user controlled flexible group formation and end-user controlled access rights management, our approach combines a room metaphor-based shared workspace system with the key-metaphor. We implemented this approach in CURE, a web-based shared workspace environment. While previous papers on CURE [7][8] reported solely on end-user tailoring of room structure and content for collaborative learning, this paper presents for the first time CURE’s functionality for group formation and access control.

Section 2 reviews related work on group formation and access rights management. Section 3 presents our approach. Section 4 reports our experiences. Section 5 concludes the paper discussing the applicability of our approach to other application domains and raises questions for future work.

2. STATE OF THE ART
2.1 Group Formation
Group formation is the process needed to identify group members for a collaboration episode. Some systems support group formation only. Examples include agent-based expert finders or peer-learner finders [19][24]. These systems store profile data on its users and enable users to find partners matching certain constraints on their profiles. The set of matching users may be used to compose a group, e.g. of peer learners working on a joint exercise. E.g., Wessner and Pfirst [19] organize cooperative learning as activities modeled as an IPoCs (Intentional Point of
Cooperation). Based on this specification, the system can compute groups of learners fulfilling the IPoC characteristics. Learners then receive invitations and have to agree to participate. Alternatively, tutors can manually assign learners to IPoCs.

Some place-based systems offer group formation. VITAL [12] offers rooms for learning activities. Users working synchronously in the same room are considered a group. Thus, groups can only be formed informally as a result of entering the same room.

BSCW [1] provides public shared spaces and shared workspaces to invited users. BSCW supports group formation via invitation: the workspace owner can send invitations by e-mail to users, which may complete their registration and become workspace members. In BSCW, users have their own workspaces. They can create new sub workspaces and may decide to share them with other users they invite. A workspace becomes a shared workspace either by inviting other users into it, or by declaring this workspace a sub workspace of another shared workspace. In the latter case, the users of the original shared workspace become also users of the new contained sub workspace. BSCW’s access permissions are discussed in the next section.

### 2.2 Access Rights Management

Access rights management is an important issue in operating systems and data management systems. Access control lists and role-based access control [14][15] are today’s standard means of managing access permissions in these systems. Most of this work is based on the classic matrix model from Lampson [10] proposed for protection in operating systems. Rows of the matrix list subjects (called domains in [10]), columns objects, and each matrix cell entry defines the access rights the respective subject has over the respective object. While the matrix defines the basic access rights, i.e. the rights of the subjects on the objects, the model also provides access right administration or meta-access control, i.e. it comprises also a set of commands and rules that specify how the matrix can be altered.

Several implementation options have been explored for the matrix model resulting in capability-lists (access rights stored with the accessing subject), access-lists (access rights stored with the object to be accessed) or the single-key-lock mechanism [20]. The latter equips every subject with a single key and every object with a single lock in such a way that the subjects’ keys and objects’ locks match according to the specified access rights in the matrix.

The HYDRA [23] operating system (kernel) provides a protection mechanism for the application of operations (procedures/subjects) to instances of resources (objects) called capabilities (not to be confused with capability-lists, cf. above). A capability consists of a reference to an object together with a collection of access rights to that object. Objects may only be referenced by capabilities, also when building up complex object structures.

The classic matrix model has been extended for groupware by Shen and Dewan [17]. They propose an extended access matrix with more than 50 basic access rights to cope with the large variety of commands supported by the Suite multi-user framework in order to describe the potential dimensions of groupware such as session management or coupling. To ease the specification of these basic rights, the authors introduced concepts such as right grouping, inheritance, negative rights and right implication. The actual determination of the access rights of a given subject on a given object based on all these concepts may result in conflicting specifications for which resolution rules are specified [17][3]. The meta-access control for the model described in [4] includes concepts such as revocation, i.e. removing previously granted rights, and extends the notion of object ownership by providing the possibility to add roles to an object’s owner list.

Stevens and Wulf [16] add another dimension to access control in groupware by allowing not only pre-planned access right specification (ex-ante control), but by also supporting access right specification at the moment of the access (uno-tempore control) or access checking after the access took place (ex-post control). By taking into account the roles of trusted third persons, awareness, and negotiation in real world situations of unexpected access demands [18] the authors develop a model that provides six different mechanisms to support users in regulating access conflicts [21]. The model is based on the combination of three design options. Combining the design options (1) having the right to intervene against the activation of a function and (2) availability of awareness of activation of a function results in four different situations for access conflict resolution. If awareness is available, two options for conflict resolution can be further refined depending on the availability of a communication channel.

While all previously discussed models focus on accessing individual objects or documents, Bullock and Benford [2] proposed the SPACE access control model for controlling the access of groups to cooperation spaces. SPACE uses the concept of boundaries to group objects and control access to these objects. While the concept of boundaries is very intuitive (and as [2] argues can be found in many approaches), we see a problem for end-users in understanding the distribution of permissions: access control is managed by access graphs that define numerically, whether a user can move from one boundary to another boundary. For us, such a technical description seems to be difficult to comprehend and define for end-users because they have to abstract from the virtual environments (constructed by boundaries) and move to a numerical model that only considers the environment’s structure.

TeamRooms [13] provides persistent places. Access to places can be constrained by access rights defined by an administrator. These access rights apply to a complete server or to a list of workspaces. Thus TeamRooms does not support end-users in managing access permissions on individual places.

A metaphor of locks has been used in MOOs, where interaction is organized in places (such as rooms)[9]. By default, these places are public. The owner of a place can make it private by adding a lock to the place. The lock can be opened for specific users by the place owner using ‘unlock’ commands. Thus, groups may be formed only by the owner by assigning users to the place.

In BSCW [1], a complex role-based access control method is used to define the roles of users. A role has a name and defines what a user having this role can do with the object/workspace. BSCW distinguishes three types of roles: normal, inheritable, and fixed roles. Roles may be redefined in workspaces. While this is a powerful and flexible concept, users must also deal with roles that may have similar names but mean different permissions in different workspaces.
2.3 Conclusion

All above approaches support only limited modes of group formation. Next to free access, most models support invitation. Only one [19] supports assignment, too.

All approaches discussed above do not support simple end-user controlled definition of access permissions. An exception is the approach described in [16], [18], and [21] extending access rights specification by enabling negotiation of access rights between requester and grantee at the moment of the access. The broader theoretic work on access models and cooperation environments for building cooperative system put forth rather complex access control models, making use of concepts like negative right, inheritance, and group-hierarchies (cf. [5] also for an anticipation of this situation). Access rights management by end-users is insufficiently supported, either due to too complex role-models, access control parameters and user interfaces that end-users cannot easily understand, or due to insufficient functionality.

In summary, current approaches do not sufficiently support dynamic group formation and appropriate adaptation of access permissions for shared workspaces by end-users.

3. APPROACH

Our approach was developed in a project aiming at a tailorable groupware platform for CSDL [7][8]. During early design and development it became clear, that in order to empower end-users we need to “smoothly mesh the access model with the user’s conceptual model of the system” at the user interface [5].

The resulting model is a combination of two metaphors and a corresponding user interface that support end-users in forming groups and managing access permissions for their shared workspaces. We begin with the room concept as a metaphor for places for collaborating groups and then extend it with the concept of virtual keys as a metaphor for both, access permissions for places and membership in the group using the place. We illustrate the concepts and the key design principles of the user interface with examples of our CURE shared workspace environment.

3.1 Rooms as Metaphoric Places for Interaction

Room metaphors have been widely used to structure collaboration [6][12][13]. Though the models vary slightly, they have a lot of commonalities. For our model of group formation and access rights management, we assume that a room has the following characteristics: A room represents a virtual place for collaboration. It contains documents as artifacts for collaboration and keeps those artifacts persistent, and it may have adjacent rooms to further structure the virtual space for collaboration. The room provides group-awareness by reporting to each user which other users are currently in the room. The room provides a synchronous communication channel, e.g. a chat in CURE, and a room-specific bulletin board as asynchronous communication channel, e.g. a room-specific persistent mailbox in CURE.

We assume that users present in a shared workspace system are always located in a room. Users enter the system via a public entry room. Starting from that room, users can either navigate to adjacent rooms or create new adjacent rooms. Within every room they may access the rooms’ documents, add or change documents, or use the communication channels.

A system based on this model has no explicit support for group formation or access rights. Users enter the system, move around, and interact as they like. Groups form only informally as a result of users entering the same room at the same time. The remainder of the paper introduces our model of adding the metaphor of virtual keys to the metaphor of virtual rooms to cope with both, group formation and access rights management.

3.2 The Metaphor of Virtual Keys

3.2.1 Our model

Each room is equipped with a lock, and users need a virtual key matching this lock in order to get access to this room. To support public areas which can be entered by all users, so-called public rooms provide a default key that can be used by every user.

The key represents not only the permission to enter a room but also specifies what the key holder can do in the room. We identified three different classes of rights: key-rights defining what the user can do with the key, room-rights defining whether or not a user can enter a room or change the room structure, and interaction-rights specifying what the user can do in the room.

Figure 1 shows which rights are currently supported by our model in the context of a key request in CURE. Users can select their desired rights for each class of rights. Rights are ordered from least (most restrictive) to right (less restrictive). A less restrictive right always includes all more restrictive rights. A key offering maximal rights in all three classes is called a master key for the respective room.

Key-rights specify what the key holder may do with his key. If the key holder does not need his key for the room anymore, he may return the key to the room the key belongs to, delete his key, or pass it to another person. If the user wants to keep her key but wants to propagate her key to other users, the key needs to have the key-right “copy key”. Having the “copy key”-right, the user can create copies of this key with potentially restricted rights and give this copy to other users. To grant access rights only for a certain time frame, keys may be equipped with temporal restrictions, e.g. being valid from a certain point in time until a certain point in time.
Looking at the classic access matrix of Lampson [10], protected objects are rooms, subsuming the protection of all documents contained in the room, and access requesting subjects are individual users. The matrix’ cells are filled up with keys, specifying both, basic and meta-access control a single user has on a certain room.

Virtual keys correspond to capabilities and key-rights to capability-rights in HYDRA [23]. HYDRA has no notion of object ownership to ground meta-access control. Instead, they allow capabilities for protected objects to be passed to others, by associating a capability not only with the rights to the referenced object but also with a special copy right to the capability itself. Similarly, keys may be passed to other subjects and are not only associated with the rights to the referenced object but also with a special copy right to the key itself (cf. figure 1). Actually, the key-rights associated with a key specify the meta-access control features the owner of the key owns. These meta-access control features are complemented by operations on keys to further distribute and revoke keys.

Whereas the key-rights are related to meta-access control, room-rights and interactions rights define basic access control. Room-rights and interaction-rights may be considered being two right groups as defined by [17]. But these are the only two groups of rights we provide and there is no inheritance of rights. This simplification aids end-users in understanding our model.

Similar to the work in [17], implication of rights is used to ease right specification. Compared to the fine-grained specification possible in the work of Dewan and Shen, the reading-right already implies other rights such as scrolling or the edit right implies insertion. Instead of more than 50 basic rights we just provide 8 basic rights which seem to be powerful enough for our application domain (cf. section 4). In particular, since each group of rights is ordered completely with respect to an include relationship (see below), the specification of basic rights requires two decisions only, namely to select the desired room- and interaction-right. This simplifies right specification drastically.

The include relationship chosen for interaction-rights and shown in figure 1 is comparable to the imply relationship used for interaction-rights. Whereas the imply relationship [17] demands that right implication respects operation consistency, i.e. an implied right R1 is a prerequisite to execute the operation associated with the right R2 implying right R1, inclusion is just a means for right specification. However, inclusion respects implication in the sense that for a right R1 including a right R2, R2 may not imply R1. So, our room-right inclusion scheme respects the imply relationship reflecting the semantics of our room model: In our model, users are always in a room. The prerequisite for adding an adjacent room therefore is to be in the respective room. Nevertheless, a user does not have to be in a room (“enter room”-right) to delete this room (“delete room”-right). However, since deleting a room is such a drastic operation we decided that this right includes all other rights on a room.

To keep the model simple and understandable for non-expert computer users, we do not provide inheritance among right groups (there are just two groups) nor negative rights. We found conflict resolution being too difficult to understand for non-expert users. Simply, the set of rights a given user possesses on all the objects in a room is determined by the union of all keys the user is holding for that particular room.

Whereas other approaches express access to single or complex objects, we express access rights to a room and thus to all

3.2.2 Comparison to Related Work

Equipping an object with a lock and providing accessing subjects (i.e. users) with keys is close to the proposal of [20]. While [20] uses a single-key-lock approach for implementation, we use keys and locks as metaphoric complements of a room-based groupware system. Consequently, in our model a room has a single lock, but a user may possess more than one key for that lock.

Looking at the classic access matrix of Lampson [10], protected objects are rooms, subsuming the protection of all documents contained in the room, and access requesting subjects are
documents contained in the room at once. One may argue that such a specification is too coarse and end-users may tend to choose maximal access rights. However, we provide a versioning mechanism inside a room (cf. section 3.4), which facilitates negotiation of concurrent changes.

3.3 Group Formation using Virtual Keys

In the basic room-based collaboration environment introduced in section 3.1, a group is defined as the set of people using a certain room at a certain point in time. Introducing virtual keys to virtual rooms naturally extends the notion of a group to the set of those users holding a key for that room. Consequently, our model does not support groups as first class objects.

Consider the example of student working groups formed during a lab course in software development. Figure 3 shows one of these working groups consisting of nine people, which use the room “No Complaint”. Six users have full rights (master keys) whereas three other users (Bourimi, Lukosch and Schuemmer) are only allowed to delete or return their key, to enter the room, and to edit its content. They are not allowed to create more keys or to create adjacent rooms. The latter three users were actually tutors advising the six students. The three tutors were invited into the room by the six students managing this place.

In self organizing virtual teams, the group formation has to be administered by the potential group members themselves. In CURE, this is achieved by letting the users distribute keys for rooms among themselves. The possibilities for group formation are determined by key- and room-rights and the operations defined on keys and rooms.

In a shared workspace system using rooms and keys, users enter the system via a public entry room, e.g. called entry hall in CURE. Starting from the hall, users can either navigate to accessible rooms (i.e., other public rooms or rooms for which they hold a key) or, if they own the right to create adjacent rooms, create a new room. If a new room is created, the user automatically gets a master key for that room. This key permits every possible operation on the room.

Consider the example of student working groups formed during a lab course in software development. Figure 3 shows one of these working groups consisting of nine people, which use the room “No Complaint”. Six users have full rights (master keys) whereas three other users (Bourimi, Lukosch and Schuemmer) are only allowed to delete or return their key, to enter the room, and to edit its content. They are not allowed to create more keys or to create adjacent rooms. The latter three users were actually tutors advising the six students. The three tutors were invited into the room by the six students managing this place.

In self organizing virtual teams, the group formation has to be administered by the potential group members themselves. In CURE, this is achieved by letting the users distribute keys for rooms among themselves. The possibilities for group formation are determined by key- and room-rights and the operations defined on keys and rooms.

In a shared workspace system using rooms and keys, users enter the system via a public entry room, e.g. called entry hall in CURE. Starting from the hall, users can either navigate to accessible rooms (i.e., other public rooms or rooms for which they hold a key) or, if they own the right to create adjacent rooms, create a new room. If a new room is created, the user automatically gets a master key for that room. This key permits every possible operation on the room.

Properties of a room: No Complaint

<table>
<thead>
<tr>
<th>General Properties</th>
</tr>
</thead>
<tbody>
<tr>
<td>Created at: 2003-10-25 14:40:12.907</td>
</tr>
<tr>
<td>Description: Interaktives Spiel mit wundervollem Smalltalk</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Users with keys for this room</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bourimi, Mohamed</td>
</tr>
<tr>
<td>Laaks, Sven</td>
</tr>
<tr>
<td>Rücker, Kerstin</td>
</tr>
<tr>
<td>Schuemmer, Till</td>
</tr>
<tr>
<td>Vogl, Ingrid</td>
</tr>
<tr>
<td>Khouklaiev, Pavel</td>
</tr>
<tr>
<td>Lukosch, Stephan</td>
</tr>
<tr>
<td>Schobert, Wolfram</td>
</tr>
<tr>
<td>Taib, Mike</td>
</tr>
</tbody>
</table>

Figure 3: Properties of a room in CURE showing in particular the room users and their rights with respect to this room.

The creator of a room may distribute copies of the master keys for her room to other users. Those users are considered trusted third persons of the room creator since this master key empowers them to act on behalf of the room creator. All users holding a master key for a room form the management team for this room. Every member of the management team may distribute keys with the maximal key-right “copy key” but restricted room- or interaction-rights to other users. Those users are considered gate-keepers of that room with respect to their restricted room- and interactions rights. So, there is no strict notion of ownership in our model. Whereas [4] extends the notion of object ownership by providing the possibility to add roles to an object’s owner lists, we pursued the approach of trusted third persons [18] and gate-keepers [16].

All users of a room possessing a key with the “copy-key”-right may initiate group formation in the room by creating copies of the respective key with potentially restricted rights (cf. figure 1) and make them available to other users. Three distribution schemes for keys are supported:

- Firstly, the user may specify a default key every registered user gets assigned when entering this room.
- Secondly, the user may explicitly assign keys to registered users.
- Thirdly, the user may put keys on the door of the room.

In contrast to key distribution, users may also actively ask for a key for a room. This operation on rooms is available for every room in the system and may be executed by every user.

Different uses of replicable keys allow different modes of group formation: by assignment, by invitation, by free enrollment (placing the keys on the room’s door), and by confirmed enrollment (requesting a key from the room’s management team). In addition, public rooms may be entered by every user registered in the system. In the following, we discuss each mode of group formation in more detail.

To make a room public, any holder of the “copy-key”-right of the room can specify a default key that every registered user can use to enter this room. Since the set of rights a user possesses in a room is determined by the union of all keys the user is holding for that room, the default key defines the minimal access permissions of registered users in this public room.

For groups formed by assignment, any holder of the “copy-key”-right of the room may determine the group’s members. In this case, she assigns other users to the group by creating copies of her replicable key that, among other possible restrictions, at least have no key-rights linked to the key copy. One of those keys is then passed to each user, who should be in the group. Receivers of keys are notified by e-mail. Since in group formation by assignment the users receive keys with no key-rights, they cannot delete, return or pass on the key. This group formation scheme is, e.g., applicable to learning situations, where tutors have to assign students to groups working on collaborative exercises.

In order to form a group by invitation, any holder of the “copy-key”-right of the room may create and assign keys with the key-right to return the key. Thus, users receiving the key may access the room and belong to the group, or they may reject the invitation by returning the key. Students inviting peer learners into a room for exam preparation may use this group formation method. If the key distributed comprises the key-right to pass on the key, the invited person may delegate a deputy.

In case of a group with free enrollment, any holder of the “copy-key”-right of the room may create a set of copies of her key
In our model, we do not propagate group membership. In BSCW [1] if a workspace is included in a super workspace, the group members of the super workspace become automatically members of the new sub workspace. While this simplifies access right management in some cases, it may also cause problems if some sub groups should only have access to certain sub workspaces. This is, e.g., the case for virtual teams, which span company borders (e.g. sub contractors should only have access to some places in the project workspace hierarchy, see also [16] for a discussion of inter-organizational cooperation), or for learning groups that are dynamically divided into sub groups. Our model supports these situations by inviting the members of a room into other rooms explicitly.

Our model supports the spectrum of group formation ranging from (1) assignment to (2) invitation to (3) free enrollment to (4) confirmed enrollment. In particular, the ability to request a key for a room via the yellow door knocker is provided for every room. This is a solution for uno-tempore access control [16], i.e. specification of access rights at the moment of the access. In the CURE implementation of the model, the decision of a user to request a key for a certain room is supported by allowing every user to inspect the room properties of a room: Every listed room in the user-centered view shown in figure 4 is equipped with an information button labeled ‘i’ that shows the user the properties page of the room. Based on the information given, e.g. the description of the room or the users already working in this room, the user may decide to request a key for this room.

Using the yellow door knocker for a key request results in access control resolution via negotiation [18]: Initially a semi-structured message (i.e. structured specification of requested room rights and unstructured request justification) is sent to the management team, i.e. the addressee of the negotiation is the room’s management team. The members of the management team may discuss their decision or a single member may negotiate the requested rights with the requesting user using the communication channels provided by the room-based workspace system.

Users may also send spontaneous key requests to any other user of that room using the built in e-mail feature. In this way, requesting users may also ask gate-keepers for a key for that room.

Next to access control resolution via negotiation, our model also supports forms of access control resolution via discussability [18]. Group-awareness makes users of a room aware of each other. If a room user observes the execution of undesired operation by other room users, she may talk to them using the built-in communication channels. In the worst case, the management team may decide to exclude the suspected user by retracting his keys for that room, or the suspected person may decide to leave that group voluntarily and return or delete his keys.

Since keys comprise meta and basic access rights, negotiation and discussion of access rights addresses both, as in [22].
3.4 Access Rights Management using Virtual Keys

Defining access permission for a user on a room requires only 3 specifications from the user granting the right: Which key-right should be granted, and two decisions on basic access rights, i.e. the room- and interaction-right to be granted (cf. previous section 3.2). All other rights are defined by the inclusion relationship. This simplifies access right specification drastically.

Section 3.3 already discussed the administration of access rights by a management team of trusted persons or by gate keepers. The prerequisite for right administration is the “copy key”-right. For meta-access control, Dewan and Shen [4] introduce a *-right for every basic right, which allows the granting and revoking of that right. In our approach, possessing the “copy-key”-right for a key implies possessing the granting right for every basic right associated with that key. Whereas Dewan and Shen introduce shallow revocation of rights, i.e. removing the previously granted right, and deep revocation of rights, i.e. recursively revoking rights from those to whom the grantee granted the right, our approach offers shallow revocation of rights via key retraction. However, key retraction can only be performed by users possessing a master key for the room the key to be retracted belongs to. We do not support deep revocation since revoking a right does not undo the operations performed while the right was granted. This includes meta-access control, e.g. if a CEO fires a manager, he would not automatically fire team members hired by this manager. We argue right revocation has to be considered on an individual basis. Consequently, keys should be revoked individually. However, we allow a key holder to waive his rights by returning or deleting his key or passing it to another user. This also supports different ways of group formation (as shown in 3.3).

Initially, the access permissions for a room are defined by the room owner. But flexible emergent group work by definition constantly changes. Thus, access permissions must be adapted. We support two types of adaptations: spontaneous and pre-planned adaptations.

Firstly, room owners and all group members with sufficient rights may react to some new needs by changing the access permissions of the room’s users. They can simply retract some or all keys and assign new keys with changed access permissions to the users. Thus, ad hoc adaptation is possible.

Secondly, room owners may plan for certain phases of collaborative work. Consider the example of a working room of a sub group of a seminar. The students may all have the editing interaction right for the first eight weeks, which then, when the submission of the contribution is due, turns into the communication interaction right (to discuss the submitted papers). To implement such changing access permissions, the tutors of the seminar assign two keys to each student at the beginning of the seminar: one being valid from the starting date of the seminar lasting for eight weeks with the “edit”-right and another key being valid from eight weeks after the starting date of the seminar with the “communicate”-right (cf. figure 1). Thus, by distributing keys with limited validation period, different phases of collaboration can be supported.

Moreover, as discussed in the previous section, these mechanisms are complemented by the option to actively ask for keys.

Since group work is organized in rooms, access permissions are defined for rooms and their content as a whole. One may argue that this approach is too simplistic and may cause the unintentional destruction of others’ work. To avoid or at least reduce such risks, BSCW supports the concept of ownership of documents that influences and overlaps with the effect of having a certain access right to a space. Supposing that these interference may also contribute to some of the confusion of users using these systems, we chose the WIKI approach [11] to prevent the accidental destruction of other peoples work: Whereas BSCW allows its users to explicitly enable version control on selected individual documents, our implementation of the model in CURE provides a built-in, all-time-active version control mechanism on all documents. Editing a document simultaneously leads to parallel versions that can be merged later. Also [18] argue for storing versions of the same object if no consentaneous solution for a conflict can be found.

Moreover, in order to avoid unwanted versions, users may coordinate their behavior in different ways. If users are working synchronously, group awareness makes users aware of each other and they may coordinate behavior using the chat tool of the room. Asynchronous collaboration may be coordinated via soft locks by putting comments in a document. Since the first thing users do when beginning to work on a document is to read it, they will see the comment and may avoid conflicting access. However, individuals may still want to continue their work on the same document – and they can do this as their changes are automatically stored in another version of the document. Since old versions are accessible, a new version can be created that merges different changes.

If access conflicts become more frequent, the group should consider restructuring their workspaces. Maybe, a set of documents needs more or less protection and work on those documents should be pursued in another room that is equipped with different access rights.

In summary, our approach goes beyond current approaches by supporting a large range of group formation schemas, and providing simple-to-use access rights management via keys.

4. EXPERIENCES

According to our experiences, end-users can use our approach to successfully collaborate and create joint artefacts (see [7] for more details on collaboration in CURE). In order to evaluate whether end-users can use and do use our approach to form groups and manage access rights of their shared workspaces, we observed the usage of the system in the winter term 2003/04 in our university.

4.1 Method

Setting. Our university is the German distance learning university. Teaching includes different forms of education: courses, seminars, and different forms of practical problem solving in lab courses. Course material and accompanying individual exercises are sent to distributed students via surface mail or the Internet. Course-specific newsgroups and direct e-mail communication with professors and teaching assistants support asynchronous discussions. Students may also use the telephone to contact staff directly. In addition, students can meet tutors and other students at one of the 60 learning centers.
We evaluated our system in three different distance learning courses: a virtual seminar in psychology, a one term project-based course in computer science and tutor-guided learning groups in a course on linear algebra. Students were studying from home or office using the Internet and our system. Teachers from our university did communicate with their students via Internet and our system. While the virtual seminar and the tutor-guided learning groups were based on distributed learning, the project-based course used a form of blended learning. Here, groups would collaborate at a distance, then meet for 3 days at our campus, followed by distance collaboration during the term, until a final presentation meeting at the campus again.

In addition to these settings, all students of the university could use the system in the context of other courses. Examples of groups that formed on their own are exam preparation groups for business administration, discussion groups of the students’ self-administration body, or project groups. These groups did not receive any administrative help from the system administrators.

**Design.** We observed the system usage in the three courses mentioned above during the first 120 days (4 months) of system usage, and also looked at the overall use of the system by the entire user population during that time.

**Subjects.** We did not select subjects on a controlled basis. Rather, we deployed our system in three existing courses. Students did enrol in these courses as part of their regular studies. Students at our distance learning university are mostly employed elsewhere and thus studying part-time from home or office.

Teachers are regular professors or teaching staff at our university with experience in e-learning over the Internet (e.g. using Mail, Newsgroups, WWW, BSCW, IRC).

**Procedure.** In the above three courses, teachers were introduced to our system by the development staff using some scenarios. After initial training, teachers developed their learning environment (a set of rooms and materials) on their own. Then, teachers prompted students to use the system. Students did not receive any training. Rather, they were pointed to the online system manual, which contained a detailed introductory scenario and a reference section.

The groups mentioned above that formed on their own received no administrative help from the system administrators. Students and project groups independently found out about our system and began to use it.

**Measures and evaluation infrastructure.** Due to the distance learning setting, it is difficult to conduct direct observational studies (e.g. taking videos). Instead, we regularly conducted interviews with the teachers present at our university and with some student groups, when they were present at our campus. In order to get data about the user interactions, all user actions were logged in a log file, which contains information about each web request (each click) of the users. Since the basic system architecture is a web application, we can restore the full interaction from the log files.

We observed the use of the system with a focus on group formation (i.e. initial creation of rooms, initial creation and distribution of keys) and tailoring access rights (i.e. later creation of sub rooms, later creation and distribution of keys).

Measures taken from log files were computed based on event types and time stamps. Measures resulting from interviews are reported as anecdotal evidence.

### 4.2 Results

We analyzed the data with respect to two questions:

1. **Group formation.** Were the end-users (i.e., teachers and students) able to form groups? Did they form groups in a flexible way (i.e. with changing group membership during the life cycle of a group)?

2. **Access rights management.** Were the end-users able to manage and tailor the access permissions of their shared workspaces themselves?

**Group Formation.** 36 students enrolled in the virtual seminar in psychology. The teachers formed 8 groups of 4-5 students working on the same topic. For each group, the teachers created a room and assigned keys to the respective students. The teachers created keys that could not be copied or forwarded, so that only students of the group could access the group’s room. Students did not want the teachers in their rooms, the teachers deleted their keys to the student rooms within the first weeks of the seminar.

36 students enrolled in the project-based course in computer science. Teachers created an initial room structure consisting of the course room and a project ideas room. Before the first face-to-face meeting at the campus, students were required to generate project ideas and to make sure that similar ideas were merged into one proposal. During the first face-to-face meeting, students selected 6 projects and formed groups of 6 students, created a group room in our system, and assigned keys to the six group members as well as two more keys with restricted rights to two teachers acting as advisors. The teachers’ keys allowed them only to delete or return their key, to enter the room, and to edit its content. Two groups created further rooms to organize their group work. The teachers did not get access to these private group workspaces. In order to support collaboration between groups on overall issues (such as the technology used by several groups), the teachers created plenary discussion rooms for these topics.

The tutor-guided learning groups in algebra were organised as a voluntary offer for students of the respective course on algebra. Thus, the tutors created a room for each learning group offered, and made 10 keys available at the door of each room. Students took a key when they wanted to become a group member. This strategy limited the group size to a manageable number.

From the log files, we could see that in the other settings, rooms were created by self-organized student groups and project groups within our university. In these settings, we could also observe changing group membership.

Interviews with the teachers and students of the three above courses and the university-wide working group indicated that they understood the concept of rooms and keys and could apply it to create the groups and access permissions they wanted.

Analysis of the log files confirms that, in general, rooms were created and keys were distributed in different ways to users. From the 1127 keys generated for the 257 rooms during the study period, 50% were made available at the door of a room, 30% were explicitly assigned to users, and 20% were requested by users from room’s management team.

**Access rights management.** Access rights management in CURE occurs at two points in time: upon creation of the room, and when
later adaptation is needed. When a room is created, the creator creates keys for the users of the room. By defining the corresponding key rights, the room creator determines the initial access permissions of the (later) key holders.

All settings showed that the end-users were able to create rooms and distribute keys to users. From the three courses, only the project-based course did show students creating their own rooms. In the other two cases, teachers did set up the initial room structure. However, when looking at the exam preparation case and the project group case, we see that lay-users created rooms and managed to create appropriate keys. Our interviews with the members of the project-based course and the project group showed that they had no trouble in setting up the room structure and creating keys.

We analyzed the log files and the actual state of the objects with respect to the types of keys generated, and how these have been distributed (which and when). We can see that in most cases end-users (i.e. teachers, students) created keys with full permissions. This enables users to implement simple collaboration scenarios, e.g. based on mutual trust. Users used this strategy without reporting any problems. Some groups did show more elaborate use of key rights, such as some groups in the project-based course that invited teachers into their rooms, but giving them only restricted permissions. We saw the same in the project group case. The keys created in these cases showed interaction rights that were fairly specific (i.e. some users got different rights than others) while the room-rights were mostly at maximum level (i.e. most users were allowed all actions).

This led us to a closer look at the learning scenarios employed by the three courses. None of the didactical approaches used in these courses required users to change access permissions at a later stage. Consequently, these operations were not needed. However, in the cases of exam preparation and project work, we could observe the use of more restricted keys.

4.3 Discussion

Our results indicate that end-users were able to form groups in CURE. Different forms of group formation were used in the different settings. While group membership did not change in those courses organized by teachers (e.g. seminar and lab course), the more dynamic cases such as exam preparation, tutor-guided learning groups and project groups showed many changes in group composition over time. Our observations and the interviews with the members of the project-based course and the project group provide some evidence that end-users understand the concept of rooms and keys and can apply it to form groups.

Regarding group work, we found groups in all settings used the room-specific communication means and shared documents. Cooperative work patterns between groups seem to vary a lot. Nevertheless, all teachers confirmed that students did successfully produce results, and the interviews show that students and teachers express positive attitudes about CURE’s usefulness in organizing and supporting collaborative learning.

With respect to end-users managing and tailoring the access permissions of their shared workspaces, we found that all groups were able to set up the initial access permissions satisfactorily. In the interviews, users expressed that they felt comfortable with the interaction rights. While teachers/project managers reported that they selected key- and room-rights carefully in order to support the planned course of collaboration, students reported that they said they were not much concerned with room rights. This may point to the fact that the students created the room for the purpose at hand (i.e. interaction) and do not think at that time much about future development of the workspace (i.e. they selected maximum rights). Our results show that for the three courses not many instances of later adaptation could be observed. This may be due to the fact that none of these settings explicitly asked for later adaptations. However, in the more dynamic case of project work, we saw frequent restructuring as the project evolved.

This finding surprised us, as some of this functionality was explicitly requested by teaching staff of the three courses. Why did they not adapt their didactical strategy to employ the new functionality? An explanation could be that teachers may tend to replicate usual ways of teaching, especially under time pressure. When CURE was available for preparing the courses, not much time for experimentation was available. Interviews with teachers confirmed both, their continuing interest in more self-organized forms of learning as well as problems in adjusting the way of teaching to new technology. Thus, we are currently working with teachers on employing new didactical approaches.

5. CONCLUSIONS

Our approach supports group formation and management of access rights in a shared workspace system by end-users. It is based on the metaphors of rooms and keys. Rooms provide shared workspaces. Virtual keys define what permissions the key holder has in a room on three dimensions: key-rights, room-rights, and interaction rights. Groups are defined as the set of current key holders of a room. Group formation by end-users is supported via different schemas for key distribution and request. Access rights management by end-users is supported by operations (re)defining the set of keys of a room, which allows flexible tailoring of access permissions. Our experiences in the context of project work and different forms of distributed collaborative learning in university courses show that end-users are able to create shared spaces, form groups, perform collaborative work, and manage the access permissions of their shared spaces.

Our approach exceeds current approaches in group formation by supporting a spectrum of forms of group formation ranging from assignment to invitation to free enrollment to confirmed enrollment. Public rooms are also supported. The approach differs from other access control approaches by combining the include relationship, waiving of rights, negotiating access rights, and discussing access rights.

Our model was developed and tested for the application domain of collaborative learning and also proved to be valuable for project work in virtual teams. To adapt our approach to other application domains, six steps have to be conducted:

1. For meta-access control in the domain, the key-rights and operations on keys that should be made available to the users have to be defined, e.g. if returning keys in the application domain is not meaningful, than no such key-right should be offered.

2. The right groups for the basic access rights have to be determined. Our model supports room and interaction-rights. For example, we do not provide coupling-rights as in [17], since the cooperation modes are defined in our room model. Another model of a shared workspace may require different groups of rights.

3. The rights for each group have to be specified and ordered
along a meaningful inclusion relationship respecting the imply relationship. For example, in our model all users may inspect the properties page of a room, revealing all users of that room. If more information security is needed an extra right for inspecting the room properties could be meaningful to restrict the access to this information. Then, a room-right “inspect room properties” should precede the room-right “enter room” (cf. figure 1).

(4) Another design option is about the subjects and objects keys should link. In our model, a key regulates the access of individual users to a room. Our approach could also be used to support more static or organizational notions of groups. Keys could also be granted to organizational groups, i.e. each member of such an organizational group could automatically get an appropriate key. Instead of equipping rooms with a lock, one may also consider equipping documents with locks, i.e. keys may regulate the access to an individual document. In any case, the designer should take into account the amount of specification work imposed on the end-users by the design decisions.

(5) Concepts from other computer science domains like concurrency control (e.g. version support) or conflict resolution [21] complementing access control should be considered.

(6) The chosen user interface should “smoothly mesh the access model with the user’s conceptual model of the system” [5].

From fall 2004 on, CURE will become a standard means for collaborative learning at our university. CURE is available for public non commercial use at http://cure.pi6.fernuni-hagen.de.

6. ACKNOWLEDGEMENTS
We gratefully acknowledge the valuable comments of the reviewers, and constructive discussions with Volker Wulf.

7. REFERENCES


[18] Stiemerling, O., Wulf, V.: Beyond ‘Yes or No’ – Extending Access Control in Groupware with Awareness and Negotiation; In Group Decision and Negotiation; Vol. 9, 2000, pp. 221-235.


