

# The Public Privacy – Patterns for Filtering Personal Information in Collaborative Systems

Till Schümmer

Computer Science VI - Distributed Systems

FernUniversitaet in Hagen

Informatikzentrum, Universitaetsstr. 1, D-58084 Hagen, Germany

till.schuemmer@fernuni-hagen.de

## Abstract

Privacy, isolation, and fairness are important issues for virtual communities and collaborative applications that are on the other hand hard to establish. This paper presents six patterns that focus on how to establish boundaries for interaction. The patterns address issues regarding information perceived from and transmitted to other users. They discuss the importance of shielding the user from too much information in specific situations and the need for controlling how much information is provided by the local user.

Note to the writer's workshop participants: The following pattern language exceeds the length that can be reasonably discussed in one writer's workshop session. Thus, I'd be happy to receive feedback on the patterns ATTENTION\_SCREEN<sub>-8</sub>, BUDDY\_LIST<sub>-13</sub>, and RECIPROCITY<sub>-21</sub>. Additional feedback is welcomed in the coffee breaks ;-).

## 1 Introduction

Interaction in communities often takes place in subgroups of the community. Consider for instance the welcome reception at the CHI conference where several hundreds of scientists gather. While the opening plenaries are conducted in a lecture style (where the interaction is unidirectional), the receptions encourage people to group in small subgroups and exchange ideas. For the nature of the reception, people wander around and change their communication partners frequently. They join in other groups and engage in the communication if they feel a common topic.

Goffman (1963) studied the interaction in gatherings that take place in public places. He identified the concept of communication boundaries that provide social seclusion for the group. These boundaries are physical in nature providing real separation of groups of people. If the interaction takes place at one physical location, Goffman identified the concept of *Conventional Engagement Closures* that provide a comparable boundary based on social norms. For instance, people would visually

express their attendance so that other people can clearly identify in which encounter a person is engaged.

Providing comparable mechanisms for identifying and controlling engagement in groupware applications and online communities is a difficult task that has been addressed in several research prototypes. But besides research systems only a few number of applications address the above issues.

## 1.1 The Pattern Language

Within this paper, six patterns will be presented that discuss different mechanisms for providing engagement closures in the above sense. All patterns have in common that they help to control what kind of information is made visible for other users and what information is perceived by the local user.

The patterns can be divided in two groups: patterns that filter personal information that is produced by the local user and thus help to provide privacy and patterns that filter the personal information received from other users and thus help to reduce information overload.

MASQUERADE<sub>-4</sub> is a pattern of the first group that controls how much of the local user's actions are visible to remote users.

ATTENTION SCREEN<sub>-8</sub> is a pattern that defines what signals for attention are processed from other users. It is extended by the BUDDY LIST<sub>-13</sub> pattern that helps to filter users from the user community based on a previous knowledge. BIRDS OF A FEATHER<sub>-17</sub> also helps to filter signals from remote users by grouping people with comparable interests.

Finally, two more patterns connect the above groups: RECIPROCITY<sub>-21</sub> ensures that users contribute to an encounter when they benefit from the encounter. WHO'S LISTENING<sub>-25</sub> reduces the user community to all users, who are currently engaged in the encounter including lurkers and free-riders.

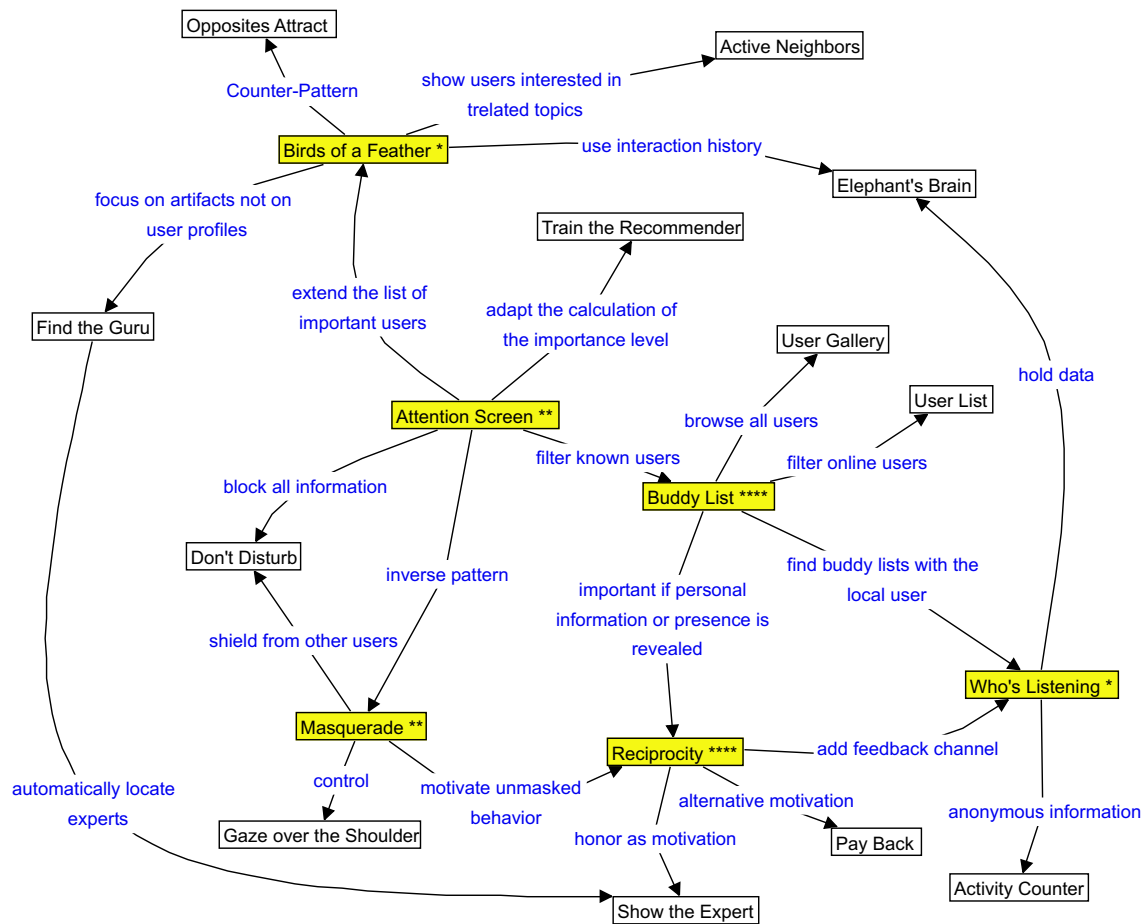
The pattern map in figure 1 outlines the relation between the patterns and shows additional patterns that are related to the six patterns of this paper but not presented in a long form.

The individual patterns will be presented in the next chapter. Thumbnails of the additional patterns are provided in the last chapter of this paper.

## 1.2 Structure of the Individual Patterns

Each pattern is presented in an Alexandrian form. The pattern name appears as a section title followed by a star rating that shows how common the pattern is. A pattern can have between one and five stars. One star denotes a good solution that solves the problem. It can be one solution among many other good solutions. Five stars on the other hand denote that the pattern seems to be an omnipresent solution and that there probably is no better solution available. Two to four stars denote pattern between these two extremes.

Each pattern begins with a scenario that should sensitize the reader for the problem. This is followed by the intent and the context of the pattern. It helps



**Figure 1:** The patterns presented in this paper.

the reader to decide whether or not the following pattern may fit into his current situation.

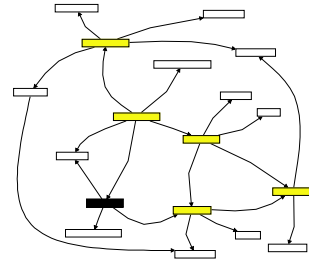
Then follows the core of the pattern composed of the problem and the solution statement separated by a diagnosis section that provides clues how to detect the problem. Other pattern authors have referred to this section as forces.

In the remaining sections of the pattern, the solution is explained in more detail. The collaborations section names all components that interact in the pattern's solution and how the interaction takes place. The rationale section argues why the pattern's solution resolves the conflicting forces and the danger spots section lists counter forces that could complicate the pattern's application. Since a pattern should have been applied by many developers before, the known-uses section elaborates on the experiences with the solution by showing different applications of the pattern.

The related patterns section concludes each pattern by pointing to other patterns that may be relevant in the context of the current pattern.

## 2 Patterns for Filtering Personal Information in Collaborative Systems

### 2.1 MASQUERADE \*\*



*Alice, Jane, and George are a group of math students who prepare for their examinations. They meet in the cafeteria of the university to discuss problems of the course. Within their meeting, it becomes obvious that George has large deficits in calculating the product of two vectors. Although Alice and Jane tried to explain this topic to him, he was not fully confident on the topic at the day of the examination.*

*Now consider Jack, an evil professor who is only interested in finding weak points in his students' knowledge. Jack was informed that a group of students would meet in the cafeteria for their preparation and thus corrupted the university's security division to pass on the observation tapes of the cafeteria. And fortunately, he finds a tape with Alice, Jane, and George and uses this tape for the preparation of the examination. In the examination, he asks George to calculate the product of two vectors and, as foreseen, George is not able to perform this task. George fails the examination and Jack is satisfied.*

*You might think that this story is absurd. But think of a public internet-based discussion group (e.g. a newsgroup) for a course where all contributions can be associated with a specific user. In this case it is very easy to get an impression of all participating users and identify their strengths and weak points. Can a learner in this case be sure that this information will not be used against him?*

AKA	Anonymous Interaction
Intent	Control how much private information you reveal to other users when interacting in a collaborative environment.
Context	You are working in a monitored environment (e.g. an environment

with GAZE OVER THE SHOULDER<sub>→30</sub> in place), where personal information such as mail addresses or the work status are visible for other users of the environment.



**Problem**      **User monitoring is required for providing awareness information to remote users or associating work results with a specific user. On the other hand users often do not act as confident if they know that they are monitored as they would act in an anonymous environment.** Especially openness and the courage for taking risks may be much lower.

**Symptoms**      *The problem becomes obvious when ...*

- personal information is abused by some users.
- after the personal information was abused for the first time, the users drastically reduce the amount of information that they provide to the system. This results in a lack of collaboration.
- potentially secret information can be related to specific work episodes and the user knows that he does not want to be recognized before he enters this work episode.

**Solution**      **Therefore: Let the user control how much interaction information he provides to the system.** This means that the user should be able to filter the information, which is revealed from his personal information. Remember to consider RECIPROCITY<sub>→21</sub>.



**Collaborations**      While interacting within the collaborative environment, the local user can enter a desired level of publicity or a publicity profile. The difference between a publicity level and a profile is that the publicity level arranges all possibly revealed information on a scale. Selecting a specific publicity level implies that all information with smaller publicity levels will also be revealed. In contrast, a publicity profile will allow the user to select each aspect that he wants to reveal.

Monitoring systems (cf. GAZE OVER THE SHOULDER<sub>→30</sub>) such as sensors or state trackers in client-server systems reduce the perceived information according to the user's publicity settings. The reduced information is then forwarded to a central ELEPHANT'S BRAIN<sub>→30</sub> or to other interested clients so that the information can be visualized for other users.

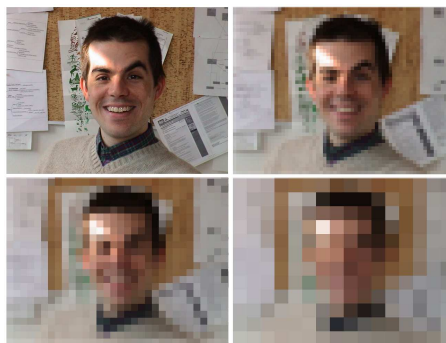
**Rationale**      Since users can explicitly control how much personal information they provide to other users, they do no longer have to fear that their personal information is misused by strangers. This provides them with an environment that is as private as the situation demands

it. The user can decide to discuss private matters without the possibility of being monitored by other users by simply changing his privacy profile.

Using the  $\text{RECIPROCITY}_{\rightarrow 21}$  pattern on the other hand can enforce that a user will reduce his privacy level when the need for privacy is no longer present. This is done by connecting privacy levels with permissions for interaction. Activities that affect common artifacts will for instance demand that the user reveals some personal information to the community. Examples are provided in the  $\text{RECIPROCITY}$  pattern.

**Danger Spots** Anonymous interaction with the system may lower the inhibition threshold for destructive or forbidden behavior. The users do not have to fear that destructive activities are associated with their identity. Thus, you should provide only limited functionality for anonymous users (e.g. only read access or only moderated postings to a discussion board).

**Known Uses** **Video systems:** NYNEX Portholes (Lee, Girgensohn, and Schlueter 1997) is a video system in which users can blur their video image to communicate less detail to other users. Zhao and Stasko (1998) evaluated different video filters for video awareness systems. Users were able to decide, which filter should be applied to their video image. One example of such a filter is the pixelize filter (cf. fig. 2), where the image is reduced in its resolution. By these means, the users were able to provide less detail about their current status. The authors found out that users who where interacting often were still able to draw presence information from very distorted images, while others could not make much sense out of these images anymore.



**Figure 2:** Different levels of distortion in a pixelized video image.

Boyle, Edwards, and Greenberg (2000) present a video link awareness system that allows the users to blur their images (using filtered video streams) or to block the image totally

(projecting an image of a palm in front of the video – cf. fig. 3).



**Figure 3:** Blocking Video using a projected picture of a palm (Boyle, Edwards, and Greenberg 2000).

**TUKAN** (Schümmer and Haake 2001) is a collaborative software development environment that lets the user decide how much awareness information he wants to provide to other users. The options range from process awareness, where the user publishes information about his current programming task over change awareness, where the developer informs other developers on changes that he performed in the project up to the presence level, where the current position in the project (the method that a user browses) is published to other users.

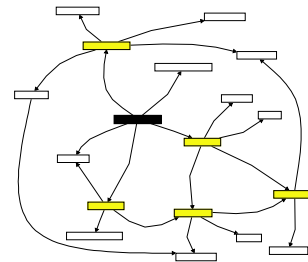
**Anonymous access** is provided by many web-based community systems or file servers. Users can access information using a public user identity. Access is in most cases limited to read access.

**Related Patterns** **DON'T DISTURB<sub>→30</sub>** shields the user from being contacted by other users. **Masquerade** can fulfill the same purpose if it is used in a way that the user interacts with the system anonymously and users can only be contacted if they are registered with an identity.

**GAZE OVER THE SHOULDER<sub>→30</sub>**: The **MASQUERADE** controls which information will be tracked by information trackers. The information trackers are described in detail in **GAZE OVER THE SHOULDER**.

**RECIPROCITY<sub>→21</sub>** is very important if users can perform secret actions. Without reciprocity, users may consume presence information from other users without revealing this information personally. One could imagine that one user always acts in a **MASQUERADE** which would prevent that the user establishes relationships to other users.

## 2.2 ATTENTION SCREEN \*\*



*Imagine – or better remember – a classical concert. Some of the visitors carry their mobile phone with them. While the orchestra is playing one of the most beautiful passages of the music, a telephone rings and catches everyone’s attention.*

*This is annoying.*

AKA	Message Filter.
Intent	Define, who and what may take the user’s attention.
Context	Other users provide information to a shared environment and you are consuming this information. Other users may feel the need to get in contact with the local user. To do so, they send information to the local user that catches his attention.



Problem	<b>Every request for attention needs to be processed by the user. Thus, it already takes some of his attention. But in situations, where the user needs to focus his attention on other things, this is disturbing.</b>
---------	---

Symptoms	<i>The problem becomes obvious when ...</i> <ul style="list-style-type: none"><li>– information that is unrelated to the user’s current task is pushed in the foreground.</li><li>– meetings are interrupted by unrelated topics (e.g. a ringing telephone or a colleague dropping in and looking for a social chat).</li><li>– users are disturbed by requests for attention and have problems to focus on their original topic again.</li><li>– users are contacted by people that they don’t want to interact with in the current situation.</li><li>– users are flooded with information that is not valuable for them.</li></ul>
----------	---

Solution            **Therefore: Enable the user to filter the information which reaches him.** Use meta-information (e.g. sender details) or content information (e.g. important keywords) to distinguish important information from not so relevant information. **Collect the less important information at a place where the user can process it on demand and forward relevant information directly to the user.**



Collaborations    The *local user* defines *rules* for *incoming attention request* (also referred to as message). The rules define, which attribute of the incoming request should be considered. A rule also defines, which values lead to which level of attention. A rule could for example check the sender field of an e-mail message and calculate an importance value based on the sender's address. Methods for calculating the importance value include for instance the comparison of an address with addresses stored in a BUDDY LIST<sub>-13</sub>. In the easiest case, the importance value is binary and indicates whether or not an action should take place.

Whenever an attention request is received, the local user's system checks the rules and weights the received request with the corresponding importance value. According to the importance values, the local user's system decides what action needs to be performed with the attention request. Again, the easiest decision is to decide whether or not any action should take place. Examples for actions that can be associated to an importance level can be

- specific audio signals or other ambient awareness information (Wisneski, Ishii, Dahley, Gorbet, Brave, Ullmer, and Yarin 1998),
- more or less obtrusive ways for propagating the message to the user,
- storage of the message in specific storage areas that can be perceived by the user when the user has no other important task, or
- the removal of the request.

Rationale            DeMarco and Lister (1999) already pointed out that interruptions can drastically reduce the user's productivity. They claimed that it takes approx. 15 minutes to reach an ideal productivity level. A five minutes interruption would thus result in 20 minutes of reduced productivity. Jackson, Dawson, and Wilson (2003) studied the effects of interruptions by e-mail messages. They observed 16 employees regarding their e-mail behavior and found out that the recovery time is much lower for e-mail interruptions (64 seconds). But even with such a short recovery time, it is still a significant factor.

Rules reduce the number of messages that are propagated directly to the user. This leads to fewer interruptions and keeps the local user focussed to his current task unless there is an important event that requires the user's attention immediately. The effect of the recovery time is reduced because users can actively decide to receive the messages when they are currently idle.

On the other hand, important messages are not delayed. This reduces the fear for missing an important request.

**Danger Spots** The definition of rules can get too complicated for naive users.

Rules may classify important information wrong so that it may be ignored by the user. Thus, you should not be too rigid when filtering the information. If for instance only requests from users that are on the local user's BUDDY LIST<sub>→13</sub> are reaching the local user, he will not be able to establish new contacts anymore. This would be crucial in a community system because members are required to be open for other members.

**Known Uses** **Siemens S55:** Mobile phones like the Siemens S55 allow the user to assign callers to different groups (cf. fig. 4 left). For each group, the user can decide how calls should be signalled by selecting a different tune or no signal at all (cf. fig. 4 right).

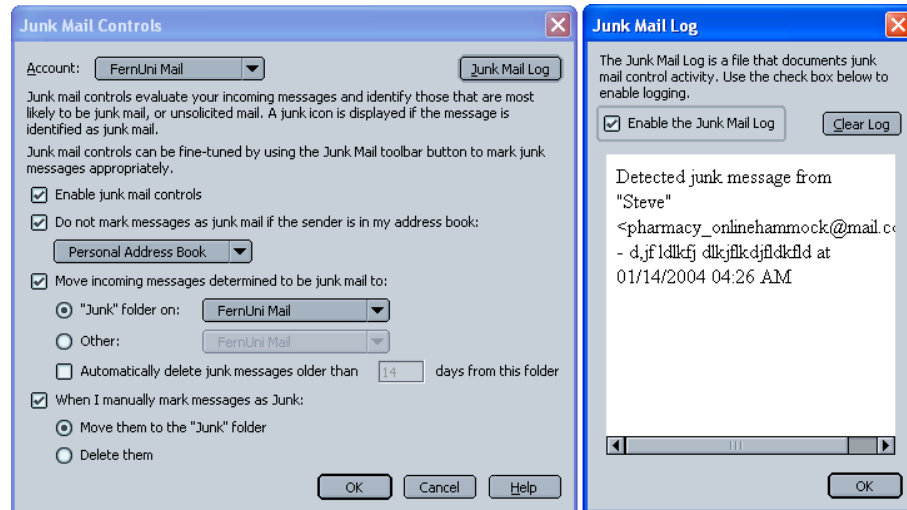


**Figure 4:** The Siemens S55: Different sounds can be assigned to different caller groups.

**Mozilla Junk Mail Filter:** The junk mail filter of Mozilla<sup>1</sup> blocks incoming e-mail messages. Whenever a message is received by the Mozilla mail client, it is analyzed using user-defined filters. The result of this analysis is a decision whether or not the message is a junk message. The user can control the aspects of the filter by specifying attributes that should be considered during the analysis (cf. fig. 5). If the user for instance provides an address book of trustable contacts, mails from these contacts are never considered as spam. Such an address book is called a white list in Mozilla (it is an implementation of the BUDDY LIST<sub>→13</sub> pattern).

Another way to change the filters is that the user provides

<sup>1</sup>The source files for the junk mail filters are available at: <http://lxr.mozilla.org/mozilla/source/mail/base/content/mailCommands.js#474> and <http://lxr.mozilla.org/mozilla/source/mailnews/extensions/bayesian-spam-filter/src/nsBayesianFilter.cpp>. The basic concept of the filtering algorithm is a Bayesian Filtering approach. More information is provided by Graham (2002).



**Figure 5:** Mozilla junk mail filter.

feedback on the filter result (an implementation of the TRAIN THE RECOMMENDER<sub>→31</sub> pattern).

Within the junk mail configuration dialog, the user also defines an action that should take place if a message was classified as junk. Options include the removal of the message, moving the message to a specific folder, or marking the message as junk.

**TeamSpace** (Fuchs, Poltrock, and Wetzel 2001) provides different *working modes* – the individual, the social, and the meeting mode.

Transitions between the different modes are modelled to allow the user to select the mode that is appropriate for his current task.

**Instant Messaging Systems** like MSN Messenger, ICQ, AIM, or Jabber provide modes where only messages from users who are on the BUDDY LIST<sub>→13</sub> are accepted.

**WebWasher**<sup>2</sup> is an example for a system that blocks specific content from the web like links to offensive content, banner advertisements, or specific media types (e.g. MP3 files). Again, the user or the administrator first has to select the filtering criteria. According to these criteria, the WebWasher filters specific responses and prohibits that disturbing content is transmitted to the user.

**Related Patterns** DON'T DISTURB<sub>→30</sub> is a more rigid way than the ATTENTION SCREEN to control what information can reach the local user. If the user decides to be not disturbed, no requests for attention are actively forwarded to him.

<sup>2</sup><https://www.webwasher.com/>

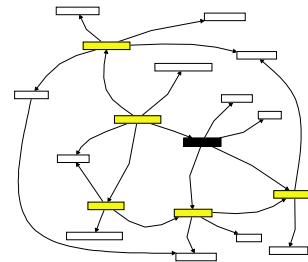
BUDDY LIST<sub>-13</sub> defines the number of users, who are considered as valuable communication partners by the local user.

TRAIN THE RECOMMENDER<sub>-31</sub> can be used to adapt the filters of the ATTENTION SCREEN that are used to calculate the importance level. The idea is that the user can correct the provided importance level and that the calculation algorithm takes this corrected importance level into account when performing the next calculations.

BIRDS OF A FEATHER<sub>-17</sub> can be used to extend the list of allowed communication partners. If two users are sharing many preferences, they are more likely to donate some of their attention to each other.

MASQUERADE<sub>-4</sub> provides means for interacting invisibly in the system. Whenever attention requests are based on the notion of co-presence (e.g. in instant messaging systems), other users will not be able to ask for the local user's attention since they assume that the local user is not present.

## 2.3 BUDDY LIST \*\*\*\*



*Consider Maria, an inhabitant of Munich. She knows many people living in her vicinity. One day, she decides to invite all of them to her birthday party. But since she knows the other people only by name, she has to look up all their addresses. She takes a look at the telephone book and looks up her first contact called Alfred Meier and is shocked by the 1280 people called Meier in Munich. It takes her some time to find Alfred but even then she finds 11 people called Alfred Meier in Munich. Thus, the number of unimportant Meiers is much too large and Maria spends a long time to look up all neighbors that she wanted to invite to her party.*

- AKA roster (in Jabber), contact list (in MSN or ICQ), address book
- Intent Show only selected known users.
- Context You are using an interaction space like a communication channel, a groupware application or a collaborative virtual environment together with many other users.



- Problem **When many users are able to interact in the interaction space, it is hard to maintain an overview of relevant interaction partners since the number of users exceeds the number of relevant contacts for a specific user.** User lists grow very large and it is hard to find people who the local user knows. On the other hand, the local user is often only interested in those people who he knows.

- Symptoms *The problem becomes obvious when ...*
- users spend a long time searching for another user.
  - the system may be used by more than one user with a specific name who only differ in their address or login. Remembering logins on the other hand is considered difficult by the users.

- Solution **Therefore: Provide buddy lists, where a user can enter**

**other users who are of interest to him. Whenever the local user browses other users, first show only users from the buddy list.**



**Collaborations** Whenever the *local user* interacts with *another user*, the local user can add the other user to his *buddy list*. The buddy list is a set of user objects. A user can be added to the buddy list by selecting his representation in the user interface and executing a command (e.g. a menu item associated to the user object).

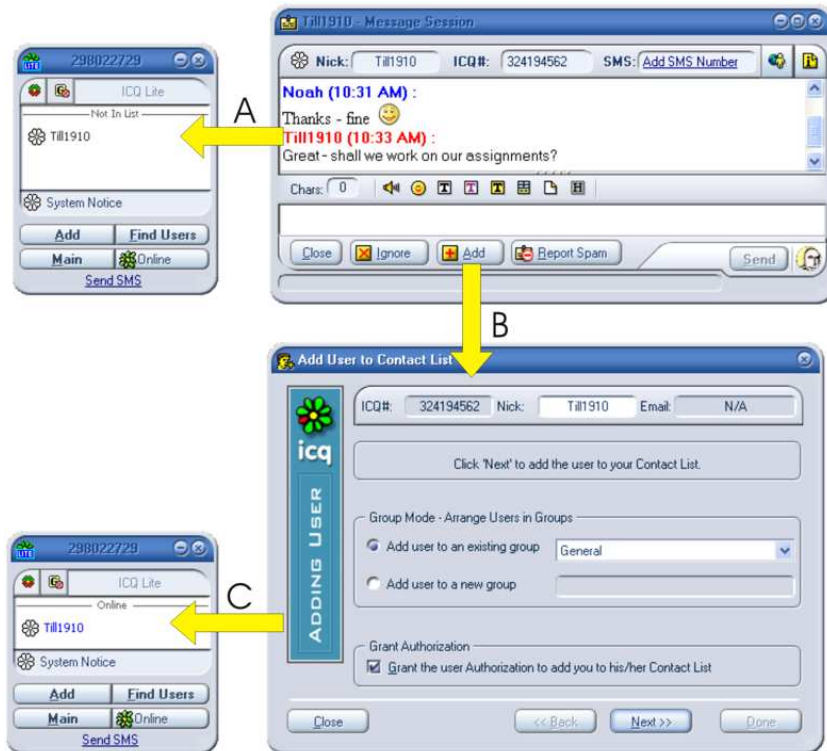
**Rationale** The main reason why this pattern works is that it eases the process of finding other users by storing these users in a personal list. Compared to public directories of users, the personal list only contains the users that are important for the local user. This will reduce the number of name conflicts and enlarge the hamming distance between two names in the buddy list (it is less likely that two users with close names like *Alfred Meier* and *Alfred Meyer* are both buddies of a user).

Connecting the means for adding users to the buddy list with the user's representation (or the interface elements that are used to interact with the other user) makes the process of adding a user to the buddy list intuitive and reminds a user to consider adding the user.

**Danger Spots** If users only consider buddy lists for maintaining contacts to other users, they will hardly find new users in the system. Thus you should ensure that users can also browse other users who are not on their buddy list (e.g. by providing a `USER GALLERY`<sub>→32</sub>).

**Known Uses** **Instant Messaging Systems** like MSN Messenger, icq, AIM, or Jabber all provide buddy lists. Presence information is retrieved from the presence server only for those users, who are on the buddy list.

Figure 6 shows how a contact can be added during a conversation with icq. If the local user allowed other users to contact him, they can directly talk to the local user by sending him a message. Note that the remote user has to retrieve the address of the local user by other means than the buddy list since the local user is not yet on the remote user's buddy list. The remote user will be shown as a user "Not in List" (cf. fig. 6-A) on the buddy list for the time of the chat. If the local user feels that the contact is valuable, he can add the remote user to the buddy list by pressing the "Add" button, which will open the "Add User" dialog (cf. fig. 6-B). The local user can provide a nickname for the remote user under which the remote user will appear in his buddy list. Note that the "Add User" dialog by default checks an option so that the



**Figure 6:** Adding a contact to the buddy list in icq.

remote user will be able to add the local user to his buddy list as well. This is important to maintain  $RECIPROCITY_{-21}$  although it can be bypassed (by unchecking this option).

From then on, the remote user will be shown on the buddy list with his current status (fig. 6-C). The local user can from then on get in contact with the remote user easily without looking up this user's icq number.

**e-Mail address books:** Mail clients allow the local user to interact asynchronously with a remote user by reading messages from the remote user or sending messages to a remote user. Together with the message, most systems show information regarding the sender of this message (cf. fig. 7). The sender can be selected and added to an address book, which will make future interaction easier since the local user does not have to remember the remote user's address.

Related Patterns  $USER\ GALLERY_{-32}$  provides means for browsing all users in the system. As outlined in the safety rules, a  $USER\ GALLERY$  complements a  $BUDDY\ LIST$  and opens the opportunity to meet new community members.

$RECIPROCITY_{-21}$  is important if buddy lists reveal much personal information. In this case, a user should be able to control who can put the user on his buddy list. On the other hand, if the local user adds a remote user to his buddy list, the remote

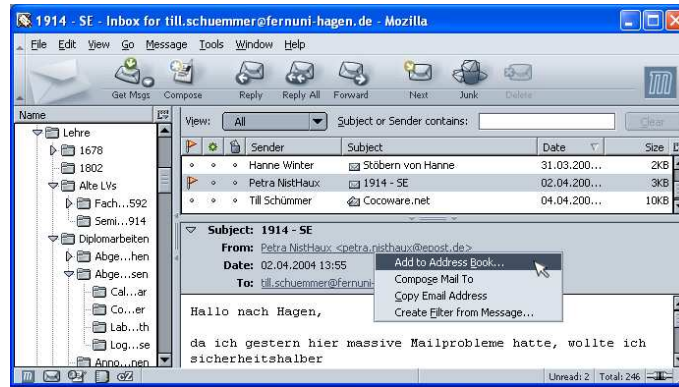


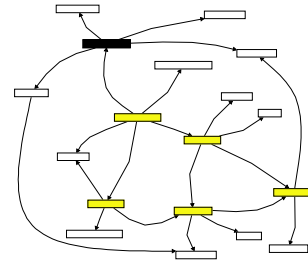
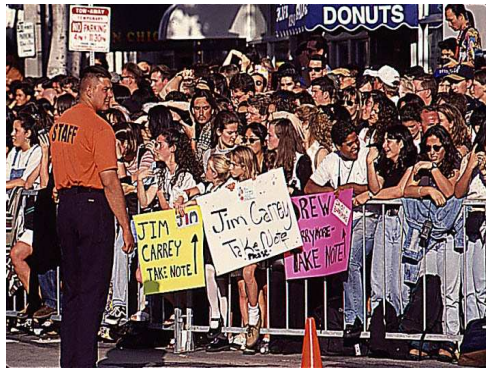
Figure 7: Adding a user to an e-mail address book.

user should also be allowed to add the local user to his buddy list.

WHO'S LISTENING<sub>→25</sub> can be used by the local user to find out on which other users' buddy lists he appears.

USER LIST<sub>→32</sub> provides another filtering on the set of users: it shows only those users, who are currently logged in, whereas the BUDDY LIST shows only known users regardless of their current status. Both patterns are often combined to provide awareness on buddies.

## 2.4 BIRDS OF A FEATHER \*



*Harry was lazy, and although he had nothing else to do but drive his goat daily to pasture, he nevertheless groaned when he went home after his day's work was done. [...] One day, he] seated himself, collected his thoughts, and considered how he could set his shoulders free from this burden. [...] "I know what I will do," he cried, "I will marry fat Trina who has also a goat, and can take mine out with hers, and then I shall have no more need to trouble myself."*

*So Harry got up, set his weary legs in motion, and went [...] to where the parents of fat Trina lived, and asked for their industrious and virtuous daughter in marriage. The parents did not reflect long. "Birds of a feather, flock together," they thought, and consented.*

*From: Jacob and Wilhelm Grimm, "Der faule Heinz".*

AKA	Expertise selection (Yiman 2000)
Intent	Find the users who have most in common with the requesting user.
Context	Users interact with artifacts in a shared information space. The artifacts refer to specific topics.
————— ◇◇◇ —————	
Problem	<b>If people don't know one another, it is hard to decide who could be a good partner for a collaborative activity.</b> For co-located situations, humans have developed intuitive strategies (based e.g. on visual clues) that help them to select, whom they should contact if group formation is needed. In distributed work environments, the presence of other users is often reduced to their user name. This makes it hard to find another user for a collaborative activity.
Symptoms	<i>The problem becomes obvious when ...</i> <ul style="list-style-type: none"> <li>– users are unsatisfied with the selection of their interaction</li> </ul>

partners because they did not know them well enough in advance and the interests are too different.

**Solution**            **Therefore: Compare user profiles or interaction histories to identify two users who share big parts of their history. Propose these users as collaboration partners.**



**Collaborations**    Users perform activities on artifacts in the community. These activities are logged and used to create a profile of important artifacts for the user. Artifacts that are accessed frequently or recently are considered as the user's interests.

When the local user is seeking for a collaboration partner, the system compares all user profiles of remote users to find those profiles of that correlate best (that have many similar interests). Birds of a feather are users who have the best matching profiles. They are recommended for the local user as possible possible collaboration partners.

**Danger Spots**     Recommendation systems in general have the tendency to affect the local user's attitudes. As Cosley, Lam, Albert, Konstan, and Riedl (2003) found out, users rate recommended items differently if they knew the recommendation in advance. This may also have an effect in systems that recommend collaboration partners although there has not yet been any investigation of this hypothesis.

**Known Uses**        **MEMOIR** (Pikrakis, Bitsikas, Sfakianakis, De Roure, Hall, Reich, and Hill 1998) is a system that monitors the users' web browsing activities to recommend other users with related browsing histories. For each user, the addresses of all web pages that he visited are stored in a web trail. The web addresses form the user's interests. Whenever a user is interested in users with a comparable interests, the system calculates birds of a feather that have a high overlap in the browsing history. These users are then recommended for collaboration.

**Autonomy CEN** (Autonomy 2002) is a knowledge management portal that manages collaboration and expertise networks. Users are monitored regarding the documents that they read or author within the document repository. These documents form the interests of the user's profile. If users feel the need to collaborate with other users, they can query the system for users with close interests. Closeness in this case means a close  $\text{SEMANTIC DISTANCE}_{-31}$  between the relevant documents.

**Yenta** (Foner 1996) is an agent-based system to support match-making. An agent observes the local user's mail, news and file consumption and creates user profiles based on the user's these artifacts. As in the case of *Autonomy CEN*, similarity is calculated as a semantic distance between the two artifacts.

Since all artifacts are textual, this distance can be calculated using linguistic approaches.

The unique part in Yenta's architecture is the way how profiles are compared since each profile resides under the local user's control. Yenta uses the concept of autonomous agents. Each user provides 'his' agent with a profile for that he wishes to find a matching user. The agent then contacts other agents at other clients. Both agents compare their profiles and see if they could match. The agent finally presents to 'his' user the set of other agents that match well.

Note that the local user first will only be able to reference the other users' agents. Whether or not the identities of the other users is revealed can be subject to further negotiation between the agents. It could also be possible that the users communicate anonymously through their agents.

**BoF-Sessions at Conferences** are events where people with the same interests can come together to discuss a specific topic. They are in most cases announced by stating the theme or the attribute and other participants of the conference who identify with the theme join the event. Although matchmaking is not supported by any technology, BoFs share the same flow of interaction as it was described in the solution of this pattern.

**Collaborative filtering systems** like the book recommendations at Amazon, music recommendations in Ringo (Shardanand and Maes 1995), or the movie recommendation system Movie-Lense (Cosley, Lam, Albert, Konstan, and Riedl 2003) can be considered as known uses of a variant of the BIRDS OF A FEATHER pattern since they do not aim on bringing together users. Instead they aim on recommending artifacts to a user that have been liked by another user with a comparable taste. Internally, collaborative filtering systems first calculate birds of a feather for the requesting user. The recommendation consists out of artifacts that are included in the profile of top-rated like-minded people but absent in the requesting user's profile.

**Related Patterns** **ACTIVE NEIGHBORS<sub>→30</sub>** extends the interests of a user to related artifacts. The **ACTIVE NEIGHBORS** pattern is intended to detect co-present users but can also be used to find users with related interests in their history.

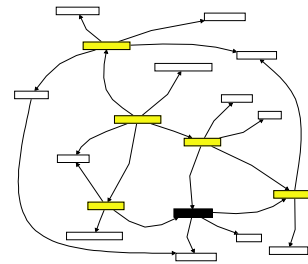
**ELEPHANT'S BRAIN<sub>→30</sub>** logs the users' activities and thus provides the information needed to calculate similarities between the users' interaction histories.

**FIND THE GURU<sub>→30</sub>** only considers the local user's current artifact to find other user's who share a long history with this artifact. While **BIRDS OF A FEATHER** finds people who share a

common interaction history across many artifacts, FIND THE GURU in most cases finds people with different interaction histories and an expertise in an area, where the requesting user is a novice. BIRDS OF A FEATHER thus focusses on users' interests and backgrounds to bring together users while FIND THE GURU<sub>→30</sub> focusses on interaction histories with a specific artifact regardless the history of the local user.

OPPOSITES ATTRACT<sub>→??</sub> is a counter pattern based on the assumption that some tasks can benefit from group members with contradicting preferences.

## 2.5 RECIPROCITY \*\*\*



*Consider a hierarchical organization. Managers had difficulties to bring all their group members together for team meetings because the group members had many appointments with colleagues.*

*At one point in time, the upper management made a binding decision that all employees should use a group calendar. They were asked to include all their appointments in the calendar. The management from a higher level in the management hierarchy could add appointments to the calendars of members of a lower hierarchy level.*

*Such a solution provides a large potential for conflicts because the upper management benefits from the calendar much more than each employee. Managers gain new freedom (the possibility to control and modify their group members' schedules) while each employee has to spend time for adding the appointments to the calendar and loses the freedom of controlling his personal schedule. Appointments that were initially negotiated could now be dictated by the management.*

AKA	Fair distribution of efforts, win-win situation
Intent	Ensure that the users benefit, if they contribute to the system. Let the benefit grow, when the user contributes more.
Context	Your system uses the user's input to produce the group result. To facilitate collaboration, ideally all group members participate in the group process to reach a goal.



Problem	<b>It is easy to agree on participation, if the goal is beneficial for everyone. But in many work situations, some people benefit more than others from a reached goal. In the extreme case, the beneficiaries of the reached goal do not have to participate in the group efforts at all. This leads to a situation, where the people who have to spend efforts on the group result no longer see the need to participate since</b>
---------	--

**the results are not valuable for them.**

Symptoms

*The problem becomes obvious when ...*

- users are not comparably involved although the group process would demand that all users participate to the same extent.
- group members have the feeling that they do all the work and others only act as lurkers or free-riders.

Solution

**Therefore: Establish reciprocity. Ensure that all group members' activities result in an improved group result that is beneficial for all group members again.** Prohibit people to benefit from group results if they are not willing to help the group in return.



Collaborations

Users can interact with the groupware system using specific *functionality*. In the design phase all stakeholders decide for each functionality who benefits from the functionality and who has to spent efforts. This means that they also determine who gains freedom if the functionality is present and whose freedom is reduced.

Identify benefits and drawbacks for the same group of stakeholders that are functionally related and combine them in one collaboration mode. This means that if a user wants to use a functionality that is beneficial for him, he also has to accept or commit to the corresponding functionality that results in additional efforts for him.

Rationale

Motivating people to participate is the key issue in most collaborative systems. Without the participation of the users, the user community soon becomes inactive and a feeling of lacking fairness spreads between those users who contribute to the system and others who just consume group results.

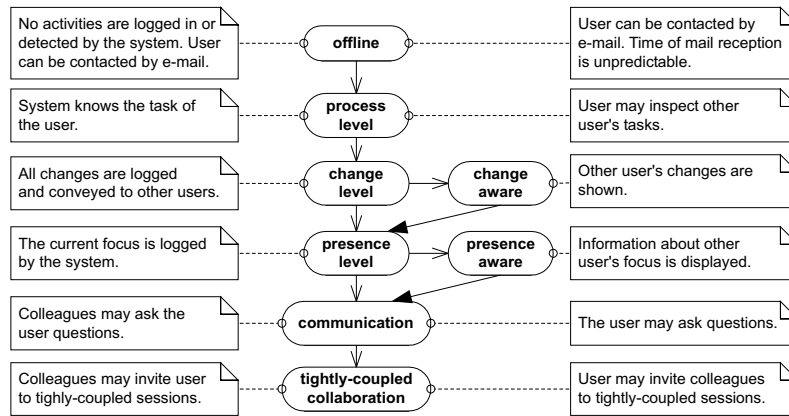
This problem of “social loafing”, where only parts of the group contribute to the group result has been widely discussed in the field of social psychology; compare, e.g., Wilke and van Knippenberg (1996) discussed the relation of group performance and individual participation. It is known to be especially obvious in systems that provide common goods that can be used by anyone. Community systems on the other hand are based on the idea of common goods that are shared in the community.

Early contributions like the seminal work of Grudin (1988) as well as more recent studies like Pipek, Hinrichs, and Wulf (2003) have shown that a lack of equally distributed benefits will be a reason why groupware applications fail.

Finding the inequalities in the design phase involving all stakeholders can reduce the objections for participating to the system since the benefit is made explicit to the end-user.

**Danger Spots** The pattern is only needed in situations, where the critical mass of participation can only be reached with most users participating. If the community is very large (e.g. a news group), it can succeed with a small number of active participants and a larger number of inactive participants (free riders, lurkers).

**Known Uses** **TUKAN:** The collaborative programming environment TUKAN introduced the concept of modes of collaboration (MoC) to ensure reciprocity. A MoC is a lightweight mode, which defines possible collaborative activities. It combines a specific level of privacy (cf. MASQUERADE<sub>-4</sub>) with the right of receiving information about other users. It thus provides a set of predefined ATTENTION SCREEN<sub>-8</sub>s. This combination ensures that a user can only utilize information from other users at a privacy level on which he is also willing to reveal personal information.

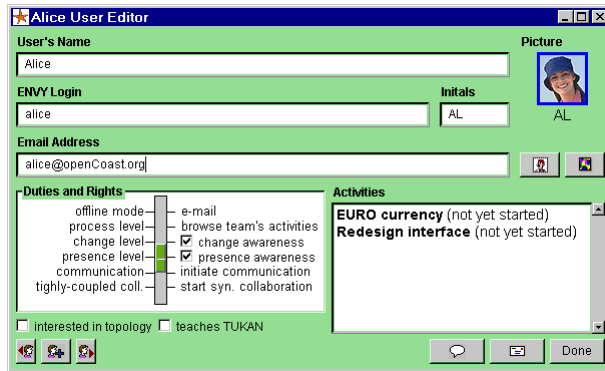


**Figure 8:** Modes of Collaboration in the distributed software development environment TUKAN.

The different MoCs are shown in figure 8. The left side of the diagram shows the drawbacks or limitations in the user's privacy. The right part shows the corresponding benefits from other users. In the middle column, the mode of collaboration is shown that combines the drawbacks and the benefits.

Figure 9 shows how the dependency between modes of collaboration is reflected in the user interface: The slider in the left part of the window can be used to select the appropriate mode. The legend to the slider explains which duties and rights are associated to the mode. In the case of the change awareness and the presence awareness, users can explicitly activate the benefits if they decide to be monitored in exchange.

**Buddy Lists in Instant Messaging systems:** When a user wants to add another user to his contact list, the other user is first asked for permission. If the user rejects this, the other user will not be able to see the user's online status.



**Figure 9:** Reciprocity reflected in the user interface of TUKAN.

Together with this demand for permission, the other user can decide to add the requesting user to his contact list.

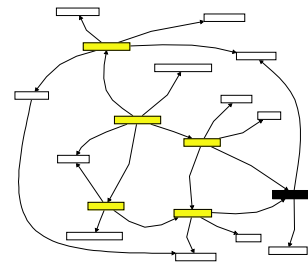
**Bulletin Board Systems:** First bulletin board systems limited the access to the included information by the amount of information that a user provided. To access more data, users had to provide more files to the system. One problem that often occurred in these settings was that users provided *useless* information just for the purpose of improving their download account. Examples are personal artwork that was not valuable for others or randomly generated text files. This example shows that the measure for measuring contribution needs to be carefully designed.

Related Patterns **SHOW THE EXPERT**<sub>→31</sub> can serve as an alternative motivation. If a user is rewarded in a prominent place of the community, he can be motivated to participate in interactions that are of no direct value for him.

**PAY BACK**<sub>→31</sub> argues for providing compensations that are potentially unrelated to a specific functionality to ensure reciprocity also in situations where no related functionality is meaningful that promises benefits for the user.

**WHO'S LISTENING**<sub>→25</sub> introduces reciprocity in classic unidirectional communication protocols by adding a feedback channel.

## 2.6 WHO'S LISTENING \*



Imagine a team of tv program directors, who have the task of designing a new program scheme. They meet in an electronic meeting room to run a collaborative brainstorming session. After identifying first concepts, they start to structure the concepts and create new related documents, in which specific aspects should be further discussed (a hypertext).

While the first phase (collection of concepts) was supported by means of an electronic whiteboard, the team members now decide to continue in a loosely coupled mode. This means that each user can navigate independently through the document and apply changes. The changes are instantly visible for all other users. Each user is working with his own laptop and the computers are connected via a network.

Experiments (Mark, Haake, and Steritz 1997) have shown that in most cases, the users lose track of the newest additions to a collaborative hypermedia document. But even after a consolidation phase, many users still lack detailed knowledge of all parts of the document. They only had an overview knowledge. This may hinder collaboration in the group and can be the reason for potential misunderstandings. Making a document accessible is thus not sufficient if it is important that other users read this document.

Intent	Show, who received information produced by the local user.
Context	The users are providing information in an information space.
————— ◆◆◆ —————	
Problem	<b>Users are providing information for other users by means of shared objects. But making an object accessible does not ensure that the object was seen by other users.</b>
Symptoms	<i>The problem becomes obvious when ...</i> <ul style="list-style-type: none"> <li>– users create information artifacts in parallel.</li> <li>– users can percept the information independently.</li> </ul>

– it is important that specific users have read the information.

**Solution**            **Therefore: Inform the author of a shared object when another user reads this object.**



**Collaborations**    A producer of information submits the information to a collaborative information space or a communication channel. Together with the information, he provides a directive for what should be done when the information reaches the consumer. The consumer perceps the information and executes the provided directive. This can for instance result in an additional log entry or a confirmation message. The producer of the information collects all confirmations and relates them to the produced information to find out, who received the information.

**Rationale**            In traditional unidirectional distribution of information, the sender has to trust in the reliability of the communication channel and in the receiver’s commitment to perceive the provided information. But reality shows that both technical and social factors are often unreliable.

By providing a back link, which is used to confirm the reception of the information, technical unreliability is detected. In addition, a social pressure is created that forces the receiver to process the information since the sender is informed that the information has reached the receiver.

**Danger Spots**        This pattern will only work, if the users trust the system that provides the information and log in personally. In web based systems that don’t require personal login, it is not possible to detect, who is visiting the site (even cookies do not reveal information about the users’ identities). This is problematic for this pattern, but it ensures that the users can control their privacy. If information like addresses of the visiting users were sent by the browser on each visit to a web site, then spammers could easily misuse the mechanism by creating a web site that collects all the addresses of the visitors.

**Known Uses**            **Electronic Mail:** The e-Mail protocol provides an option so that receivers of the mail are asked to confirm the message. It is defined in RFC 2298 (Fajman 1998). A message with a confirmation request looks as follows:

```
Message-ID: 4004159B.9030007@schuemmer.de
Disposition-Notification-To: noah@schuemmer.de
Date: Mon, 12 Jan 2004 16:58:19 +0100
From: noah@schuemmer.de
MIME-Version: 1.0
To:till.schuemmer@fernuni-hagen.de
Subject: Please read this
```

Dear Till,  
please read this mail and confirm it....  
Thanks,  
Noah

When the e-Mail client of the recipient opens a mail that includes a field labelled **Disposition-Notification-To** it automatically (or after the user's confirmation) creates an answer message that includes the following content:

```
Content-Type: message/disposition-notification; name="MDNPart2.txt"  
Content-Disposition: inline  
Content-Transfer-Encoding: 7bit
```

```
Reporting-UA: Mozilla/5.0 Gecko/20030619 Netscape/7.1 (ax)  
Final-Recipient: rfc822; till.schuemmer@fernuni-hagen.de  
Original-Message-ID: <4004159B.9030007@schuemmer.de>  
Disposition: manual-action/MDN-sent-manually; displayed
```

The message also contains a confirmation in a human readable form. The client of the sender can automatically process the message notification and mark the sent message as read.

In cases, where the recipient has to confirm a message notification, the sender cannot be sure that a user has not received the mail since the recipient could deny sending a confirmation. Important messages could thus be sent twice because the sender thought that the message did not reach the recipient.

Another way for detecting whether or not a message was shown at a client's site has recently become popular: the use of personalized links for images that are included in image tags in an HTML formatted message. In this case, the sender assumes that the client will display mail messages as HTML. When the client displays the message, it formats the content according to the HTML syntax. Especially, it loads the images that are included in `<img>` tags. This tag can point to any address for the image. The content of the above mail could look like this:

```
<html>  
<head>  
  <title></title>  
</head>  
<body text="#000000" bgcolor="#ffffff">  
Dear Till,<br>  
please read this mail and confirm it....<br>  
Thanks,<br>  
Noah  
<br>  
OK.<br>  
</body>  
</html>
```

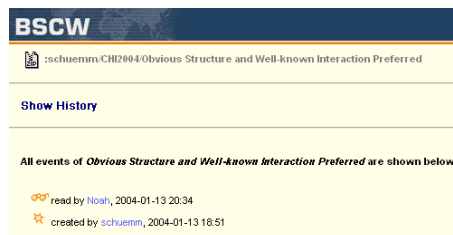
The mail client will then requests the URL

`http://www.schuemmer.de/confirmMail/4004159B.9030007`

to retrieve the image. The web server at `www.schuemmer.de` interprets this URL as a confirmation for the message with the id `4004159B.9030007` and returns an empty picture. The recipient will not notice that the message was confirmed.

While the above method works from the sender's point of view, it can harm the recipient's need for privacy. This is especially problematic if the confirmation is used to check the correctness of e-Mail addresses for junk mails as it is often done by spammers.

**BSCW:** The BSCW shared workspace system (Bentley, Horstmann, and Trevor 1997) uses an event log (cf. ELEPHANT'S BRAIN<sub>→30</sub>) to remember all users' accesses to the shared data. The event log can be queried (cf. fig. 10) and for each document stored in the shared workspace, the users can define notification patterns. By these means, it is possible for an author of a document to find out who read the document (and when).

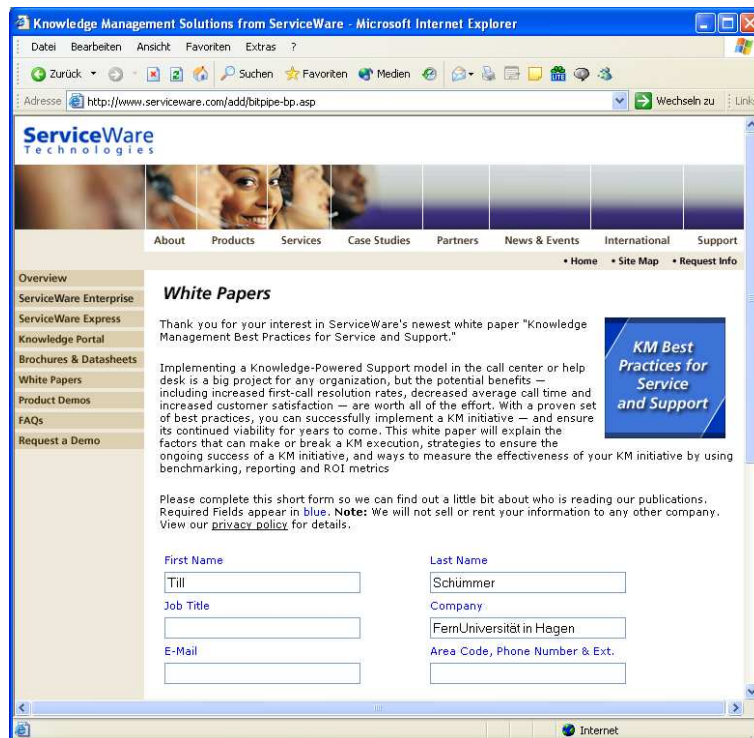


**Figure 10:** BSCW activity report tailored for a single document.

**Whitepaper download at serviceware.com:** Many companies offer free technical white-papers relating to their technical expertise and product spectrum. These white-papers are often offered for free, but knowing who reads the papers is a large capital because people interested in the companies expertise are potential customers. For that reason, more and more companies require that the user provides some personal information that reveals his identity before he can download the desired publication.

Although the request provides serviceware with the desired customer information, it may also have a negative effect: Users may not be willing to reveal their personal information or provide faked answers.

Since the information can no longer be accessed without registration, the inhibition threshold is raised. In addition, search engines will no longer be able to find the information, which may reduce the number of hits.



**Figure 11:** Requesting personal information before granting access rights to public documents.

**Messenger:** In MSN Messenger, users can inspect who has put them on their contact lists. This does not necessarily mean that they are currently watching the local user but that they potentially watch the local user. Nevertheless, it can be considered as a known use since the other users have perceived and utilized the local user's contact information.

**Related Patterns ACTIVITY COUNTER<sub>-30</sub>:** A visitor counter provides the number of users, who have perceived a specific information. Compared to WHO'S LISTENING, it does not reveal the receivers' identities. But it still provides the sender with an awareness, whether or not the provided information has been noted.

**ELEPHANT'S BRAIN<sub>-30</sub>** can be used to log information on other users' activities. By querying the Elephant's Brain, one can find out, who viewed the artifact (and who authored it).

**MAGIC DOCUMENT** (Völter 2003) is an ironic pattern that highlights the need for Who's listening even more. In the Magic Document "pattern" it is claimed that everything you have to do if you want to be sure that all team members have read important information is to put in a document and send this document to all team members. Obviously, this is an illusion – and it's the reason, why this pattern is thought as an ironic pattern.

## 3 Additional Thumbnails

### ACTIVE NEIGHBORS

The LOCAL AWARENESS<sub>-30</sub> pattern only signals confocal users on the same artifact. If users work on related artifacts, they are not aware of each other, which implies that no collaboration will be established.

**Therefore:** Provide awareness on peripheral activities that take place on related artifacts.

### ACTIVITY COUNTER

Especially for a newcomer, there is no easy way to distinguish important from less important objects.

**Therefore:** Add an activity counter to the visualization of the shared artifact.

### DON'T DISTURB \*\*\*

To allow spontaneous collaboration, users have to be open for contact requests. But each contact request disturbs the user in his current task.

**Therefore:** Include a virtual don't disturb sign in the application that signals that the user should not be disturbed.

### ELEPHANT'S BRAIN

Merging two user's (past or current) work is a difficult task. It requires that the activities are transferred to the same context and that the goals are aligned. But many applications don't provide access to the artifact's history, its use, and its evolution. Thus, merging is vulnerable to errors and often collaboration does not take place since the merging efforts exceeds the estimated gains of a collaboration.

**Therefore:** Remember all activities that users perform on shared artifacts – not only modifying accesses, but also read accesses.

### FIND THE GURU

You know that other users have more expertise with the artifact, but you don't know how to find them.

**Therefore:** Find the person, who shares a long history with the artifact.

### GAZE OVER THE SHOULDER \*\*

Many proprietary tools are not designed for extendability. They do not provide means to modify the application's behavior. This makes it difficult to automatically track user's activities, which you would need to provide awareness.

**Therefore:** Add an additional layer in the communication between the application and the shared data to monitor user actions.

### LOCAL AWARENESS

Although most systems that work on shared data provide support for coordinating shared access, they often don't tell the user, who is working on a specific arti-

fact. Such information is needed to establish ad-hoc teams that share a common focus. Without such information, users assume to work alone – and do not see the possibility or urge for collaboration.

**Therefore:** Provide awareness in context. This means that the system tells the local user, who else is currently interested in the local user’s focussed artifact and what they do with this artifact.

## PAY BACK

To keep the users providing personal information, they have to be motivated. But not in every case, a motivation is possible within the desired transaction. Thus, it is difficult to maintain RECIPROCITY<sub>→21</sub>.

**Therefore:** Provide each user with a virtual currency that can be used to purchase services of other users or institutions. Let users earn a specific amount of that currency, when they help others.

## SEMANTIC DISTANCE

Your SEMANTIC NET<sub>→31</sub> is very dense in a sense that artifacts have a semantic relation to many other artifacts. But not all artifacts have the same importance for the user. If the user sees only the semantic net, he might get lost in the diversity of relations.

**Therefore:** Use weighted edges to describe the strength of the semantic relation. Interpret these edges as distances.

## SEMANTIC NET

Detecting short semantic distances between artifacts based on a similarity measure often leads to ineffective and inexact results.

**Therefore:** Produce a semantic net that contains artifacts and relations between artifacts. Relate two artifacts, if they have much in common (as in the SEMANTIC DISTANCE<sub>→31</sub> pattern). Define the distance between two artifacts as the length of the shortest path between these artifacts.

## SHOW THE EXPERT \*

Individual user’s knowledge may be useful for other users. On the other hand, exposing knowledge can mean that the knowledgeable user is confronted with questions of the other users.

**Therefore:** Show experts for each shared object in the collaborative application.

## TRAIN THE RECOMMENDER

Recommendations may be wrong because the recommender worked with incomplete or incorrect user profiles.

**Therefore:** Whenever a recommendation is made let the user provide feedback whether or not the recommendation made sense for the user. If not adapt the data that led to the recommendation.

## USER GALLERY

Without knowing who is using the system, it is hard to establish collaboration or to become aware of other users' activities.

**Therefore:** Provide a list with all users, who are members of the community.

## USER LIST

You don't know, who is using a collaborative application.

**Therefore:** Show the names of all users who are in the same session in a user list.

## References

- Autonomy (2002). Collaboration & expertise networks – cen. Aut cen 11.02, Autonomy Inc., <http://www.autonomy.com/>.
- Bentley, R., T. Horstmann, and J. Trevor (1997). The world wide web as enabling technology for cscw: The case of bscw.
- Boyle, M., C. Edwards, and S. Greenberg (2000). The effects of filtered video on awareness and privacy. In *Proceedings of the 2000 ACM conference on Computer supported cooperative work*, Philadelphia, Pennsylvania, United States, pp. 1–10.
- Cosley, D., S. K. Lam, I. Albert, J. A. Konstan, and J. Riedl (2003). Is seeing believing?: how recommender system interfaces affect users’ opinions. In *Proceedings of the conference on Human factors in computing systems*, pp. 585–592. ACM Press.
- DeMarco, T. and T. Lister (1999). *Peopleware (2nd ed.): productive projects and teams*. Dorset House Publishing Co., Inc.
- Fajman, R. (1998, March). RFC 2298: An extensible message format for message disposition notifications. Status: PROPOSED STANDARD.
- Foner, L. N. (1996, April). A multi-agent referral system for matchmaking. In *The First International Conference on the Practical Applications of Intelligent Agents and Multi-Agent Technology*, London, UK.
- Fuchs, L., S. Poltrock, and I. Wetzel (2001). Teamspace: An environment for team articulation work and virtual meetings. In *Proceedings of WBC '01, Web Based Collaboration (DEXA 2001)*, Munich, Germany.
- Goffman, E. (1963). *Behavior in Public Places – Notes on the Social Organization of Gatherings*. New York: Free Press.
- Graham, P. (2002). A plan for spam. <http://www.paulgraham.com/spam.html>.
- Grudin, J. (1988). Why cscw applications fail: Problems in the design and evaluation of organizational interfaces. In *Proceedings of the ACM Conference on Computer-Supported Cooperative Work (CSCW88)*, Portland, Oregon, pp. 85–93.
- Jackson, T. W., R. Dawson, and D. Wilson (2003). Understanding email interaction increases organizational productivity. *Commun. ACM* 46(8), 80–84.
- Lee, A., A. Girgensohn, and K. Schlueter (1997). Nynex portholes: initial user reactions and redesign implications. In *Proceedings of the international ACM SIGGROUP conference on Supporting group work : the integration challenge*, pp. 385–394. ACM Press.
- Mark, G., J. M. Haake, and N. A. Steritz (1997). Hypermedia use in group work: Changing the product, process, and strategy. *Computer Supported Cooperative Work (CSCW)* 6(4), 327–368.
- Pikrakis, A., T. Bitsikas, S. Sfakianakis, D. De Roure, W. Hall, S. Reich, and G. Hill (1998). Memoir: Software agents for finding similar users by trails. In H. S. Nwana and D. T. Ndumu (Eds.), *Proceedings of the 3rd International*

- Conference on the Practical Applications of Agents and Multi-Agent Systems (PAAM-98)*, London, UK, pp. 453–466.
- Pipek, V., J. Hinrichs, and V. Wulf (2003). Sharing expertise: Challenges for technical support. In M. S. Ackermann, V. Pipek, and V. Wulf (Eds.), *Sharing Expertise: Beyond Knowledge Management*, pp. 111–136. Cambridge, MA, USA: MIT Press.
- Schümmer, T. (2004). Gama – a pattern language for computer supported dynamic collaboration. In K. Henney and D. Schütz (Eds.), *EuroPLoP 2003, Proceedings of the 8th European Conference on Pattern Languages of Programs, 2003*, Konstanz, Germany. UVK.
- Schümmer, T. and J. M. Haake (2001). Supporting distributed software development by modes of collaboration. In *Proceedings of ECSCW 2001*, Bonn.
- Shardanand, U. and P. Maes (1995). Social information filtering: algorithms for automating 'word of mouth'.
- Stroebe, w., M. Hewstone, and G. M. Stephenson (Eds.) (1996). *Sozialpsychologie*. Springer.
- Völter, M. (2003). Hope, belief and wizardry – three different perspectives on project management. In A. O’Callaghan, J. Eckstein, and C. Schwanninger (Eds.), *EuroPLoP 2002, Proceedings of the 7th European Conference on Pattern Languages of Programs, 2002*, Konstanz, Germany, pp. 443–462. UVK.
- Wilke, H. and A. van Knippenberg (1996). Gruppenleistung. In (*Stroebe, Hewstone, and Stephenson 1996*) (3 ed.), pp. 455–502. Springer.
- Wisneski, C., H. Ishii, A. Dahley, M. Gorbet, S. Brave, B. Ullmer, and P. Yarin (1998). Ambient displays: Turning architectural space into an interface between people and digital information. In N. A. Streitz, S. Konomi, and H. J. Burkhardt (Eds.), *Cooperative Buildings, Integrating Information, Organization, and Architecture, First International Workshop, CoBuild’98*, Volume 1370 of *LNCS*, Darmstadt, Germany, pp. 22–32. Springer.
- Yiman, D. (2000). Expert finding systems for organizations: Domain analysis and the demoir approach. In *Beyond Knowledge Management: Sharing Expertise*, Boston, MA, pp. online at: <http://citeseer.nj.nec.com/yimam00expert.html>. MIT Press.
- Zhao, Q. A. and J. T. Stasko (1998, November 14 - 18, 1998). Evaluating image filtering based techniques in media space applications. In *Proceedings of the 1998 ACM conference on Computer supported cooperative work*, 1998, Seattle, Washington, United States, pp. 11–18.

# Contents

<b>1</b>	<b>Introduction</b>	<b>1</b>
1.1	The Pattern Language . . . . .	2
1.2	Structure of the Individual Patterns . . . . .	2
<b>2</b>	<b>Patterns for Filtering Personal Information in Collaborative Systems</b>	<b>4</b>
2.1	MASQUERADE ** . . . . .	4
2.2	ATTENTION SCREEN ** . . . . .	8
2.3	BUDDY LIST **** . . . . .	13
2.4	BIRDS OF A FEATHER * . . . . .	17
2.5	RECIPROCITY *** . . . . .	21
2.6	WHO'S LISTENING * . . . . .	25
<b>3</b>	<b>Additional Thumbnails</b>	<b>30</b>