COBRA: A Constraint-Based Awareness Management Framework

Tom Gross
Faculty of Media
Bauhaus-University Weimar
Bauhausstr. 11, 99423 Weimar, Germany
tom.gross(at)medien.uni-weimar.de

ABSTRACT
The effective and efficient coordination of remote users for cooperative work or any other social interaction requires that the users have adequate information about each other and the environment. This paper outlines some basic challenges of managing awareness information. It analyses the management of awareness information in face-to-face situations, and discuss challenges and requirements for the support of awareness management in distributed settings. A simple, yet powerful framework for awareness management based on constraints is introduced.

Author Keywords
Computer-Supported Cooperative Work; Group Awareness; Presence Information

ACM Classification Keywords
H5.2 [Information Interfaces and Presentation]: User Interfaces – Graphical User Interfaces, User-Centred Design; H5.3 [Information Interfaces and Presentation]: Group and Organisation Interfaces – Computer-Supported Cooperative Work

INTRODUCTION

The effective and efficient coordination of remote users for cooperative work or any other social interaction requires that the users have adequate information about each other and the environment. This information—often referred to as (group) awareness—can include data about the presence and availability of others, as well as data about the state of shared artefacts.

In the fields of Computer-Supported Cooperative Work, Human-Computer Interaction, and Ubiquitous Computing a great number of concepts, systems and prototypes to support awareness have been presented over the past 15 years. Examples are event and notification services such as Elvin [14], Khronika [8], and NSTP [9]; context-aware computing applications [11] and frameworks [2]; awareness modelling approaches and systems such as AETHER [10], and MoMA [15]; as well as context-based collaborative infrastructures such as ENI [6].

Yet, as two recent (one in 2002, and one forthcoming) special issues on awareness in the Kluwer/Springer Journal of Collaborative Computing show, several challenges still remain [12, 13]. In particular, the dual trade-off between awareness and disturbance and awareness and privacy remains a challenge for system designers: on the one hand users need information to create and maintain awareness providing a basis or context for our own activities [3], on the other hand users have a legitimate need for little disruption while performing their tasks and for privacy regarding the information being sent to others about oneself [7].

This paper suggests awareness management mechanisms based on constraints to balance these trade-offs. The term awareness management has been used for collaborative virtual environments, but was limited to a mechanism to reduce information overload [e.g., 1]. Here awareness management is defined as the process of controlling incoming awareness information of others (thus the control of disruption) and outgoing awareness information about oneself (thus the control of one's privacy) in digital environments. The paper first discusses some basic challenges and requirements for managing awareness information. It then briefly introduces COBRA—a COnstraint-Based awaReness mAnagement framework.

MANAGING AWARENESS INFORMATION

In face-to-face situations humans have natural behaviour and conventions for both creating and maintaining their own awareness, as well as for stimulating other's awareness about oneself. For creating and maintaining their own awareness, the continuous perception of information is necessary. In the real world we are used to shift our attention by focussing on certain things while neglecting others. We do not perceive everything at the same time the same way—that is, we control our information acquisition and maintenance processes, mostly implicitly. For stimulating others' awareness about oneself humans reveal some information about themselves and hide other information. This corresponds to a social presentation of one's identity [5]. Typically, humans adapt the facet to the respective situation (e.g., time, location, occasion) and to the interpersonal context (e.g., present people, roles).

Through the mediation of remote users' interaction by (computer) technology, some new challenges for acquiring and exposing awareness information arise. These challenges lead to the following major design requirements, which were also raised in the references above:

- Incoming and outgoing information should be considered: a selective dissemination of information and a context-dependent presentation is needed.
- Information selection should be done carefully. Supporting situational and interpersonal context should be a key goal.
- Means supporting self-awareness such as views from another user's perspective should be provided.
- Controlling incoming and outgoing information should be efficient for users, thus staying a secondary task.
- Synchronous and asynchronous modes should be supported.

These design challenges were central issues in developing our simple constraint-based awareness management framework introduced below.

A CONSTRAINT-BASED AWARENESS MANAGEMENT FRAMEWORK

Based on the design requirements outlined above, we developed COBRA—a COnstraint-Based awaReness mAnagement framework—that is comprised of a four-staged flow of information applying a constraint pattern as information filter (cf. Figure 1).

Flow of Information

Basically, the information is *gathered* from one user or particular situation, *distributed* and eventually *presented* to another user or another situation. Furthermore, in an additional stage—the *process* stage—information can be enriched by external data sources (e.g., LDAP or other databases). Additionally, information may also be manipulated to comply with certain formats or conventions, which means that some of its parts may be rearranged or even be removed again.

The framework offers asynchronous and synchronous modes of dealing with awareness information. Both can be distinguished by the information's source, the way it is disseminated (server push or client pull), and by the number of recipients. The synchronous gathering stage considers information that can be either actively generated or passively collected from its originator.

Filtering Through Constraints

At each of the four stages constraint-based filters can be applied. Constraints represent circumstances and conditions and are restrictions on our freedom to pursue certain activities; they can be used to prevent disruption and protect privacy. There are several kinds of constraints, which can be grouped into constraint categories: physical constraints (e.g., proximity, size); social constraints (e.g., norms, relations and roles); individual constraints (e.g., cognitive, logical, physical, physiological capabilities, expertise); legal constraints (e.g., rules, regulations, laws); organisational constraints (e.g., organisational rules and regulations); and technical constraints (e.g., limitations due to the equipment used)

While there are constraints that can be violated (e.g., social constraints), there are also others that cannot (e.g., physical constraints). Some are harder to implement than others; some even ought to be left to mechanisms outside the digital realm like social protocols.

Constraints can be combined to constraint patterns in order to meet the above requirements. For instance, a *social constraint pattern* was designed to support users' facets. Facets expose certain information to a group of people. They help to set up interpersonal contexts. In our framework a user may define multiple facets, each with a name, which may be opened or closed determining the sending and reception mode of information associated with a particular facet. Thus, a user could have a particular facet with working colleagues (e.g., allow them to see the working email address), and a different facet with private friends (e.g., allowing them to see the private email address

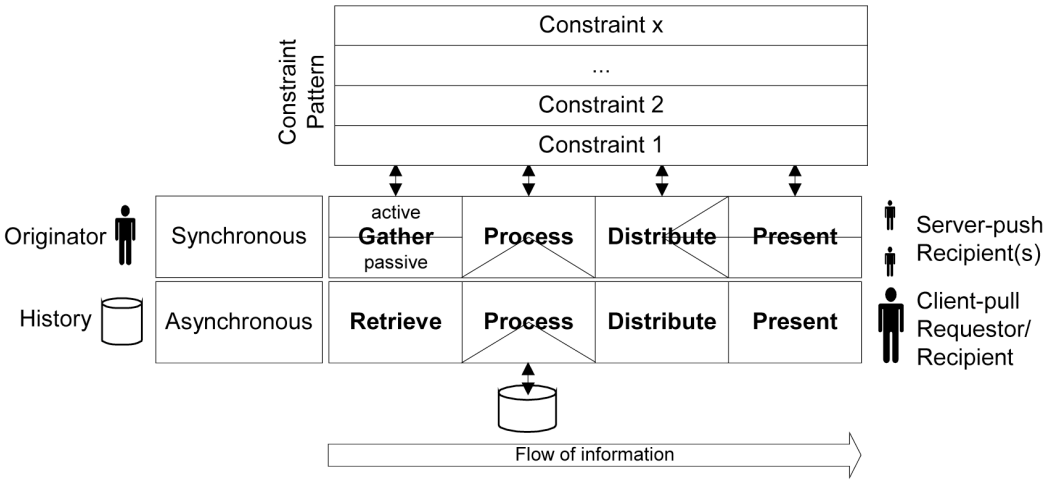

Figure 1. Constraint-based awareness management framework.

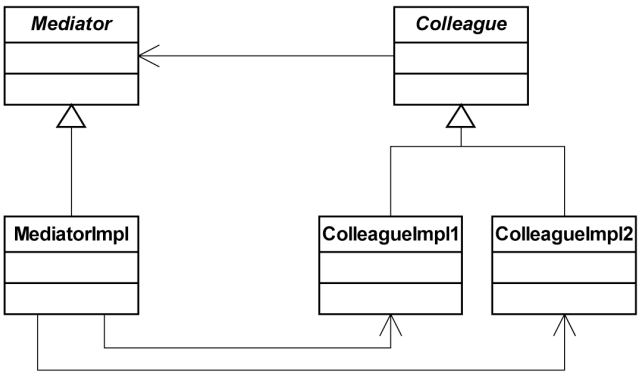

Figure 2. Mediator design pattern in UML.

as well as the private mobile phone number). One or more open facets allow for concurrent contexts. An active facet promotes the user's current context. Every user has a public facet, which corresponds to a general face exposed especially to new or unknown contacts.

Two users are related to one another via one or more bridges. A bridge describes a user-facet combination in relation to another user-facet combination. For instance, user A assigned user B to his facet 1, while user B assigned user A to his facet 2 (i.e., A-1:B-2). This reveals that facets are not necessarily symmetrical but personal—as in the real world. User A might be in user B's "work" facet, while the user B is placed in A's "public" facet. There may be multiple bridges describing a relationship between two users. Additionally, a bridge allows determining when another user is within or outside one's context (the one originally negotiated and stored as a bridge).

Implementation

The above concept of COBRA is realised as a *mediator pattern*, which is widely used in object-oriented modelling of software systems [4]. It is comprised of a central controlling instance (called mediator in the pattern language), which promotes loose coupling by keeping the collaborating objects (called colleagues in the pattern language) from referring to each other (cf. Figure 2).

The mediator controls and coordinates all ongoing interaction and represents the application's overall behaviour. All communication between colleagues goes through the mediator—colleagues cannot refer to each other directly. The mediator receives and forwards information to the respective colleague(s). In the COBRA framework the communication works according to negotiated bridges. The mediator also realises the connection to other external services needed at the process stage like databases or directory services. Additionally, the mediator has knowledge of the whole situation and all bridges to its colleagues and external data sources. Therefore, the mediator can also be used as a basis for the context-sensitive social network analysis. For instance, a graphical representation of the social network among the users could be generated automatically.

A colleague is an entity, which can gather information from its environment, or present information to its environment. There are three types of colleagues:

- Colleagues just *gathering* information. These are mostly so called agents or bots, which constantly provide information about a certain topics (e.g., the stock market).
- Colleagues just *presenting* information. These can be used to integrate awareness features into other applications. For instance, a colleague of this type could present a user's status on a Web site or in an expertise location system. Only one facet needs to be bridged to this colleague, thus the user can control his availability in those systems with the same mechanisms described above as for any other system or user.
- Colleagues *gathering* and *presenting* information. These are regular clients as described in previous sections.

All types may be combined in facets with the user being able to control the information they send us and the information they receive from us.

The COBRA framework and its mediator pattern are implemented as a *client-server infrastructure*. A central mediator, which is always running, guarantees persistency. This is particularly the case for asynchronous situations, where some colleagues simply gather information and it is not yet clear how this information will be used in future. The software architecture consists of a centralised mediator facilitating the communication among the colleagues; we have three types of colleagues (according to the description above): a colleague solely consisting of one or more sensors, a colleague solely consisting of one or more widgets, and a colleagues consisting of both (i.e., which is able to both gather and present information). The communication in this implementation is done via XML-RPC—that is, the single components communicate via remote procedure calls (RPCs), and the representation of the data is done in the extended markup language (XML).

All data represent events that happen in this architecture. We have different types. Examples are person-related events such as conversation messages, or presence information (e.g., status changes such as opening or closing a facet); and environment-related events such as changes in the environment (e.g., changes of prices at the stock exchange).

CONCLUSIONS

In this paper I have described some issues of managing awareness, and introduced the COBRA framework with its flow of information, filtering through constraints, and implementation.

In this position paper I could only address some selected items. In the workshop I would be particularly interested in discussing results and ideas for conceptual issues of *managing* awareness (e.g., modelling information, mining of data, filtering incoming and outgoing information), for conceptual issues of *presenting* awareness (e.g., on the computer desktop, but also in the physical world through ambient interfaces), and for technical issues of *capturing* information, transferring, processing, and storing information.

BIOGRAPHICAL INFORMATION

Tom Gross is associate professor for Computer-Supported Cooperative Work and head of the Cooperative Media Lab at the Faculty of Media of the Bauhaus-University Weimar, Germany. His research interests include Computer-Supported Cooperative Work, Human-Computer Interaction, and Ubiquitous Computing. He recently co-organised a special issue on 'Context-Aware Computing in CSCW' for the Kluwer/Springer Journal on Collaborative Computing (forthcoming). From 1999 to 2003 he was a senior researcher at the Fraunhofer Institute for Applied Information Technology FIT in St. Augustin, Germany. He holds a diploma and a doctorate degree in Applied Computer Science from the Johannes Kepler University Linz, Austria.

ACKNOWLEDGEMENTS

I would like to thank Christoph Oemig and Tareg Egla for many inspiring discussions, and for co-organising student projects at the Cooperative Media Lab.

REFERENCES

1. Antunes, M., Silva, A.R. and Martins, J. An Abstraction for Awareness Management in Collaborative Virtual Environments. In Proceedings of the ACM Symposium on Virtual Reality Software and Technology - VRST 2001 (Nov. 15-17, Banff, Canada). ACM, N.Y., 2001. pp. 33-39.
2. Dey, A.K. Providing Architectural Support for Building Context-Aware Applications. Ph.D. thesis, Georgia Institute of Technology, Nov. 2000.
3. Dourish, P. and Bellotti, V. Awareness and Coordination in Shared Workspaces. In Proceedings of the Conference on Computer-Supported Cooperative Work - CSCW'92 (Oct. 31-Nov. 4, Toronto, Canada). ACM, N.Y., 1992. pp. 107-114.
4. Gamma, E., Helm, R., Johnson, R. and Vlissides, J. Design Patterns: Elements of Reusable Object-Oriented Software. Addison-Wesley, Reading, MA, 1994.
5. Goffman, E. The Presentation of Self in Everyday Life. Anchor, N.Y., 1956.
6. Gross, T. and Prinz, W. Modelling Shared Contexts in Cooperative Environments: Concept, Implementation, and Evaluation. Computer Supported Cooperative Work: The Journal of Collaborative Computing (to appear).
7. Hudson, S.E. and Smith, I. Techniques for Addressing Fundamental Privacy and Disruption Tradeoffs in Awareness Support Systems. In Proceedings of the ACM 1996 Conference on Computer-Supported Cooperative Work - CSCW'96 (Nov. 16-20, Boston, MA). ACM, N.Y., 1996. pp. 248-257.
8. Loevstrand, L. Being Selectively Aware with the Khronika System. In Proceedings of the Second European Conference on Computer-Supported Cooperative Work - ECSCW'91 (Sept. 24-27, Amsterdam, NL). Kluwer Academic Publishers, Dortrecht, NL, 1991. pp. 265-278.
9. Patterson, J.F., Day, M. and Kucan, J. Notification Servers for Synchronous Groupware. In Proceedings of the ACM 1996 Conference on Computer-Supported Cooperative Work - CSCW'96 (Nov. 16-20, Boston, MA). ACM, N.Y., 1996. pp. 122-129.
10. Sandor, O., Bogdan, C. and Bowers, J. Aether: An Awareness Engine for CSCW. In Proceedings of the Fifth European Conference on Computer-Supported Cooperative Work - ECSCW'97 (Sept. 7-11, Lancaster, UK). Kluwer Academic Publishers, Dortrecht, NL, 1997. pp. 221-236.
11. Schilit, B.N., Adams, N.I. and Want, R. Context-Aware Computing Applications. In Proceedings of the Workshop on Mobile Computing Systems and Applications (IEEE Computer Society, Santa Cruz, CA, 1994. pp. 85-90.
12. Schmidt, A., Gross, T. and Billinghurst, M. Introduction to special issue on Context-Aware Computing in CSCW. Computer Supported Cooperative Work: The Journal of Collaborative Computing (to appear).
13. Schmidt, K. The Problem With Awareness: Introductory Remarks on Awareness in CSCW. Computer Supported Cooperative Work: The Journal of Collaborative Computing 11, 3-4 (2002). pp. iii-iv.
14. Segall, B. and Arnold, D. Elvin has Left the Building: A Publish/Subscribe Notification Service with Quenching. In Proceedings of Australian UNIX and Open Systems Users Group Conference - AUUG'97 (Sept. 3-5, Brisbane, Australia). 1997.
15. Simone, C. and Bandini, S. Integrating Awareness in Cooperative Applications through the Reaction-Diffusion Metapher. Computer Supported Cooperative Work: The Journal of Collaborative Computing 11, 3-4 (2002). pp. 495-530.