ISSN 1861-4280

international reports on socio-informatics

volume 2 issue 2
2005

Guest Editors:

Yvonne Dittrich
Paul Dourish
Anders Mørch
Volkmar Pipek
Gunnar Stevens
Bettina Törpel

Editors:

Volkmar Pipek
Markus Rohde

Publisher:
IISI - International Institute for Socio-Informatics

Table of contents

The ‘international reports on socio-informatics’ are an online report series of the International Institute for Socio-Informatics, Bonn, Germany. They aim to contribute to current research discourses in the fields of ‘Human-Computer-Interaction’ and ‘Computers and Society’. The ‘international reports on socio-informatics’ appear at least two times per year and are exclusively published on the website of the IISI.

Impressum

IISI - International Institute for Socio-Informatics
Heerstraße 148
53111 Bonn
Germany

fon: +49 228 6910-43
fax: +49 228 6910-53
mail: contact@iisi.de
web: http://www.iisi.de

Supporting Appropriation Work: A Workshop Report

Volkmar Pipek
Int. Institute for Socio-Informatics, Bonn, Germany,
and University of Oulu, Finland
volkmar.pipek@iisi.de

Abstract. A significant amount of research in the field of Computer-Supported Cooperative Work has been done to understand the mutual shaping of collaboration technologies and work practices. The outcome of such research helped improving the design of information systems as well as improving the related design processes with regard to a better technology-practice fit. Nevertheless, the approaches to support the related activities focused on design work and designer activities. With a workshop at the European Conference on Computer-Supported Cooperative Work in 2005, we opened a forum for research that focuses on supporting 'appropriation work', the activities that users engage in to shape and make sense of the technologies that are provided to them from different design arenas. The workshop established a broader research focus on technological and non-technological approaches to support users in reflecting and re-designing the use (and non-use) of collaborative technologies.

1 Introduction

The CSCW community was always aware of the fact that not only designer's skills contribute to successfully putting collaborative technologies into practice. It also requires user activities, and these should be addressed:

> <…> a CSCW system should provide facilities supporting users in appropriating, exploring, modifying, negotiating etc. - cooperatively and yet distributed - 'community handbooks' that are openly incomplete and inconsistent. Providing support for distributed cooperative

> appropriation, circumvention, modification of the system is, perhaps, the toughest challenge in designing computer systems for cooperative work. (Schmidt 1991)

However, research regarding 'appropriation' of CSCW systems developed in two main directions. Studies on the evolving use of CSCW systems (Orlikowski 1992, Robertson 1998, Karsten and Jones 1998, Pipek and Wulf 1999, Dittrich et al. 2002, Hansson et al. 2003, Törpel et al. 2003, Karasti and Syrjänen 2004) described 'appropriation' as a phenomenon or process to improve the understanding of the critical success factors of the use of collaborative tools. On the one hand the studies clarified the importance of the user's contributions to the successful establishment of collaborative technology, and stressed the notion that appropriation often goes beyond the intentions and expectations that have been associated with the original design of a collaborative technology. On the other hand most of the research maintained an analytical perspective on the phenomenon of appropriation, and did not give advice on how to stimulate or support appropriation activities.

Several prerequisites for supporting user activities have been developed. Regarding approaches with a technological focus, concepts of (re-)designing technologies during use (to make CSCW systems 'tailorable') have been developed. The research trajectory here started with the introduction of 'tailorability' as a requirement for collaborative technologies (Trigg et al. 1987, Henderson and Kyng 1991). The "architectural" perspective then explored tailorability to develop concepts and examples for very flexible software systems, which could be tailored to their use scenarios (Maclean et al. 1990, Malone et al. 1992, Morch 1997, Stiemerling and Cremers 2000). Object-Orientation (Morch 1997) and Component-Based Systems (Stiemerling and Cremers 2000) have been explored to increase the flexibility of software artefacts, other approaches addressed issues of analyzing, separating and composing tailoring entities along the typical functionality of CSCW systems (Malone et al. 1992, Teege 2000).

The "user-interface" perspective explored how tailorable software should present itself to the tailors. Henderson and Kyng (Henderson and Kyng 1991) distinguished three levels of tailoring (choosing between predefined alternatives, constructing new artefacts from existing pieces, and reprogramming the artefact) that require different levels of expertise regarding the supporting technology. Obviously, ordinary groupware users cannot be expected to acquire programming skills to be able to tailor an artefact accordingly. Several approaches, some inspired by Nardi's (1993) work on end-user programming, aim at developing tailoring environments that provide simple concepts and interfaces for end-users (MacLean et al. 1990, Malone et al. 1992, Stiemerling et al. 1997, Teege 2000, Liebermann et al. 2005).

Approaches to support collaborative tailoring (e.g. Kahler 2001, see overview in Pipek and Kahler 2005) stressed the social dimensions of tailoring work, but still suffered from several weaknesses:

- They still maintain a ‘designer’ perspective by focusing on tailoring *one* tool, while the users at the workplace face the challenge of orchestrating the diversity of tool infrastructures, interdependencies and restrictions. (Robertson 1998, Dourish 2003, Pipek 2005b)
- They focus on supporting altering tool configurations and settings, but not tool *usages*. The latter does not necessarily involve altering a tool (see example of using a comment field of a helpline’s database in Orlikowski 1996).
- They often support only indirect user-user-interaction (e.g. by providing shared configuration repositories) instead of direct communication (e.g. for negotiation and sensemaking).

Latest research approaches tried to combine these two research traditions of ‘tailoring/end-user development’ and ‘appropriation studies’ to provide concepts to improve appropriation support (Dittrich et al. 1998, Dourish 2003, Pipek 2005a) and that allow users an active and collaborative reflection of their use of CSCW systems.

2 Workshop Course and Results

The workshop attracted more than 30 researchers from areas like Human-Computer Interaction, Psychology, Work Sciences, etc. The diversity has been present in the submissions, and it is present in the extended position papers that we present in this special issue.

21 researchers found their way to the conference site in Paris, France. After an introductory discussion the workshop participants divided up into four discussion groups that focused on ‘Appropriation and Ubiquitous Computing’, ‘Affordances for Appropriation’, ‘Organisation of Use vs. Design/Appropriation Processes’, and ‘Simple systems and social structures’. A final discussion about future research agendas concluded the workshop.

This special issue collects the position papers of the workshop as extended versions of the original submissions. This allowed the authors to integrate the results from the workshop discussions into their argument.

In addition, the group discussing the topic of ‘Simple systems’ also provided a discussion report that concludes this special issue.

3 Acknowledgements

I’d like to thank the other workshop organisers Yvonne Dittrich, Paul Dourish, Anders Mørch, Gunnar Stevens and Bettina Törpel for sharing the research initative towards supporting appropriation work with me. Matthias Betz and Veronika Voigt helped in completing this special issue.

4 References

Dittrich, Y. (1998): How to make Sense of Software - Interpretability as an Issue for Design, Dep. of Computer Science and Business Administration, University of Karlskrona/Ronneby, TR 98/19, Ronneby, Sweden, 1998, 9.

Dittrich, Y., Eriksen, S. and Hansson, C. (2002): PD in the Wild; Evolving Practices of Design in Use. in Participatory Design Conference, (Malmö, Sweden, 2002), CPSR, 124-134.

Dourish, P. (2003): The Appropriation of Interactive Technologies: Some Lessons from Placeless Documents. Computer Supported Cooperative Work (CSCW) - The Journal of Collaborative Computing, 12 (4). 465-490.

Hansson, C., Dittrich, Y. and Randall, D. (2003): "The development is driven by our users, not by ourselves" - Including Users in the Development of Off-The-Shelf Software. in 26th Information Systems Research Seminar in Scandinavia (IRIS 26), (Haikko Manor, Finland, 2003), IRIS Association.

Henderson, A. and Kyng, M. (1991): There's no place like home: Continuing Design in Use. in Greenbaum, J. and Kyng, M. eds. Design at work: Cooperative Design of Computer Systems, Lawrence Erlbaum Ass., Hillsdale, NJ, 1991, 219-240.

Kahler, H. (2001) Supporting Collaborative Tailoring, PhD Thesis, Department of Communication, Journalism and Computer Science, Roskilde University, Roskilde, 2001.

Karasti, H. and Syrjänen, A.-L. (2004): Artful Infrastructuring in two cases of community PD. in Participatory Design Conference (PDC'04), (Toronto, Canada, 2004), CPSR, 20-30.

Karsten, H. and Jones, M. (1998): The long and winding road: Collaborative IT and organisational change. in Int. Conference on Computer Supported Work (CSCW'98), (Seattle, WA, USA, 1998), ACM Press, 29-38.

Lieberman, H., Paternó, F. and Wulf, V. (eds.) (2005): End User Development. Kluwer, Dordrecht, NL, in press.

MacLean, A., Carter, K., Lövstrand, L. and Moran, T.P. (1990): User-Tailorable Systems: Pressing the Issue with Buttons. in Int. Conference on Computer-Human-Interation (CHI'90), (Seattle, WA. USA, 1990), ACM Press, 175-182.

Malone, T.W., Lai, K.-Y. and Fry, C. (1992): Experiments with Oval: A Radically Tailorable Tool for Cooperative Work. in Int. Conference on CSCW (CSCW'92), (Toronto, Canada, 1992), ACM Press, 289-297.

Morch, A. (1997): Three Levels of End-user Tailoring: Customization, Integration, and Extension. in Kyng, M. and Mathiassen, L. eds. Computers and Design in Context, MIT Press, Cambridge, MA, USA, 1997, 51-76.

Nardi, B.A. (1993): A small matter of programming. Perspectives on End-User Programming. MIT Press, Cambridge, Massachusetts, 1993.

Orlikowski, W.J. (1992): Learning from Notes: organizational issues in groupware implementation. in Proceedings of the 1992 ACM conference on Computer-supported cooperative work, (Toronto, Ontario, Canada, 1992), 362-369.

Orlikowski, W.J. (1996): Evolving with Notes: Organizational change around groupware technology. in Ciborra, C.U. ed. Groupware & Teamwork, Wiley, Chichester, 1996, 23 - 60.

Pipek, V. (2005a): From Tailoring to Appropriation Support: Negotiating Groupware Usage Faculty of Science, Department of Information Processing Science (ACTA UNIVERSITATIS OULUENSIS A 430), University of Oulu, Oulu, Finland, 2005, 246.

Pipek, V. (2005b): Negotiating Infrastructure: Supporting the Appropriation of Collaborative Software. Scandinavian Journal on Information Systems. (submitted).

Pipek, V. and Kahler, H. (2005): Supporting Collaborative Tailoring. in Lieberman, H., Paterno, F. and Wulf, V. eds. End-User Development, Kluwer, Dordrecht, NL, 2005, to be published.

Pipek, V. and Wulf, V. (1999): A Groupware's Life. in European Conference on Computer Supported Cooperative Work (ECSCW'99), (Copenhagen, Denmark, 1999), Kluwer, Dordrecht, Netherlands, 199-218.

Robertson, T. (1998): Shoppers and Tailors: Participative Practices in Small Australian Design Companies. Computer Supported Cooperative Work (CSCW), 7 (3-4). 205 221.

Schmidt, K. (1991): Riding a tiger, or Computer Supported Cooperative Work. in Proceedings of the Second European Conference on Computer-Supported Cooperative Work, (Amsterdam, NL, 1991), Kluwer Academic Publishers, 1-16.

Stiemerling, O. and Cremers, A.B. (2000): The EVOLVE Project: Component-Based Tailorability for CSCW Applications. AI & Society, 14. 120-141.

Stiemerling, O., Kahler, H. and Wulf, V. (1997): How to Make Software Softer - Designing Tailorable Applications. in DIS '97, (Amsterdam, 1997), ACM Press, 365-376.

Teege, G. (2000): Users as Composers: Parts and Features as a Basis for Tailorability in CSCW Systems. International Journal of Computer-Supported Cooperative Work, 9 (1). 101-122.

Törpel, B., Pipek, V. and Rittenbruch, M. (2003): Creating Heterogeneity - Evolving Use of Groupware in a Network of Freelancers. Special Issue of the Int. Journal on CSCW on "Evolving Use of Groupware", 12 (4). 381-409

Trigg, R.H., Moran, T.P. and Halasz, F.G. (1987), Adaptability and Tailorability in NoteCards. in INTERACT'87, (Stuttgart, Germany, 1987), 723-728.

Three Contexts of Appropriation for an Urban Simulation System

Alan Borning,
Janet Davis
Department of Computer Science & Engineering University of Washington Seattle, Washington 98195, USA
{borning, jlnd}@cs.washington.edu

Abstract. UrbanSim is an integrated land use and transportation simulation system. Its purpose is to help inform public deliberation and decision-making regarding major land use and transportation decisions, by simulating the consequences of different alternatives for an urban region over periods of twenty to thirty years. Indicators provide the primary tool for portraying key results from the simulations to users. We describe three contexts of appropriation for UrbanSim. The first is appropriation by different metropolitan regions, to simulate urban development in those regions. The second is appropriation by advocacy groups, business associations, and other organizations, which can use an Indicator Perspectives mechanism to present their particular viewpoint on what is important to measure in the simulation results and how it should be interpreted. The third is appropriation by individuals, using Personal Indicators to help understand how decisions would affect them personally.

1 Introduction

In many regions in the United States and globally, there is increasing concern about pollution, traffic jams, resource consumption, loss of open space, loss of coherent community, lack of sustainability, and unchecked sprawl. Elected officials, planners, and citizens in urban areas grapple with these difficult issues as they develop and evaluate alternatives for such decisions as building a new rail

line or freeway, establishing an urban growth boundary, or changing incentives or taxes. These decisions interact in complex ways, and, in particular, transportation and land use decisions interact strongly with each other. To help the understanding of the long-term consequences of these decisions, Waddell, Borning, and their colleagues have been developing UrbanSim, a large simulation package for predicting patterns of urban development for periods of twenty years or more, under different possible scenarios (Waddell and Borning, 2004). UrbanSim's primary purpose is to provide urban planners and other stakeholders with tools to aid in more informed decision-making, with a secondary goal to support further democratization of the planning process.

As Schmidt (1991) points out, models are limited abstractions and must evolve with the world they reflect. Users should be able to appropriate systems creatively to adapt them to particular situations. One important characteristic of the urban planning domain is that it centers on public deliberation and decision-making involving multiple stakeholders, who often have very different values and perspectives with respect to land use and transportation. Furthermore, each region in which UrbanSim has been applied has unique physical, social, and political characteristics. In this position paper, we discuss three contexts of appropriation of UrbanSim: by urban planners and modellers in different urban regions, by advocacy groups and other organizations, and by individual citizens. The first is operational and the others in the early design stages.

2 Appropriation by Different Metropolitan Regions

From the start of the project, UrbanSim has been designed as a reusable modeling system, for use by many different metropolitan regions—which, given the variation among regions, requires that it be designed for appropriation by urban planners and modellers. To date, UrbanSim has been applied experimentally in the U.S. in metropolitan regions around Eugene/Springfield, Oregon; Seattle, Washington; Honolulu, Hawaii; and Salt Lake City, Utah. Working largely independently of the UrbanSim team, groups have also experimentally applied UrbanSim in Houston, Texas; Phoenix, Arizona; Paris, France; and Tel Aviv, Israel; with other applications in process. UrbanSim played a significant role in an out-of-court settlement in Utah regarding a major freeway construction project (Waddell and Borning, 2004). The first major use in a public planning process is scheduled to begin in the Puget Sound (Seattle) region in summer 2005.

To enable the UrbanSim software engineering team to respond more readily to requests from modellers applying the system in different regions, the system architecture uses a collection of component models that interact via a shared database rather than by invoking each other directly (Noth et al., 2003). This

modular architecture, implemented in Java, makes it easier to modify individual component models without modifying other components of the system. The team also uses an agile development methodology (Freeman-Benson and Borning, 2003), to be more flexible and responsive to modeller requests.

Licensing UrbanSim under the GNU Public License is another important step in supporting appropriation for different regions. While open source licensing is familiar in computer science, it is less common in urban modelling. Most urban simulation systems are proprietary, and there are a variety of barriers to sharing code improvements among the government agencies in different regions. By contrast, UrbanSim is freely available for download from the project website (http://www.urbansim.org). Downloadable information includes the source code, executable code, a sample dataset, and a user manual.

As open source software, all parts of the system can be modified by the end users, but of course this is not always easy in practice. One part of the system where such extension is more straightforward is in the mechanisms for computing and viewing indicators. In urban planning, indicators are often used to monitor changes in a region with respect to specific attributes of concern. In UrbanSim, simulation results can be presented using the same set of selected indicators for all the policy alternatives being considered, aiding the assessment and comparison of different scenarios. To make it easier to modify or add new indicators, raw simulation results are stored in an SQL database. The indicator computations are then expressed as SQL queries, decoupling them from the simulation itself. We have also done considerable design and testing work on the indicator documentation and interface, addressing issues such as information fragmentation and transparency of the system, and ensuring that documentation about the indicators is ready-to-hand in the course of using them (Borning et al., 2005).

The software engineering group had hoped that writing the Java source code well, with good abstractions, coding style, comments, and documentation, would enable the domain experts (modellers) to read the code and make some changes—in other words, to support a further level of appropriation. However, this rarely happened in practice, because modellers found Java and its interactive development environment too daunting. However, the software engineering group found that the modellers *are* willing to read and write in a high-level scripting language, namely Python. Another factor has been a desire to join forces with other land use and transportation modelling groups world-wide, to provide a common platform and system that enables greater sharing and collaboration. In response, the group has built a new Python framework, the Open Platform for Urban Simulation (OPUS), and begun the process of converting UrbanSim to be built on this framework. We have already found that the modellers are indeed willing to read and make simple changes to the Python code.

As another step toward supporting appropriation by the wider community of modelers, a group at the University of Massachusetts has established the

UrbanSim Commons (http://www.urbansimcommons.org). The UrbanSim Commons provides a place where UrbanSim users and developers can share knowledge and experiences.

3 Appropriation by Organizations

A recent paper (Borning et al., 2005) discusses the development of Technical Documentation for UrbanSim indicators, guided by the Value Sensitive Design theory and methodology (Friedman et al., in press). As much as possible, the Technical Documentation is intentionally neutral, and does not advocate for any particular use of the indicators to evaluate policy alternatives. Yet, the planning process is rife with strong opinions and perspectives. Indicator Perspectives support organizations in appropriating UrbanSim indicators and simulation results to advocate for their own positions. In the Indicator Perspectives section of the UrbanSim website, a set of organizations each present their own views on which indicators are most important for evaluating policy alternatives, and how those indicators should be interpreted. We believe that these perspectives will be also useful to stakeholders and decision makers because the organizations have well thought-out positions and can present them clearly and coherently. Indicator Perspectives are intended to provoke thought and public deliberation, as well as to give groups a venue in which to state their positions.

We are currently in the early stages of developing Indicator Perspectives. We have partnered with three local organizations to construct perspectives for the initial prototype: a government agency (King County Budget Office, which publishes the King County Benchmark Reports), a business association (Washington Association of Realtors), and an environmental group (Northwest Environment Watch). Later, we plan to provide opportunities for involvement to all who are interested, actively soliciting partners as needed to help ensure continuing coverage of the political and policy space.

4 Appropriation by Individuals

A natural question for any citizen learning about a new government policy is, "How will this affect me?" A new mechanism under development, Personal Indicators, is intended to address this question for land use and transportation policy alternatives simulated with UrbanSim. As this question is necessarily tied to each citizen's particular situation, users of Personal Indicators will begin by providing some information via a web-based interface about their situation, such as the neighbourhoods in which they live and work, approximate household income, and the number of adults and children in the household. Values of Personal Indicators for the simulated future under each policy alternative would

then be provided. Rather than reflecting the region as a whole, Personal Indicators will reflect the user's individual and family situation: for instance, the amount of time required for the user's commute to and from work, the mix of commercial and residential construction in the user's neighbourhood, and housing options throughout the region for similar households.

We hypothesize that Personal Indicators can help to engage citizens in the urban planning process by addressing the question, "How will this affect me?" They may also be more comprehensible to ordinary citizens than indicators at the metropolitan level, because they can be readily related to an individual's everyday experiences of living, working, and getting around in the region. However, Personal Indicators also raise significant questions for the use of UrbanSim in a democratic society. How can we resolve tendencies to take a short-term view of the future and with the long-term view required by regional planning? How can we balance an individual's self-interest with interests of other individuals and the good of the region as a whole? How can citizens using Personal Indicators engage in deliberation when each has a different view of the future?

5 Acknowledgments

Thanks to Peyina Lin for her feedback on this paper. This research has been supported in part by the U.S. National Science Foundation under Grant Nos. EIA-0121326 and IIS-0325035, and in part by gifts from IBM and Google.

6 References

Borning, A., Friedman, B., Davis, J., and Lin, P. (2005): 'Informing Public Deliberation: Value Sensitive Design of Indicators for a Large-Scale Urban Simulation', to appear at the *9th European Conference for Computer-Supported Cooperative Work (ECSCW 2005)*.

Freeman-Benson, B., and Borning, A. (2003): 'YP and Urban Simulation: Applying an Agile Programming Methodology in a Politically Tempestuous Domain', *Proceedings of the 2003 Agile Development Conference*, Salt Lake City, Utah, June 2003.

Friedman, B., Kahn, P. and Borning, A. (in press): 'Value Sensitive Design and Information Systems', to appear in P. Zhang & D. Galletta (eds.): *Human-Computer Interaction in Management Information Systems: Foundations*, M.E. Sharpe, New York. Available from www.ischool.washington.edu/vsd.

Noth, M., Borning, A., and Waddell, P. (2003): 'An Extensible, Modular Architecture for Simulating Urban Development, Transportation, and Environmental Impacts', *Computers, Environment and Urban Systems*, Vol. 27 No. 2, March 2003, pages 181—203.

Schmidt, K. (1991): 'Computer Support for Cooperative Work in Advanced Manufacturing', *International Journal of Human Factors in Manufacturing*, vol. 1, no. 4, October 1991, pp. 303-320.

Waddell, P., and Borning, A. (2004): 'A Case Study in Digital Government: Developing and Applying UrbanSim, a System for Simulating Urban Land Use, Transportation, and Environmental Impacts', *Social Science Computer Review*, vol. 22 no. 1, pp. 37-51.

Mobile Collaborative Software Adaptation

Matthew Chalmers,
Malcolm Hall,
Marek Bell
Computing Science, University of Glasgow, UK
matthew@dcs.gla.ac.uk

Abstract. Adaptive systems are often constrained by the complexity of designing for unexpected uses and preferences, integrating new software into existing systems, and supporting users in understanding and controlling system structure. In our system, Domino adaptation is driven by recommendations generated from logs of users' activity. More efficient and enjoyable functionality can be gained through contact with other users who have been in a similar context. We demonstrate the use and utility of the approach by presenting a prototype game in which players can adapt their system with recommended upgrades in order to progress through the game with improved tools, increased efficiency and enjoyment.

1 Introduction

System adaptation and evolution are especially important as the use of computers expands beyond work activities focused on pre–planned tasks into ubiquitous computing (ubicomp) for leisure and domestic life. Here, the variety and dynamics of people's activities, contexts and preferences make it especially hard for the designer to foresee all possible functions, modules, their transitions, combinations and uses. Instead of relying on the developer's foresight, incremental adaptation and ongoing evolution under the control of the user maybe more appropriate (Edwards 2001, Rodden 2003). Our system architecture

‚Domino' actively supports incremental adaptation and ongoing evolution of ubicomp systems. In effect, it changes a system's structure on the basis of the patterns of users' activity. It supports each user in finding out about new software modules through a context–specific collaborative filtering algorithm, and it integrates and interconnects new modules by analysing data on past use. Domino allows software modules to be automatically recommended, integrated and run, with user control over adaptation maintained through acceptance of recommendations rather than through manual search, choice and interconnection.

Each instance of the Domino system consists of three parts that manage and record the use of modules, handling communication, recommendation and adaptation respectively. A Domino module consists of a group of .NET classes stored in a DLL (Dynamic Link Library). Domino is implemented entirely in C# and compiled for the .NET compact framework. It relies on a database to store history logs: MS SQL Server on desktop machines and SQLCE on PDAs. Figure1 gives an overview of Domino's structure.

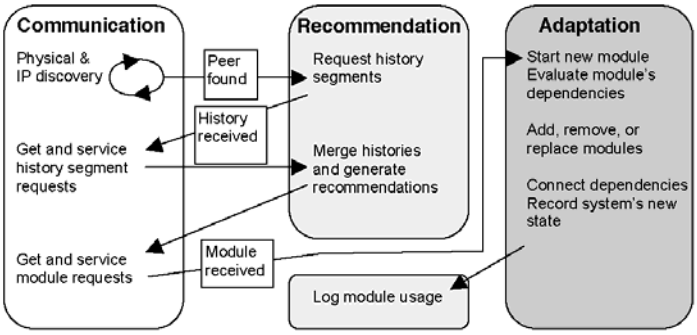
A Domino system continually broadcasts its existence over any connections available on the local device—such as 802.11 wireless or wired Ethernet. Domino systems also continually scan for available networks and devices, connecting to

802.11 infrastructure networks when available or creating their own ad hoc networks. When two Domino systems meet they immediately start transferring user history data. This data includes history data such as where the user went,what web pages they browsed and which Domino modules he or she ran. History data is passed on not only about the owner of the system but also about other people that the owner has previously encountered. In effect this is a simple epidemic algorithm (Demers 1987, Khelil 2002) offering a degree of consistency among distributed databases of history information. Domino hides all logs from the user, offering no direct interface to history data. Any receipt of history data triggers Domino's recommendation component, as new data offers new module recommendations. The recommender is also triggered by new modules being configured and run manually by the user. Recommendations are anonymous. The recommender takes the user's current set of modules, i.e. the user's 'context' in terms of modules, and compares it with the history data it has collected, and recommends new modules often used by other users in similar contexts. Essentially, this uses the same context–specific collaborative filtering algorithm applied to URLs in (Chalmers 1998).

Once a module is installed, the system may automatically start it, ask the user if he or she wants to run it, or add it to a list of recommended modules that can be browsed at the user's leisure. The programmer can select from these toolkit options depending on the application and community of use. We assume that normally a user would be offered the ability to prevent any particular system adaptation by refusing a recommendation. Domino also supports removal of modules, including stepping back through recent additions so as to let the user

rollback the system. Due to the generic nature of the system model, when a module is received there is no predetermined place for it in the system. In the simplest case,

Figure 1: Overview of Domino components



the new module can query the Domino system's running modules to find ones that satisfy its dependencies, by analysing their classes and the interfaces they implement. This allows us to choose the most appropriate 'parent' for the new module and we add it as a dependency. When multiple suitable modules are found, we can obtain a ranked list of modules previously used in conjunction with the new module, and check if one of them is currently connected to instances of the modules in the ranked list, i.e. one having the most items in the ranked list as dependencies.

In our current prototype system, modules and the protocols for their transmission are sufficiently unusual for us to feel that we can carry out our initial experiments without employing more heavyweight security measures than our minimal ones of keeping logs hidden and using anonymous recommendations. However, more general or widespread use would of course demand such measures, we are investigating signed code modules and .NET Code Access Permissions which allow the programmer to allow a range of permissions for a module to be set which specify access level for other parts of memory, code, hardware and file space.

2 Early Experience

To test the Domino architecture we developed a mobile strategy game, *Castles*. Creating the game is part of an ongoing project using mobile games to explore the deliberate exposure of system infrastructure to users in a 'seamful' way, as in (Borrielo 2005) so that users might be aware of or even take advantage of variation in the deployed configurations of system infrastructure. Similarly, we

are interested in selectively exposing and giving control of software structure to users.

The majority of the Castles game is played in a solo building mode, in which the player chooses which buildings and tools to use, and how many resources to use for each one (Figure 2). Each type of building and tool is a Domino module. The goal of this stage is for the player to create an infrastructure that efficiently constructs and maintains the player's army units. When the game starts, there are over thirty types of building and eleven types of army units available to the player, allowing for extremely varied combinations of buildings supporting distinct types of army. Tools may have different effects based on which building they are applied to. For example, the scythe tool has no effect if applied to the Knight School but doubles output levels when applied to a wheat field. In order to mimic the way that plug–ins and components for many software systems continually appear over time, new buildings, tools and units are introduced throughout the game, as upgrades and extensions that spread among players as they interact with each other. When two players' PDAs are within wireless range, one may choose to attack another. Behind the scenes, Domino also initiates its history–sharing and module–sharing processes. When a battle commences, both players select from their army the troops to enter into battle. Players receive updates as the battle proceeds, and at any time can choose to retreat or concede defeat. At the same time, players can talk about the game, and the modules they have recently collected, and modules they have used, and found useful or discarded.

With such a high number of buildings, tools and units, there is significant variation in the types of society—module configurations—that a player may create. Selecting which buildings to construct next or where to apply tools can be a confusing or daunting task. However, Domino helps by finding out about new modules as they become available, recommending which modules to create next, and loading and integrating new modules that the player accepts.

We have run pilot tests with students from our university department playing the game in a building away from the university and its networks, and we can offer some initial anecdotal evidence of the system's use. Each player started with the same base set of buildings, adapters and units available. Each was also initially given game resources that were different to those given to others: two extra buildings, two extra adapters and one extra unit. Thus, each player started with a substantial core set of items (thirty-three buildings, ten building adapters and eleven units) plus five items that were unique to him or her. For example, amongst the additional items given to one player was the catapult factory that constructs catapult units. As anticipated, when players met for battle, their

Domino systems exchanged usage information and transferred modules between PDAs so as to be able to satisfy recommendations. Several players who had been performing poorly because of, for instance, a combination of buildings that was not efficient for constructing large armies, felt more confident and seemed to improve their strategies after encountering other players. They started constructing more useful buildings by following the recommendations, with the system showing how or where modules can be used based not only on general or objective fit, but with specific patterns of use in play.

Overall, this early experience has been promising and productive—but preliminary. We are now planning a larger trial involving participants recruited from the public. The basic system structure will stay the same, but we are making minor changes to the interface. In future, we hope to report on the details of these trials, both in qualitative terms, e.g. how people understood and interacted around the dynamic process of recommendations and system changes as they move through the city, and in quantitative terms, e.g. the rates and statistics of module transmission, sharing and spatial movement in the course of the trial.

3 References

Borriello, G. et al. Deploying Real–World Location Systems, CACM 48(3), March 2005, 36–41Chalmers, M. et al. The order of things: activity-centred information access. *Proc. WWW 1998*.359-367.

Demers A. et al, Epidemic algorithms for replicated database maintenance, *Proc. 6th ACMSymposium on Principles of Distributed Computing* (PODC), 1987, 1-12

Edwards, W.K., Grinter, R. At Home with Ubiquitous Computing: Seven Challenges. *Proc.Ubicomp 2001,* Springer LNCS, 256–272

Khelil, A, et al. An Epidemic Model for Information Diffusion in MANETs, *Proc. ACM MSWIiM*,

Rodden, T., Benford, S. The evolution of buildings and implications for the design of ubiquitousdomestic environments. *Proc. ACM CHI 2003*, 9–16.

Socio-Technical Self-Descriptions as a Means for Appropriation

Gabriele Kunau,
Thomas Herrmann,
Kai-Uwe Loser
Information and Technology Management,
Institute for Applied Workscience, Ruhr-University Bochum
{*thomas.herrmann, gabriele.kunau, kai-uwe.loser*}*@rub.de*

1 Introduction

Processes of appropriation of groupware systems in organizations are social processes that are strongly intertwined with processes of reflection and communication. We suggest facilitating these processes by supporting the creation of socio-technical self-descriptions. Self-description is an important element of social systems such as organizations and can be extended to include descriptions of an organization's usage and adaptation of technology. In the following we discuss the concept of self-description; how support for self-description can effect appropriation of groupware systems; how the analysis of self-descriptions could be a measure for the degree of appropriation that has already taken place in an organization; methods for supporting self-description in the context of groupware; empirical experience with facilitating self-descriptions in projects.

2 Self-Descriptions

Self-Description in System Theory

The relevance of self-descriptions for the understanding of the development of organizations is elaborated in system theory. In contrast to other types of systems such as e.g. biological systems, social systems do not possess anything physical – like a membrane – that constitutes their boundary to the environment. Social systems must maintain their boundaries in a continuous process of negotiation deciding which communicative acts are acceptable within the system and which are not. As an orientation for this process of distinguishing between outside and inside, social systems create and use self-descriptions that allow them to make a difference between the system itself and its environment (cf. Luhmann, 1995, p. 196).

Self-descriptions occur in many forms, a few examples are given to illustrate the concept.

Examples for Self-Descriptions in Organizations

An **organization chart** that describes, who belongs to which department, is a self-description: it bears implications about hierarchical structures and information-flows. An **organization's mission** statement is a self-description, because it includes values that (should) guide the behavior within the organization. An **ISO-9000 process description** can be a self-description, because it describes expectations how certain tasks need to be carried out. All of these documents contain expectations that direct the individuals' behavior within the organization.

But self-descriptions do not only exist in written form as sustainable documentation. Self-description can also occur in more volatile modes such as oral communication or e-mail communication. A tradition like "we go for lunch each day at 12:30" can be part of a team's self-description; such traditions are usually not contained in official documents, they are rather passed on orally or in ephemeral electronic communication like chat.

If the organization uses a groupware system, then another, special, form of self-description is added: those self-descriptions that are inscribed in the groupware. Take **workflow systems** as an example: models of the organization's processes are encoded into a workflow system which then controls the coordination between its users. Also aggregated awareness data can be considered as a kind of self-description.

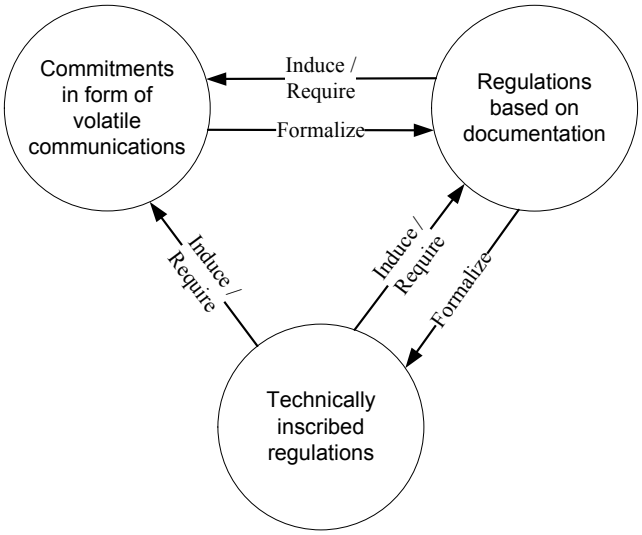
An organization's self-description is never one large canonical document, different forms of self-description add to the overall picture. The next section uses these examples to derive a more abstract description of the concept self-description.

3 Three levels of self-descriptions

Self-description within a social system takes two forms of appearance: as a process and as an artifact. As a process, self-description is made up by the continuous communications within the organization that keep alive its essential characteristics, norms and values. In this way, the process of self-description maintains the organization's identity as a unique social system that is distinguishable from other social systems.

Since communicative acts are ephemeral, organizations create artifacts that make important parts of the communications more permanently available. These artifacts are usually combinations of texts, graphics and other symbolic means.

In the context of socio-technical systems (e.g. organizations having appropriated a groupware) three levels of self-descriptions can be distinguished:



There are the volatile agreements that exist only as oral communicative acts; then there are documented regulations and then there are those rules that have become part of the groupware systems. The latter occur in various forms, for instance:

- the way how menus are organized
- the contents and structures of electronic forms
- the hierarchical structure of folders
- the sequence of actions in workflow management systems …

All these characteristics of the groupware system do not merely fulfill a functional purpose, but they also describe characteristics of the social system and its way of using and adapting – and eventually appropriate - the technical system.

But no matter how deliberate the design process was, no technical system can unambiguously prescribe its usage. Therefore the organization will agree to

additional rules concerning the usage of the groupware system in the course of appropriation. As a result socio-technical self-descriptions occur in three forms as shown in the figure above.

The arrows between the three forms of self-descriptions indicate that there is an interchange between them. A self-description that first exists only informally e.g. in form of an oral agreement may be formalized and become part of an official document. Similar, the statements in a document may be implemented as features of a groupware. The other way around, a formal agreement always needs informal agreements into which it is embedded. The arrows leading from "technically inscribed regulations" to "regulations based on documentation" and "commitments in form of volatile communications" indicate this necessity.

4 Self-Descriptions and Organizational Change

What is the relationship between self-descriptions and appropriation of groupware? Processes of appropriation are processes of organizational change; and self-descriptions can be used to support processes of organizational change.

Self-Description as a Means of Systemic Intervention

Methods of systemic intervention are methods based on concepts and insights of system theory that support processes of organizational change. The most important characteristic of systemic intervention is that it attempts to consolidate two seemingly contradictory aspects: intervention that strives to induce change processes towards a specific goal on the one hand, and the respect for the self-organizing characteristics of social systems on the other.

Self-descriptions are an important aspect of systemic interventions:

a) Self-descriptions are necessary because they provide stability for the system by defining its boundaries and making basic regulations comprehensible
b) Questioning the self-description can initiate processes of self-reflection and subsequent change that lead to a new self-description.

Deployment of Groupware in an Organization

The deployment of a new groupware system effects a change within the organization that needs to be reflected in the organization's self-description. The organization needs to describe how the new technical system is integrated into its network of communications. The process of including a new groupware system into the organization's self-description is part of the process of appropriation.

We find the concept of self-descriptions fruitful for understanding and supporting appropriation processes of groupware systems because it combines two quite different aspects:

a) Planned rather than completely arbitrary appropriation of a groupware system within an organization is necessary in order to achieve the goals for which the system was designed.
b) Appropriation is a social process that is promoted in ways which are not comparable to the engineering processes for groupware systems.

5 Self-descriptions as Support for Appropriation

Our approach to supporting processes of appropriation is to support an organization in creating and maintaining socio-technical self-descriptions. The organization should

- explicate its usage of the groupware,
- discuss alternative options,
- reach conclusions about the usage,
- document the conclusions.

Functions of groupware systems can be used to support communication processes which promote the process of appropriation. This field has been explored extensively by the work of Volkmar Pipek (2005).

Our work tries to bind together three types of methods and instruments which aim at the successful adoption of technology and the evolution of its usage:
(1) A modeling method that provides symbolic means which are specifically suitable for the creation of written forms of socio-technical self-descriptions:
SeeMe, the diagramming-technique for modelling semi-structured socio-technical systems supports modelling of (technical inscribed) formal processes but also provides special modelling concepts for the representation of vagueness, incompleteness, and contradictions that are inherent to rules and agreements in organisations (Herrmann et al. 2000).
(2) An editor with which socio-technical models can be elaborated as well as presented in co-located workshop settings: Self-description, from one point of view, is a communication process where practice is reflected. Using complex diagrams for this purpose needs help to reduce complexity and focus certain aspects. The SeeMe-Editor is specifically designed to support step by step presentation of models as well as modifying diagrams in between the presentation, to visualize the results of the ongoing discussions developing the self-description.
(3) The socio-technical walkthrough (STWT) as a method for systematically facilitating communications in a series of workshops (Herrmann et al 2004):
The core idea of the STWT is that the concept or outline of a socio-technical system is represented by a diagrammatic model which is the outcome of a participatory design process. This model is either developed from scratch or is derived from an existing model – which usually presents the given state of the work processes – by gradually modifying its elements with respect to the

technology to be introduced. A model has to be inspected step by step before it is considered as the final solution, upon which most of the participants can agree.

6 Self-Descriptions as a Measure for Appropriation

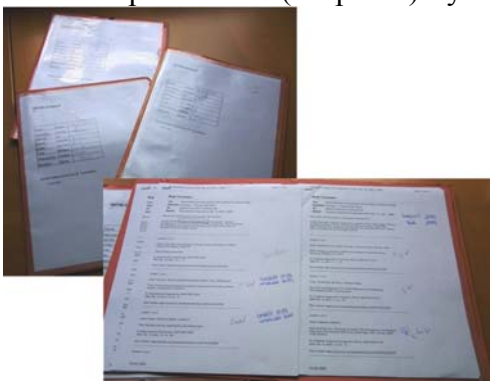
So far socio-technical self-descriptions have been introduced as a means for supporting processes of appropriation. However, we also think that they could be used as a measure to judge whether and how deeply an organization has appropriated a groupware. The overlapping between the different forms of self-description can be taken as a measure for the process of appropriation: the more the symbolic structures of an IT-solution cover the self-description of a social system and its ways of interacting with the computers, the more has the process of appropriation evolved. And the more the self-descriptions refer to the groupware system and how it should be used, the more is this system incorporated into the organization

7 Empirical Work

During the past years we conducted numerous case studies in which the triad of modelling notation, editor and workshop-concept was employed, analyzed and improved. Among these case studies were:

- (1) KatEr – Planning new work procedures in a university library in the light of a new groupware system (Loser, 2002)
- (2) Modeling to define the structure and content of a knowledge management system for a consumer counseling agency (Herrmann et al, 2002)
- (3) SpiW – Socio-technical design of a mobile application for logistic companies (Herrmann et al, 2004)
- (4) Process Maps to improve collaborative learning (Carell et al, 2005)

For this E-CSCW workshop we provide empirical material from our latest case study named “ELISE”. The University of Dortmund replaced its procedure of circulating paper copies of the contents of scientific periodicals (cf. photo) by a system that sends out e-mails to inform the scientific staff about new issues and their content.



Within our work group we decided to design and implement an **e**lectronic **li**terature **s**yst**e**m that ingests the e-mails and provides cooperative functions to support the easy communication and

coordination that was previously realized by notes on the circulating paper copies.

The figure on the last page illustrates how aspects of socio-technical self-descriptions are realized for ELISE.

- The structure of the menu buttons reflects the collaboration and the way in which the group works with journals; e.g. there are buttons to recommend certain articles to others.
- There are additional process diagrams that contain commitments about the way the group uses the system; e.g. the scientific staff will try to view new journals by Thursday of the following week; the students in the library will send out reminders every Thursday.
- As an example for communication beside the main regulations, an e-mail is attached in which one colleague asks to postpone the deadline. The e-mail demonstrates that the commitments (here, to skim through the journals until Thursday) are taken seriously but that it is also possible to agree to spontaneous changes.

8 Further Research

We argued for socio-technical self-descriptions as a concept to support processes of appropriation and as a basis for measuring how complete a process of appropriation of a groupware is. We also gave an empirical example of the concepts' relevance, also referring to the experience of various earlier projects.

The focus on socio-technical self-description suggests some interesting future research questions:

- When and where in the Software-Lifcycle can the preparation of the appropriation start by supporting socio-technical self-description (with respect to technical structures, the growing documentation, participatory design etc.).
- How far can software design and appropriation be overlapped and how far can socio-technical documentation serve as a boundary object to manage this overlapping?
- By which extent can the process of appropriation be actively promoted from outside or inside by referring to socio-technical self-descriptions.
- Can a "more or less" of appropriation be measured by analyzing the occurrence of socio-technical self-description?

9 References

Carell, Angela; Herrmann, Thomas; Kienle, Andrea; Menold, Natalja (2005): Improving the Coordination of Collaborative Learning with Process Models. Proceedings of CSCL 2005. The Next 10 Years. May 30 - June 4, 2005. Taipee.

Herrmann, Thomas; Hoffmann, Marcel; Loser, Kai-Uwe; Moysich, Klaus (2000): Semistructured models are surprisingly useful for user-centered design. In: Dieng, R.; Giboin, A., Karsenty, L., De Michelis, G. (Eds.)(2000): Designing cooperative systems. Proc. of COOP2000. Amsterdam: IOS press. pp. 159 – 174.

Herrmann, Thomas; Hoffmann, Marcel; Kunau, Gabriele; Loser, Kai-Uwe (2002): Modelling Cooperative Work: Chances and Risks of Structuring. In: M. Blay-Fornarino et al. (Eds.): Cooperative Systems Design, A Challenge of the Mobility Age (Coop 2002). IOS Press. S. 53-70.

Herrmann, Thomas; Kunau, Gabriele; Loser, Kai-Uwe; Menold, Natalja (2004): Sociotechnical Walkthrough: Designing Technology along Work Processes. In: Andrew Clement, Fiorella Cindio, Anne-Marie Oostveen, Doug Schuler, Peter van den Besselaar (Hrsg.), Artful Integration: Interweaving Media, Materials and Practices. Proceedings of the eighth Participatory Design Conference 2004. ACM, New York. S. 132-141.

Loser, Kai-Uwe; Herrmann, Thomas (2002): Enabling factors for participatory design of socio-technical systems with diagrams. In: Binder, T.;Gregory, J.; Wagner, I. (Eds.): PDC 02 - Proceedings of the Participatory Design Conference. Malmö, Sweden, 23-25 June 2002. CPSR, Palo Alto, CA. S. 114-143.

Luhmann, Niklas (1995): Social Systems (Translated by Bednarz Jr., John with Baecker, Dirk). Stanford, California: Stanford University Press..

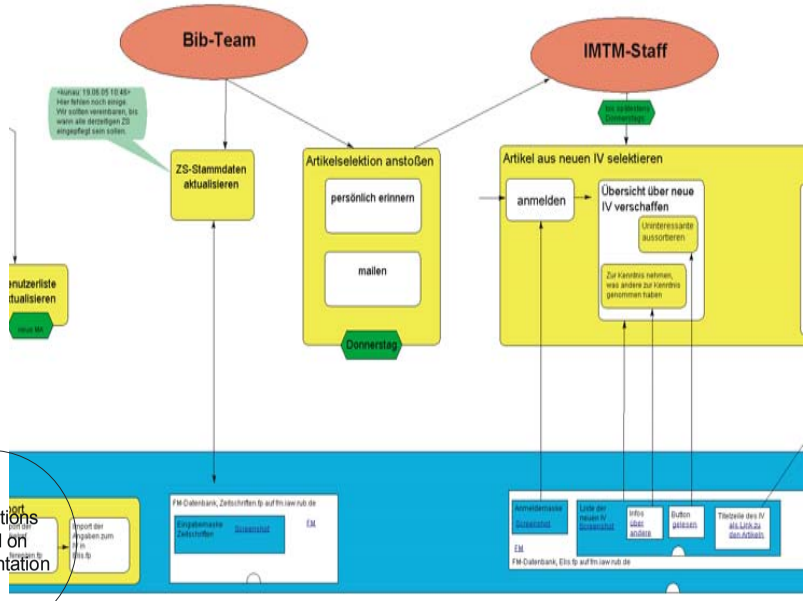
Volkmar Pipek (2005): From tailoring to appropriation support: Negotiating groupware usage. University of Oulu.

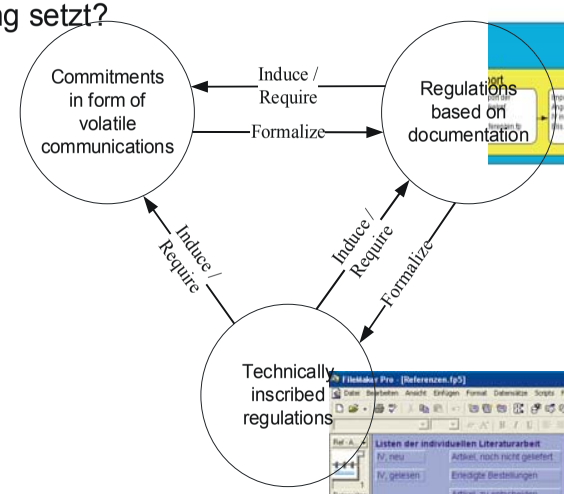
Von: Kai-Uwe Loser
Gesendet: Samstag, 7. Mai 2005 13:58
An: Alexandra Frerichs; Angela Carell
Cc: Carsten Ritterskamp; Gabriele Kunau; Isa Jahnke; Michael Prilla; Rainer Skrotzki; Thomas Herrmann
Betreff: AW: Elise-Tag

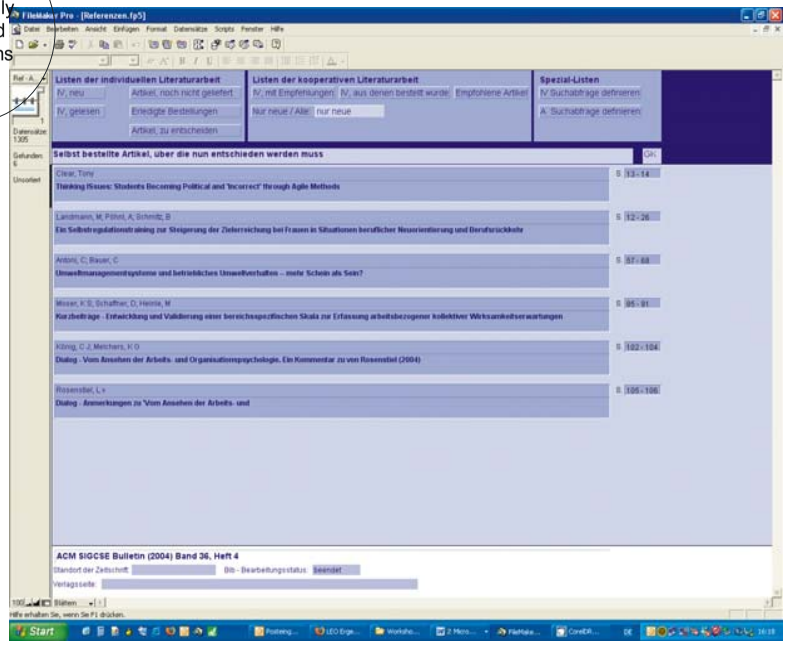
Bitte um Verlängerung...

Ich habe mir bisher noch keine IV angeschaut. Ginge das, dass ihr wegen des Feiertags ;-) die IV erst am Dienstag abend auf Working setzt?

Lus







Supporting Appropriation Work with Social Translucence, Collective Sensemaking, and Social Scaffolding

Wendy A. Kellogg,
Thomas Erickson
IBM T.J. Watson Research Center
Yorktown Heights, NY USA
wkellogg@us.ibm.com, snowfall@us.ibm.com

Abstract. How do users come to understand the capabilities of online environments in order to adapt them to their own purposes? We have argued elsewhere that creating socially translucent systems – those that support mutual awareness and accountability by providing perceptual cues about participants' presence and activities – is a key enabler for the emergence of social behavior and norms. In this paper we analyze three cases in which 1) users engage in collective sensemaking to understand an unfamiliar interface feature (the Babble social proxy), 2) the Babble designers and leader of a community of interest collaborate to provide social scaffolding to help establish healthy norms and practices in an online environment, which in turn allows new practices to emerge; and 3) designers of a broadcast messaging tool make a small interface change that enhances the ability of a population of users to self-regulate and thereby successfully appropriate a new technology. We argue that designing systems to be socially translucent facilitates social interactions like sensemaking and scaffolding that are critical to appropriation work.

1 Introduction

For the past several years, the Social Computing Group at IBM's T.J. Watson Research Center has been designing socially translucent online environments by making cues about presence and activity visible to users. We believe that such systems — by supporting mutual awareness and accountability[1] — will make it easier for people to carry on coherent discussions; to observe and imitate others' actions; to engage in peer pressure; to create, notice, and conform to social conventions; and to engage in other forms of collective interaction, including sensemaking and scaffolding. We use the phrase "social translucence" as a rubric

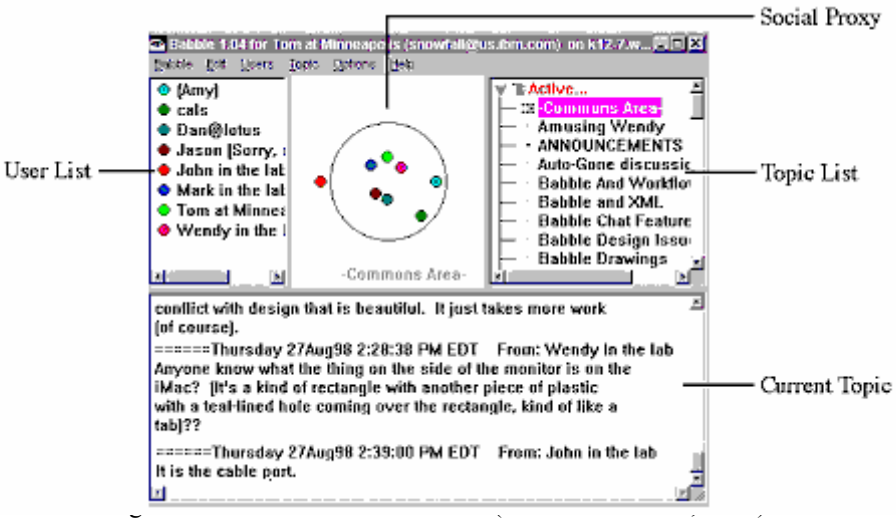
[1] Eriksen (2002) provides an interesting discussion of three views of accountability (following Garfinkel's "everyday" accountability, Suchman's "located accountability," and Dourish's "system accountability"). Accountability with respect to socially translucent systems is closest to Garfinkel's concept in the sense that making presence and activity visible in the course of everyday (online) activities both enables and demands accountability to others for one's actions.

for our approach to designing such systems. “Social,” of course, signals our interest in providing cues that are socially salient. “Translucence” has a more nuanced role: Most evidently, in an implicit contrast to “transparence,” it indicates that our aim is *not* to make all socially salient information visible. Translucence stands in for the notion that, in the physical world, cues are differentially propagated through space — something which, as social creatures, we understand and make use of in governing our interactions. Thus, we know that those across the room may see *that* we are talking, but will be unable to hear *what* we say; and we adjust our interactions to take advantage of this. If we might call this the ‘social characteristics of (physical) space;’ it suggests a design goal of creating similar regularities in the propagation of social cues in online environments (Erickson et al., 2002).

Sensemaking and scaffolding are central activities in which collectivities of people — teams, groups, communities of interest, or societies — engage. Individuals engage in sensemaking to regulate their behavior in the context of groups in which they participate. Groups regulate their behavior in part by establishing interactive norms and conventions. In this paper, we argue that for users to appropriate technology they must both understand its capabilities and have scaffolding mechanisms for collectively discovering, structuring, iterating, and promulgating practices that enable the technology to become what Ackerman et al. have termed a ‘resource’ (Ackerman et al., in preparation). To examine these propositions more concretely, we look at sensemaking and scaffolding in three examples of socially translucent technologies designed to aid people interacting online.

2 Group Sensemaking in Babble: The Social Proxy

Babble was designed to serve the communication needs of small to medium-sized corporate groups. It was intended to provide a semi-private online conversation area where members of groups such as teams, work groups, committees, and special purpose task forces could have text-based synchronous or asynchronous conversations. Figure 1 shows a screenshot of the Babble user interface. In the upper middle pane of the window is a visualization called the social proxy. Its purpose is to provide cues about the presence and activity of those in the current conversation. People in the conversation are shown within the circle; people logged on but in other ‘rooms’ (the list to the right) are positioned outside the circle; and, most importantly, when people are “active” in the conversation, meaning that they either ‘talk’ (type) or ‘listen’ (click and scroll), their dots move to the inner periphery of the circle, and then gradually drift back out to the edge over the course of about 20 minutes.



Over the course of the last seven years, we have deployed Babbles to a few dozen groups, and conducted a variety of studies of its adoption and use.[1] The social proxy is an unfamiliar element to new users of Babble, but readily attracts attention as people try to figure out what it is and how it works. Often there is one or more experienced Babbler around to answer questions. But in the excerpt shown in Figure 2, all of the users were new to Babble and none knew what the social proxy was for; this evoked the following example of group sensemaking.

> Jessica@San Jose Monday 23Jul01 1:13:45 PM EST
>
> what do the circles with different numbers of dots in them mean? (e.g., next to some peoples' names in the personal places?)
>
> dan@chicago Monday 23Jul01 1:13:49 PM EST
>
> i like the round table
>
> Randy@boston Monday 23Jul01 1:14:06 PM EST
>
> We look like dancing M&Ms
>
> dan@chicago Monday 23Jul01 1:14:22 PM EST
>
> if it can show people's icon or picture, ...
>
> Chen@China Monday 23Jul01 1:15:14 PM EST
>
> Are the people outside of the gray circle not in the chat room? For e.g. Ram are u not in the chat room?

[1] For a list of recent publications, see www.research.ibm.com/SocialComputing/SCGpapers.htm.

Lori@Italy Monday 23Jul01 1:15:33 PM EST

!

Ram@Mountain View Monday 23Jul01 1:15:35 PM EST

Hello, World!

BillY@Atlanta Monday 23Jul01 1:15:46 PM EST

Chen, it looks like they are in a different Babble area

Becki Raven@Armonk Monday 23Jul01 1:15:59 PM EST

The circles move when an individual moves...

Chen@China Monday 23Jul01 1:17:03 PM EST

Becki, you just moved out of the gray circle...what does it mean? Hv you gone to a differnt Babble area?

Mitchell@Toronto Monday 23Jul01 1:17:56 PM EST

When you enter a room/area, your marble will move to the center and you can tell who else is in the area.

William@Mountain View Monday 23Jul01 1:18:12 PM EST

Interesting, is there any significance to the differences in proximity to the center for some dots vs. others? Why are some closer to the middle and some more towards middle/outer radius?

Mitchell@Toronto Monday 23Jul01 1:18:29 PM EST

Hmmm... need more testing to find out!

Susanne@Germany Monday 23Jul01 1:18:46 PM EST

Hi

BillY@Atlanta Monday 23Jul01 1:18:53 PM EST

William, it looks like an activity statement... the longer your idle the further from the center you are.

Caitlin@SanFrancisco Monday 23Jul01 1:19:10 PM EST

perhaps the people who came in recently are further away from the core of the commons area?

William@Mountain View Monday 23Jul01 1:19:20 PM EST

Thanks for the info Bill, I'm going to see if I move in closer as a result of typing this message...

Rhonda@UK Monday 23Jul01 1:19:53 PM EST

as soon as u send a message u get closer to the center

Figure 2. An excerpt from a Babble as new, untrained users engage in sensemaking to figure out the Babble social proxy or "cookie." (Note: Names have been changed).

This excerpt is interesting in a number of ways. First, it is remarkable for how distributed and rapid the sensemaking is: 13 people from all over the world make 18 utterances over the course of 6 minutes. The people have never met and have never been in a Babble space. Yet they rapidly converge on the correct interpretation of the social proxy. The persistence of the chat facilitates participants building on each other's questions and observations. The excerpt also underscores the importance of common ground in allowing sensemaking to take place; everyone sees the same changes in the environment as they and others take action, and they see themselves as others see them — i.e., from a "third person" point of view. When everyone sees the same thing, it creates a coherent basis for conversation. When users get feedback on their own actions by seeing themselves as others will see them, they can more readily understand the relationship between their actions and changes in the visualization, and therefore the meaning of changes in the visualization for other people.

Erickson (2003) outlined six claims for designing visualizations of social activity, based partially on the design and experience of Babble, as follows:

(1) Everyone sees the same thing; no customization.
(2) Portray actions, not interpretations.
(3) Social visualizations should allow deception.
(4) Support micro/macro readings.
(5) Ambiguity is useful: suggest rather than inform.
(6) Use a third-person point of view.

While not all of these claims may be vital to facilitating sensemaking, some of them clearly are, including 1 and 6, as discussed above, and perhaps others. In the next example, we draw on the experiences of a long-running Babble community to look at how seeding the environment with suggested actions created a fertile ground for the emergence of new norms and practices among the users.

3 Appropriation Work in Babble: Seeding and Evolving Work Practices through "Social Scaffolding"

Netweavers is a community of interest at IBM consisting of globally distributed participants with an interest in communities, both "real-world" and online. The community has been active for several years as of this writing, and has a dedicated leader. In November, 2000, the leader approached the Social

Computing Group with a request to start a Babble as one of the ways Netweavers members could interact. The Babble has been active on and off ever since.

By the time the Netweavers Babble was started, we had already deployed and observed other Babbles and had developed a set of recommendations to help groups get started. We called these the "Six Habits of Effective Babblers," (see Figure 3) and gave them to the leaders of new Babbles (or posted them ourselves in the case of some Babbles in which we were also participants) along with some guidelines for how to get a successful installation off the ground. In the case of Netweavers, the leader was highly motivated and organized, and in addition our posting the "Six Habits," he posted three other topics: "Things To Do Right Away," (Figure 4) "Etiquette and Norms," (Figure 5) and "How to Make a Personal Place" (not shown). In addition, one of the developer/participants created a post called "Creating a Digital Culture," that shared some of the patterns that other Babble groups had discovered (Figure 6).

The "Six Habits of Effective Babblers" Post

1. TALK! Especially at the beginning, people aren't sure what to say. Be brave! Remember that something doesn't have to be of interest to the entire group to be posted in Babble. After a while you will find that there is benefit in listening in on other group member's conversation, even when they don't concern you directly.

2. BE RESPONSIVE. If someone writes something you like, say so. In Babble it's perfectly OK to write "I agree" or "Thanks" and nothing else.

3. BE SOCIABLE. Although Babble is intended as an environment for work groups, don't hesitate to be sociable. Say "good morning." Chat about the weather, or the headlines. Our experience is that talk breeds talk, and what begins as small talk often turns into work talk.

4. CREATE NEW TOPICS. Don't hesitate to create new topics.

5. BE EXPLICIT. If you'd like people to respond in a particular way, make that explicit: end with a question, or a request.

6. RESPECT THE GROUP'S PRIVACY. Treat Babble as a trusted space, where the group can talk freely and frankly with one another without fear of 'outsiders' overhearing. Thus, do not quote conversations that occur in Babble, either by pasting segments into email, or by verbally passing them on.

Figure 3. The Six Habits of Effective Babblers. Posted by one of the developer/participants in a new Babble community.

The "Things To Do Right Away" Post

1. Change your marble color
    step 1 - options menu
    step 2 - select marble color
2. Go to the Commons Area and say hello
    Step 1 - Click on Commons Area
    Step 2 - click on the area where you can read the text
    Step 3 - Start typing
    Step 4 - <shift+enter> to send your message
3. Comment on something that has already been written.
    Step 1 - highlight text on which you would like to comment
    Step 2 - start typing
    Step 3 - <shift+enter> to send the message
2. Create your Personal Place
    Step 1- click on Personal Places
    Step 2 -Topic menu/ New Topic
    Step 3 - enter your name
    Step 4 - Please tell us a bit about yourself.

Figure 4. Things To Do Right Away. Posted by the community leader, this message asked members who visited Babble for the first time to take some initial steps to become acquainted with the environment.

The "Etiquette and Norms" Post

- 1. When you arrive, please say hello & check in with the group. This just makes the place nicer for everyone else. This doesn't mean you have to stay and chat. But it's just nice to be real with the people around you. :-)
- 2. Please make sure that you read "Thing to do right away."
- 3. Please make sure that you have a name that everyone else knows or can link back to you. It's not okay to be anonymous here. Please...
- 4. Just jump into the conversation. Feel free. What are you working on? What is capturing your attention today? What would you like to see in the future? What's new? If someone could help you, how could they? What's the weather like outside? Are you going to be around today?
- 5. Again, please make a point of at least saying one or two things each time you visit. This could be as simple as "hello" or something about the weather.
- 6. Preferably, comment on something substantial that was mentioned earlier. Of course if nothing substantial was mentioned earlier, your comment could always be the first. ;-)
- 7. Please do not send the Babble application to people. Please do not tell people the port number for the Netweavers Babble.
- 8. Anyone who does work related to community is invited to join Netweavers. People get this information when they join.
- 9. Otherwise, if you enable someone to join Netweavers without them actually going through the registration process, this creates a lot of work tracking the person down and getting them registered. Having a process in place through which people join Netweavers and tell others about themselves is critical to keep some important characteristics of our social capital in place.

Figure 5. Etiquette and Norms. Created and posted by the community leader soon after the Babble had been deployed, except for items 7-9, which were added almost 9 months later.

The "Developing a Digital Culture" Post

As you use Babble over a longer period of time, you'll find that your group develops a set of customary ways of using Babble. Each group is different, but you may find it useful to try out some of the patterns that work for others.

THE COMMONS
This is the only usage pattern that is built into Babble. The Commons is intended to be a place where people hang out, and where a lot of casual talk occurs. To keep the amount of text in the Commons manageable, it is archived on a weekly basis.

GURU'S CORNER
In some Babble installations a topic of this ilk is inhabited by one or more members of the Babble design team, as a way of offering intelligent (hopefully!) online help.

OFFICES
Babble participants often create topics which serve as their online offices. People set the rules for their own offices, but typically an office is a place where others may leave messages (if the office owner is not online), and where an office owner may post drafts of work for comments, rough notes, their schedule, or do just about anything they wish!

BAD JOKES
Everyone seems to get a small trickle of jokes over the internet, and this is where they end up in our Babble.

ANNOUNCEMENTS, etc.
One or more topics devoted to upcoming events, announcements, interesting URLs, or other reference information can be quite useful. The Babble timeline window allows you to see whether people are actually making use of this information...

PROJECT X
Naturally, there are many topics which are oriented around a project...

Figure 6. Patterns of use in Babble, shared by one of the developer/participants in the Netweavers Babble.

The posts above were all created within couple of months of the Netweavers Babble deployment. What is striking, perhaps, is the amount of effort that the community leader and one of the Babble developer/participants exerted to seed an active, viable community. Our experience as the developers of Babble was that this kind of guidance, as well as leadership and commitment by a core group of users, was essential to establishing a successful Babble deployment. Another thing to note is the social nature of the scaffolding – in one case, the community leader making an explicit request of members new to Babble, and in the other, an experienced Babble user vouching for successful practices and encouraging others to explore and adapt them. The fact that these bits of encouragement or instruction surface in a conversational way reinforces that they are social requests, to be honored voluntarily.

The difficulty of establishing norms in online environments has been well noted in the literature. Danis & Lee (2005) provide a review in the course of reporting on their observations of the emergence of norms in an online chat environment used by summer interns. Notable points include that creating norms is difficult (Mark, 2002); that groups typically will only do the work of

developing norms with respect to things that really matter to them (Feldman, 1984), and that conflicting views of what appropriate behavior is can impair the social functioning of the group and its ability to establish norms. Jasperson et al. (1999) propose three 'social appropriation moves' (conformance, imitation, and mutual discovery) to account for social influence on individual decisions about IT use. In a similar vein, Mark (2002) attributes the failure of a distributed work group she studied to develop norms to several social factors mediated by awareness, such as being able to observe others' behavior, ability to monitor adherence to norms, and ability to apply peer pressure. We note that these are the kinds of awareness and accountability that socially translucent systems are meant to provide and support, respectively.

In the case of Netweavers, whether due to its skilled and organized leadership, to experienced participants leading the way, to a happy accident of a lively set of users, or some of all of the above, the social scaffolding was effective. This group, over time, evolved new Babble practices (albeit often with the participation of the community leader or an experienced participant), including an interview genre, a "best of Babble" topic that collected significant posts in one place, a question board for asking questions of the whole community, and a regularly-scheduled online chat for a leadership affinity group. Figure 7 shows some of the Babble dialogue that discussed the interview genre in a topic called "About Interviews." The discussion here shows that the interview concept is not yet well understood by the group, and normative practices (such as where the text will reside after the "live" interview is over) have not yet stabilized. Figure 8 shows the genesis of an idea for a question board initiated by one of the core participants (emphasis added).

The "About Interviews" Posts

Mark the Community Leader Monday 27Nov00 4:04:16 PM EST
Interviews are an experiment in ways that we can get to know one another.

Todd Friday 22Dec00 10:40:10 AM EST
As far as that goes, a couple of thoughts...FWIW.

I just read your interview (I think it's the only one done so far) with Tom, and I now understand what this process is meant to do. I signed up without really considering what it was I was signing up for - guess I thought it would have more to do with interviewing Babble users to get a sense of what they found useful/interesting, and unuseful/uninteresting about Babble - more researchy type of interview. (I believe this type of interview would still be worth conducting across our membership, from a research perspective).

In any case, as I read the interview with Tom I was reminded of my days as a reporter/journalist for the school newspapers (I did this in high school and college and loved it). As I thought about what those interview stories required to get them published, *it occurred to me that you may need to have Babble "reporters" identified who do this sort of work as more of a routine assignment, rather than on a hit and miss basis. Don't know whether I'm making sense or not, but that's just a thought.*

Mark the Community Leader Tuesday 26Dec00 9:00:19 AM EST
Excellent idea, Todd. It would be great if we reached a point where there was someone who was willing to take the lead around doing these interviews on a more regular and long-term basis. (I'd do it myself if I wasn't already doing so many other things here).

So, you understand that this has nothing to do with being a Babble user? It's mostly about learning more about each other. This is about the people inside of the company who do community-related work.

The idea started when Tom and I had a discussion here (see archives) about where we can start to address some of the issues. He felt that we don't yet know each other enough and that it would help to have some activity where we could introduce ourselves a bit more.

Mark the Community Leader Tuesday 14Aug01 7:03:58 AM EST
I wonder about interviews... here we are, finishing up Carlos' interview and we can still develop the questions as if we are the interviewers... I wonder about the possibility of the interview being integrated into a person's personal place? Would it make sense to relocate an interview there and continue the dialog with the interviewees?

Mark the Community Leader Tuesday 14Aug01 7:04:12 AM EST
I'm noting the similarity between interviews and personal places.

Carlos Tuesday 14Aug01 8:26:59 AM EST
Responding to: <<the similarity between interviews and personal places>>

And I agree with it, actually, more than anything else because it is OUR OWN interview and as such it should be in OUR OWN place... By the way, when is the next interview taking place?

Figure 7. A community clarifying what is and what is not an interview in a collective experiment to get to know each other better by conducting "live" interviews (online) in front of a "live" (online) audience.

The Genesis of the "Question Board" Idea

Lloyd @ London Thursday 1Feb01 6:22:03 AM EST
Hey people - had an idea. Could we use this place to ask questions to anyone in the community..

Here's one to start:
We have a customer coming who requires consultation advice about how to set up knowledge communities in their organisation. Would any of you be prepared to fly over here to meet this customer? (obviously we'll cover costs + provide more info if you are interested) ???

Mark the Community Leader Thursday 1Feb01 9:53:31 AM EST
Great idea, Lloyd. I was wondering if there might be a place where we could start engaging in some of the work around "building" community. Thinking that a construction zone, or something like that would be a good metaphor. There, we could share methods, info re: tools access, strategy, etc.

Scott Thursday 1Feb01 10:16:27 AM EST
Hi Lloyd - yes, it will be interesting to see what shows up here.

Figure 8. A core participant suggests the need for a new topic, and demonstrates what he has in mind with a particular request of the community. The community leader reinforces and extends the idea, and another participant seconds the idea.

4 Large-Scale Sensemaking: IBM Community Tools' Pollcast

The last example is taken from a suite of "broadcast messaging" applications that run within IBM, called IBM Community Tools (ICT). ICT has about 50,000 subscribers globally across the company, from many different organizations and parts of the business. One application, called "Pollcast," allows subscribers to compose a multiple-choice question and send it out to one of the communities defined within ICT. The poll is received by any members of the target community who are online, and within a minute or two, responses come back and are displayed in the sender's pollcast window. An example of this is shown in Figure 9.

Two types of sensemaking with respect to Pollcast are interesting from the perspective of the ability of large communities of people to self-regulate. In the initial deployment of Pollcast, a particular pattern of abuse emerged. For example, someone might send out a poll asking "Which do you like best: Pepsi or Coke?" A minute later, someone else would follow this with a poll asking "Do you think

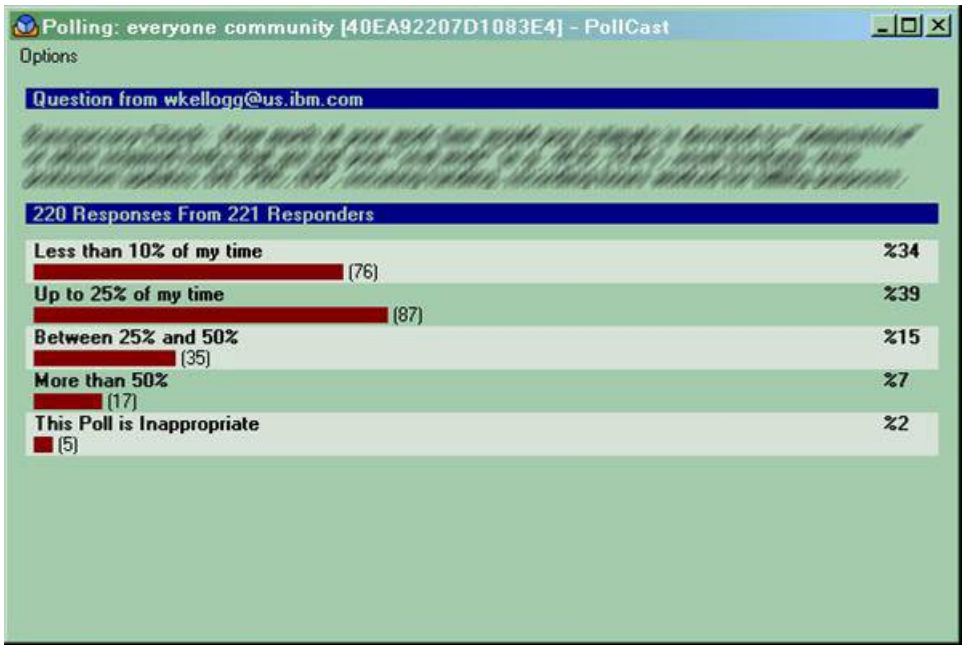


Figure 9. A poll sent to the "everyone" community.

it's appropriate to use Pollcast to ask about soft drink preferences?" and so on, leading to a kind of trivia flame fest that in practice could ruin the broadcast capability for everyone. The developers responded to this pattern by disrupting it: they made it mandatory for "This poll is inappropriate" to be added as a response to every poll that was generated (see the last response option in Figure 9).

This was a relatively small interface change, but it had the intended effect of reducing the number of inappropriate polls. It is interesting to speculate why. First, the change made inappropriateness more visible, public and at the same time less disruptive to the community. A poll sender (as well as anyone who responded to the poll) could see in the poll results how many people were annoyed or thought it inappropriate *without* the disruption occasioned by "follow up" polls. Second, making it easier to 'see' the degree of inappropriateness clearly opened up the possibility of either policy-based regulation (e.g., "people having more than X inappropriate responses will lose their privileges"), or worse, termination of a useful service if there was too much abuse. The latter possibilities were only ambiguously and subtly represented by the interface change, but were definitely not lost on this population of corporate users.

Experiments in real-time broadcast messaging are new in enterprise environments, and they raise many concerns about whether the benefit gained is worth the potential disruption to a large number of coworkers. This brings us to the second type of sensemaking in Pollcast, which has to do with choosing an appropriate community to which to send your message. Most ICT users send their messages to the "everyone" community, because it is the largest community, and the possible alternative choices are hard to understand, consisting of a list of

hundreds of communities without any information beyond their (sometimes cryptic) names. Unfortunately, using "everyone" when a more targeted community would do exacerbates the "tragedy of the commons" problem that pits an individual's interest in getting an answer against the community's interest in not being excessively pinged. During the summer of 2004, we studied this issue (in part by examining log files to ascertain the level of activity and types of questions in each community and in part by interviewing ICT users and developers) and designed a prototype to give users more information about the activity level, participants, and typical questions of various ICT communities. By making such social information visible, we hoped to better address the tradeoff between getting a swift answer and imposing unnecessarily on the larger community. At least one aspect of the prototype has been implemented as of this writing: ICT communities now show the number of subscribers currently logged on to ICT, giving questioners some idea of the size of the audience to which they are broadcasting (and thus some way to estimate the likelihood of a response).

5 Social Translucence and Appropriation

From the dawn of modern computing with "end users" getting their (situated) hands on technology and applications, technology has been "appropriated" and assimilated into activities and work practices. Designers and developers of technology can't prevent appropriation, nor should they seek to, but they can attempt to make it easier and the possibilities richer – both by creating intentionally malleable software, and, as discussed here, by designing socially translucent software. By creating support online for basic, everyday human activities – sensing who is around and with whom you are interacting, being able to observe others' behavior, knowing what others can observe about your own behavior, and being able to converse with others under well-understood circumstances – technologists create optimal conditions for appropriation work. Collective sensemaking and social scaffolding are central to the essential appropriation work of understanding the capabilities of a technology, and negotiating with others as it is adapted for use in particular circumstances. We believe that social translucence – providing perceptible social cues that lead to mutual awareness and accountability – can support these key activities in appropriation work.

6 Acknowledgments

Thanks to the members of the Social Computing Group for their inspiration and collaboration.

7 References

Ackerman, M., Halverson, C., Erickson, T., and Kellogg, W.A. (in preparation). Resources, co-evolution, and artifacts: Theory in CSCW.

Danis, C. and Lee, A. (2005). Evolution of norms in a newly forming group. In *Proceedings of INTERACT 2005*; Rome, Italy.

Erickson, T. (2003). Designing visualizations of social activity: Six claims. In *Human Factors in Computing Systems: Extended Abstracts*. New York: ACM Press.

Erickson, T., Halverson, C., Kellogg, W.A., Laff, M., and Wolf, T. (2002). Social translucence: Designing social infrastructures that make collective activity visible. *Communications of the ACM* (special issue on Community, ed. J. Preece), **45(4)**, pp. 40-44.

Erickson, T., Smith, D.N., Kellogg, W.A., Laff, M.R., Richards, J.T., and Bradner, E. (1999). Socially translucent systems: Social proxies, persistent conversation, and the design of "Babble." In *Human Factors in Computing Systems: The Proceedings of CHI'99*.

Eriksén, S. (2002). Designing for accountability. In Proceedings of the second Nordic conference on Human-Computer Interaction, NordiCHI 2002.

Feldman, D.C. (1984). The development and envorcement of group norms. In Academy of Management Review, **9(1)**, pp. 47-53.

Jasperson, J., Sambamurthy, V., and Zmud, R.W. (1999). Social influence and individual IT user: Unraveling the pathways of appropriation moves. In *Proceedings of the 20th International Conference on Information Systems*. Charlotte, North Carolina.

Mark, G. (2002). Conventions and commitments in distributed groups. Computer Supported Cooperative Work, **11(3-4),** pp. 349-387.

Leveraging the language action perspective for system accountability and end user configurability

Gianni Jacucci,
Diego Calzà,
Vincenzo D'Andrea
Dept. Sociology and Social Research University of Trento, Italy
gianni@lii.unitn.it

Arthur B. Baskin
IIT, Indianapolis, Indiana (USA)

Ina Wagner
Technical university of Wien

Abstract. Computer supported collaborative work practices could afford substantial improvements in terms of user appropriation. System accountability and user-configurability in particular could be enhanced leveraging the Language Action Perspective on Communication Modelling (LAP). This position paper provides a brief outline of our research agenda.

1 Introduction

We are hopefully at a turning point in the evolution of the Information Systems (IS) field, and of computer supported collaborative work practices [1]. User appropriation of IT appears to be methodologically within reach. In particular, design for accountability and design for emergent use appear to be within reach,

because of recent work leveraging on the Language Action Perspective, presented at the series of conferences called ALOIS (Action in Language, Organisations and Information Systems) and LAP (Language Action Perspective) [2].

Object of the present communication is a research agenda, encompassing information systems and organisations, based on the social study of information and communication technologies, and specifically on LAP, featuring a methodology based on:

i) e-Negotiation [16] to provide end-users, or their representatives, with expert support for:

a) the identification of system viable end-user tailoring moves, and

b) requirements validation to allow the end-user tailorability to be checked against eventual governing rules and laws in the domain. In that way, the users, or their representatives, could make changes but only within the allowed variations.

ii) ISAT (IS Actability Theory) [3] and Thematic Roles [4] for implementing the use of Use Cases in design for accountability, and in design for end user design in use (DEUDU, [5]), attempting to exploit thematic-role derived work-graph representations.

Let's start with a word of caution on scope, and of explanation on terminology. In this attempt, we would like to address issues related to design and co-evolution of human work practices and information infrastructure support, resulting in the complex dance of human and machine agencies in organisations [6].

Furthermore, in this paper, the concept of Use Case coincides with the original, traditional definition of a natural language description of the succession of actions and actors in activities involving computer support of human work.

In the following, the key issue is human *interpretation*, analysed with the thematic role theory [4] linking *representations* (used by humans and machines) to *actions* (of both humans and machines). A key concept will be that of the *thematic role derived work graph representation* [4], a representation of work activities which is intended to be understandable at the same time both by users and by machines, for the purpose of enabling their collaboration in many ways, in the tailoring and evolution of the work graph itself, and of the Use Case it represents.

In 2002 Jacucci et al [5] proposed the use of Use Cases in Design for End User Design in Use (DEUDU). No methodology however was given at the time for carrying out that proposal. In this paper, we advocate the development of a methodology, based on the Language Action Perspective, on e-Negotiation, and on the Thematic Roles Theory, for analysing interpretations linking representations to actions in the natural language Use Case description. The methodology should enable the use of use cases both in design for accountability, and in design for end user design in use. PD is an essential approach to IT design. LAP has been explicitly espoused with PD over a decade ago [9]. We take explicit

account of this merging in our Use Case development and we profit from it in our approach.

2 A research agenda to put the new perspective to work

In order to be successfully deployed and implemented, the technological power of ICTs needs support and respect to humans and organisations, and care taking of their needs. The social study of information and communication technology (see the book recently promoted by Claudio Ciborra at the London School of Economics [7], and his own previous book [8]) has motivated the tenets of this proposed research agenda. Participation – of users and stakeholders – to the design of IT use, is one of the most prominent needs of humans and organisations [9]

The outcome of a Participatory Design (PD) approach are a number of design imperatives towards design for change, design for configurability, and design for emergent use (ref. my DEUDU); in particular:

- design for accountability [10, 11]
- design for end user design in use (DEUDU, [5])

Improved human machine cooperation can be further promoted by three main paths enhancing possibility of establishing a web of shared understanding and co-operation between humans and computers:

- action in language and organisation for information systems [2]
- interaction design and tangible computing [10]
- double dance of human and machine agencies [6]

Let us zoom on accountability (the system capacity of giving account of itself, provide sense making to users). System users do not know what systems are for, nor how they should be operated [11]. We should care for developing accountability of IT instruments to humans in knowledge communities. Systems should display "business" logics: system displays should emphasize system action aspects, in terms of knowing how a system works, its "business" logics, rather than just its operations [4].

Let us zoom on DEUDU, and the adaptability to situation. We should beware of limitations of planned/allowed use-scenarios. We should allow change in situation: introduce DEUDU (Design for End User Design in Use) as system adaptability by user intervention. Not easy: it requires brokering the needs of humans for 'gestalt' and the need of machines for hierarchy. Analysis on the problem situation recalls a foundational book indicating/advocating the new perspective to replace the rationalistic tradition. It is entitled *"UNDERSTANDING COMPUTERS AND COGNITION"(A new Foundation for Design),* authored by Terry Winograd & Fernando Flores, Addison – Wesley 1986 [12].

This book questions the assumptions of the rationalistic tradition about the objectivity of our representations of the world. This tradition provides us with a rationalistic perspective that serves as a basis for our culture's commonsense understanding of language, thought, and rationality

3 Looking for tools from the *Language Action Perspective*

User satisfactory IT use requires IT accountability and IT configurability in use.
A seminal paper of Goldkuhl and Lyytinen of 1982 [17] opens the way of LAP (the Language Action Perspective) . In the LAP on IT use, developed for supporting social interaction and work, IT infrastructures *is* language. For linguists, language has two main dimensions: semantics (propositional) and pragmatics (illocutory) [12]. As a consequence, also IT use must have two dimensions, semantics and pragmatics. This is in fact a user's tacit assumption. If only one dimension is carefully designed for use and satisfactorily implemented, while the other dimension is not taken into account and elaborated, then user frustration in use ensues immediately.

How do we proceed to address IS accountability and DEUDU remembering that ISs are language?

Zooming on language and speech act theory permits to identify two communicative functions:

- propository (semantics: content, meaning)
- illocutory (pragmatics: intentions, commitments)

This allows a new methodology for IS design. If ISs are language, then we should address need for both kinds of communicative acts. Par Ågerfalk [3] has provided us with a very nice example, from the analysis of a system assisting the handling of college syllabus.

This section lists telegraphically highlights of recent advances in the LAP area that can help solve the problems of our research agenda.

Closes this section the reference to a proposal of a representation language for actions based on interpreting language actions with thematic roles theory [4].

Communicative Aspects of IT-Usage [12]

In the field of IT-design the prevailing language perspective is a referential one. The most fundamental activities of system design are seen as the mapping of a universe of discourse into abstract symbolic models and databases.

But the "descriptive fallacy" of methods and techniques for IT-design has been attacked, and a new set of methods, techniques and software artefacts has now evolved that may be seen as a kind of "communication paradigm", in the way Winograd and Flores argued for a "new foundation of design".

This new orientation in iT-design is directed towards the development of computer software for organisational communication and action. Organisations are viewed as networks of commitments and undertakings.

A communicative or language oriented view of IT-design may be rewarding: a large part of work is performed through language, and IT is used to support communicative activities to a considerable extent.

Conversation for Action [Flores and Winograd, 1988 13]

Speech act theory as a foundation for design has produced a generic schema of conversation for action, that has widely influenced the area of Workflow Management, CSCW, and BPR.

A conversation is a coordinated, coherent sequence of language acts.

At each point in the conversation, there is only a small set of possible action types.

A discourse may be defined in a state transition diagram, where each state-transition corresponds to a speech act.

State-transition diagram from a workflow

For each task there is a workflow, which includes communication with the customer, according to the state transition diagram for the workflow of that task.

The basic workflow loop has four phases, through the fulfilment of commitments by a performer to the satisfaction of a customer.

According to this view, any work activity can be sequenced in four basic steps:

- *preparation*: the customer makes a request, or the supplier makes an offer;
- *negotiation*: the parties establish a mutual agreement of conditions of satisfaction;
- *performance*: the supplier declares that the undertaking is complete;
- *acceptance*: the customer declares satisfaction.

Several circles of can be interconnected with links, such that a speech act in one workflow may trigger one in another workflow. In this way, one workflow can be viewed as a sub flow to another workflow.

The basic workflow loop is used as a means to articulate customer-supplier relations, with customer satisfaction in focus. There is always an identified customer and a performer, with the loop representing a particular action the performer agrees to complete to the satisfaction of the customer.

Using the ALOIS related work of Ågerfalk, Goldkuhl, Andersen.

We propose to use the conversation-for-action schema, as enriched by Kensing and Winograd, and by Andersen, to design the communicative aspects of IT-usage in the end-user-tailoring of IT, accepting to be affected by all limitations elicited in *Speech Acts on Trial (Ljungberg and Holm, 1996 [14])*, except for the additional flexibility provided by tailoring. For IT-design, we propose – within

the same limitations - to marry the language action approach to PD, as already done by Kensing and Winograd. Note that Kensing and Winograd had already coped with unanticipated breakdowns by combining specialised and more general conversation patterns in a uniform framework.

As a methodology implement, for DEUDU (Jacucci et al., 2002 [5]) and end-user tailoring for flexibility in emergent use, we intend to use the ALOIS related work of Ågerfalk, Goldkuhl, Andersen. For providing end-users, or their representatives, with expert support for:

(1) the identification of system viable end-user tailoring moves, and
(2) requirements validation to allow the end-user tailorability to be checked against eventual governing rules and laws in the domain. In that way, the users, or their representatives, could make changes but only within the allowed variations,

e-Negotiation: A Language-Action Approach to Electronic Contracts

We intend to use the work of Schoop and Jertila on e-Negotiation [16]: The Language-Action Perspective and the Semantic Web – A Language-Action Approach to Electronic Contracts, recently presented LAP 05, June 19-20 2005, Kiruna, Sweden.

e-Negotiation is communication-intensive. In order to enable electronic negotiations, the complex communicative exchanges need to be supported by means of information technology. The Language-Action Perspective can provide a suitable theoretical and conceptual basis. In addition to the communicative exchanges, document management also plays a vital role for e-negotiations. Semantic web ideas can be most useful for this part of a negotiation. Ref [16] presents an integrated approach implemented in the negotiation support system Negoisst that combines LAP and Semantic Web and enables the support of highly dynamic complex electronic negotiations in a business-to-business environment.

We propose to apply e-Negotiation as developed in [16] to provide end-users, or their representatives, with expert support for:

a) the identification of system viable end-user tailoring moves, and
b) requirements validation to allow the end-user tailorability to be checked against eventual governing rules and laws in the domain. In that way, the users, or their representatives, could make changes but only within allowed variations.

We would now be ready to come down to cases, and try to solve specific domain problems (Distributed Collaborative Engineering, IT for Tourism, eGov, Health Care, HRM, etc.). Objective: Solve IT accountability and end user configurability in use, with the help of the methodology. In particular we need to:

- Indicate how humans and machines would be able to interpret each other.
- Indicate how design expert can come in picture to support users in tailoring.

- Indicate how upper software layers would be built in the OOAD framework.
- Indicate how humans would work with the IT infrastructure.
- Indicate how the IT infrastructure would function.

In an application example, one should:

- Discuss aspects of accountability and end user configurability.
- Make example of applications of e-Negotiation.
- Make examples of application of Actability Theory of IS.
- Make examples of application of Thematic Role Theory.

4 Conclusions

In general, communication and the establishment of a web of shared understanding between humans and machines, can be better achieved following the LAP perspective and its recent developments described above. e-Negotiation with design experts can support users in understanding IT infrastructures in use and identifying their allowed and useful configuration tailoring options.

Besides, design for accountability could be further pursued by analysing and identifying human interpretations of Use Case sentences with thematic role theory, and exposing them explicitly in the user-machine interaction by drafting and displaying the thematic role derived work graph diagrams.

Design for end user design in use could be pursued by analysing and identifying human interpretations of Use Case sentences with thematic role theory, and exposing them explicitly in the user-machine interaction by drafting and displaying the thematic role derived work graph diagrams, where are also displayed both a) alternative routes to work performance, and b) tailoring opportunities of the work graph exhibited in the work graph diagrams by appropriate graph modification controls.

5 Acknowledgments

Numerous useful discussions with Par Ågerfalk, Peter Bögh Andersen, Goran Goldkuhl, Finn Kensing are gratefully acknowledged.

6 References

[1] Baskin, Kovacs, and Jacucci, eds., 1999, Cooperative Knowledge Processing for Engineering Design, Kluwer.

[2] ALOIS 2004, “Action in Language, Organisations and Information Systems”, Seminar in Linkoping, Sweden, http://www.vits.org/konferenser/alois2004/proceedings.asp ; LAP 2005, The Language Action Perspective on Communication Modelling, http://www.vits.org/?pageId=238&UltimateMenu1B=Item12

[3] Sjöström J. and Ågerfalk P. (2004) Analysis of communicative features of user interfaces. In ALOIS Workshop: Action in Language, Organisations and Information Systems, ed. G. Goldkuhl, M. Lind and S. Cronholm. Linköping, Sweden: Linköping University.

[4] Andersen P.B. (2004), Analysing and Diagramming Complex Heterogeneous Activities, In ALOIS Workshop, ibid.

[5] Jacucci G., Calzà D., and Dandrea V. (2002), Use of Use Cases in Design for End User Design in Use, Report, Department of Sociology and Social Research, University of Trento

[6] Rose, J. and Jones M. (2004). The double dance of agency: A socio-theoretic account of how machines and humans interact. In ALOIS Workshop, ibid.

[7] Chrisanthi A., Ciborra, C., and Land F., 2004, The Social Study of Information and Communication Technology: Innovation, Actors and Contexts. Oxford: Oxford UP.

[8] Ciborra, C., 2002, The Labyrinths of Information: Challenging the Wisdom of Systems. Cambridge: Oxford UP. [9]

[9] Kensing F., and Winograd T. (1991), "The language-action approach to design of computer support for cooperative work," Proceedings of the IFIP TC8 Conference on Collaborative Work, Social Communications and Information Systems, Helsinki, Finland

[10] Dourish, P., 2001, Where the Action Is: The Foundations of Embodied Interaction. Cambridge MIT Pr.

[11] Eriksen S., 2002, Designing for Accountability, in Berthelsen O., Bodker S:, and Kuutti K., eds., NordiCHI 2002

[12] Winograd T. and Flores F. (1987), Understanding Computers and Cognition: A New Foundation for Design, (220 pp.) Norwood, NJ: Ablex, 1986. Addison-Wesley.

[13] Flores, Fernando, Michael Graves, Bradley Hartfield and Terry Winograd (1988), "Computer systems and the design of organizational interaction," ACM Transactions on Office Information Systems 6:2 (April, 1988), pp. 153-172.

[14] Ljungberg, J. and P. Holm (1996) Speech Acts on Trial, Scandinavian Journal of Information Systems, Vol. 8, nr. 1.

[15] Sowa J. E. , 2000, Knowledge Representation: Logical, Philosophical, and Computational Foundations Pacific Grove, CA: Brooks/Cole

[16] Jertila A. and Schoop M., 2005, The Language-Action Perspective and the Semantic Web –
A Language-Action Approach to Electronic Contracts, LAP 2005, Kiruna, Sweden; http://www.vits.org/konferenser/lap2005/Paper%2012-LAP.pdf

[17] Goldkuhl G. and Lyytinen K., 1982, A language action view of information systems, Proceedings of the 3rd international conference on information systems, TIMS/SMIS/ACM

What's in a name? Exploring the connections between abstraction and appropriation

M. Cameron Jones,
Michael B. Twidale
Graduate School of Library and Information Science
University of Illinois at Urbana-Champaign
mjones2@uiuc.edu, twidale@uiuc.edu

Abstract. In this position paper, we discuss the role of abstraction in designing for appropriation. We examine the ways in which varying the level of abstraction of tools affects the ability of users to appropriate them. We close with some words about the difficulties of evaluating the appropriability of systems and how they might be addressed in an experimental framework.

1 Introduction

A focus of computer systems design research has been building systems which are capable of being appropriated by users. In order to build systems that explicitly support appropriation, the factors which affect appropriability must first be identified. One such factor is the manner in which software tools are described. This is especially true in component-based software development (CBSD) environments which are designed to enable end users to combine components in order meet their particular needs. These systems often are extremely flexible and powerful. They combine all three aspects of tailoring as described by Mørch (1997); namely customization, integration and extension. Users can configure customizable options for each component, arrange and rearrange components in

any number of combinations to create new compound functionalities, and even generate new components through coding or sharing templates.

Based on preliminary observations of a community learning toolkit, it would appear that tailorable tools, when abstractly described to encompass maximal flexibility and customizability, are less appropriable than when provided as a narrower, particular instance. This is a significant problem for researchers and developers of tailorable technologies, as it contradicts the practice of offering the most flexibility and the greatest customizability to the user for the sake of allowing them to fashion whatever they would like.

2 The Story of the Timeline Tool: Anti-Affording Appropriation

The ILABS system, short for Community Inquiry Labs, is a framework and a set of tools, called bricks, for supporting online communities of inquiry (Bishop, et. al, 2004). The aim of the system is to enable participants to put together a customized environment that will support learning and knowledge sharing for a particular community of users. Examples of the diversity of communities using ILABS include university courses, a Puerto Rican community library project, an African-American women's health network, and a multi-disciplinary research initiative.

One of the first bricks developed for the ILABS project was a timeline tool. The original timeline brick was built around the needs of a professor for one of his classes. The Learning Technologies Timeline existed as a static HTML page, maintained by the professor, which students researched and contributed items to. The initial timeline brick was built specifically to address data of this form, helping the professor update and manage the timeline. The resulting tool had input fields for a date, event, URL and description; the data was sorted in ascending, chronological order.

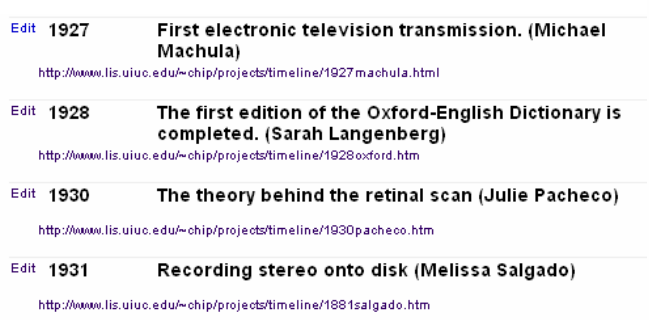


Figure 1

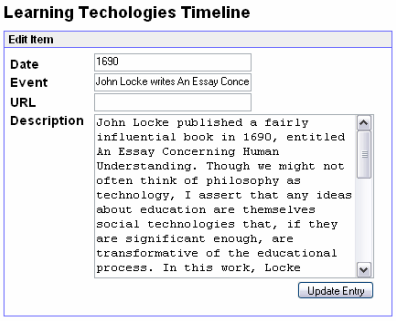


Figure 2

Figure 1 shows a view of the Timeline tool, almost identical to the original HTML based Learning Technologies Timeline. Figure 2 shows the interface for adding and editing items to the Timeline. Note the specific names of the fields.

Although developed for one professor, the timeline brick was used by other groups. This straight forward tool was easy for users to comprehend – the structure and name of the tool accurately reflected its purpose. The concept of a "Timeline" was familiar and did not need much documentation to explain its purpose and use. Shortly after introducing the tool, we noticed that many people were using the timeline brick for other purposes such as organizing upcoming events, email addresses, daily schedules, and other tasks that involved creating an ordered list of items. We consider these unintended uses to be appropriations of the software – allowing the users to accomplish their goals by creative use of the technology available to them. That is, by looking at examples of timelines created with it, not only were they able to develop their own timelines but also imagine using it for other purposes.

In light of these new uses, we redesigned the tool to be a more generalized key-sorted data table. The new tool was more customizable, allowing users to specify the number, name and data type of the fields, and change the sorting behavior. In order to communicate the breadth of this tool's functionality we renamed it the "Sorted List" to emphasize that it was now a tool from which any number of types of lists could be created, of which timelines were just one example.

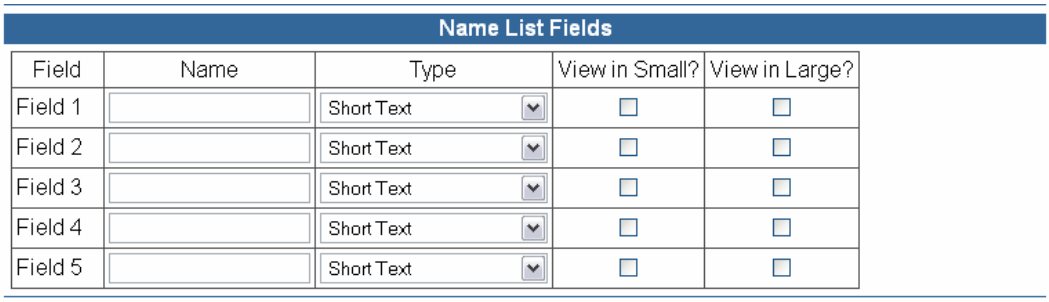


Figure 3: A screenshot of the new Sorted List creation tool, showing the numerous configuration options. Users can define a name, data-type and display options for each field. On subsequent steps of the configuration wizard, users can define sorting behaviours. This very powerful, abstract tool actually led to less appropriation.

Unfortunately we noticed in informal observation and feedback from users that most users did not know what the new, improved Sorted List tool was or why they would want to use it. The Sorted List was described as a tool which enabled users "to create different kinds of sorted lists as content for your iLab". Even when the description was modified to include examples (the addition of the phrase "like bookmarks, contact lists, blogs, etc." to the previous description), users still had troubles understanding what the tool was for. This was very frustrating to us as developers of this tool. We had initially developed a tool that happened to lead to informal spontaneous appropriation. Noticing and valuing that phenomenon, we had put substantial effort into refactoring the design, and created an abstracted

functionality far more powerful, adaptable and tailorable than the original, but had ended up with something that was used less and hence adapted and tailored less. Why was this? We suspect that in the process of adding power by abstraction, what was lost was an understandable model which users could grasp; an existing context where they could observe what the different configuration options were for.

To address this paradox we incorporated specific configurations of the list tool as starting points from which users could adapt to their particular needs. The configurations currently supported includes: the timeline, task-list, address book, bookmarks, glossary and blog. These configurations are not merely default settings for the various configuration options; they each represent a distinct conceptual purpose or use-instance of the tool. Not only can users get by with making fewer reconfigurations, but the cognitive overhead of adapting a particular list is less than that of instantiating the more abstract data-type of Sorted List.

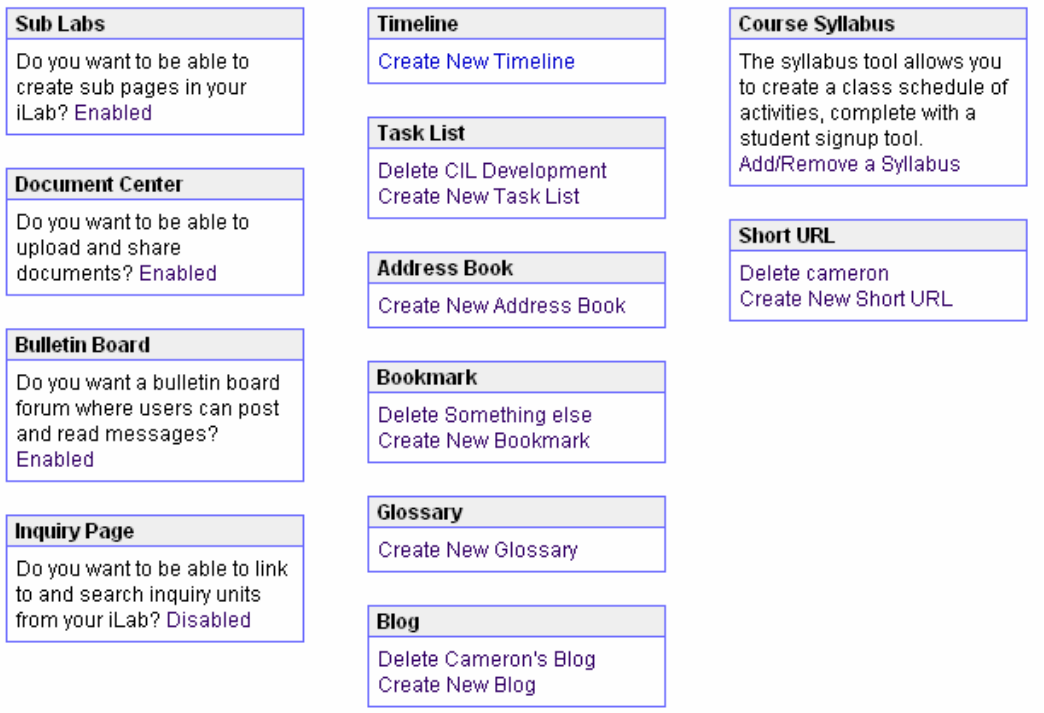


Figure 4: A screenshot of the revised iLabs system showing six different configurations of the Sorted List tool including the timeline, presented in the middle column.

From a design perspective this redesign seems a retrograde step – a hack-like inelegant duplication of work, unnecessarily multiplying the number of options that the user has to decide to choose from. From the perspective of a computer scientist no new power has been added and the elegance of the previous Sorted List tool has been corrupted. And yet we have some slight evidence from subsequent use that this change is at least better than our first redesign. The best-

practices advocated in object-oriented design lead programmers to create powerful abstractions to maximize modularity, extensibility and reusability while minimizing redundancy (Alfonseca, 1990). These principals do not seem to enable end users to appropriate and reconfigure in the way that programmers are meant to search through and select from abstract classes and superclasses in Smalltalk. Put this way, it does not sound so surprising, but that is our concern–these design precepts are rarely articulated in sufficient detail for a clear critique to be applied to their suitability for end user appropriation.

3 Using versus Programming

Object-oriented languages like Java and Smalltalk offer programmers vast libraries of classes and class hierarchies from which to select, extend and use in their applications. These libraries are similar to the sets of tools and components made available to users of CBSD systems in that they contain a large amount of pre-defined functionality from which a user must make informed selections. The process by which a programmer chooses a class or module, however, is very different from that of a user. Where a user is driven primarily by picking a component which will do what he or she wants, a programmer must consider other aspects such as memory usage, speed, and flexibility in addition to functionality.

The naming of components in systems designed for end users needs to reflect the users' frame of reference. The classes and modules in programming libraries are named and described by and for programmers. However, adopting the manner of description employed by computer programmers to the description of end user tools can be highly problematic. As evidence, we present some observations from the DATA TO KNOWLEDGE (D2K) toolkit developed at the National Center for Supercomputing Applications (NCSA). The D2K toolkit is a data-mining framework, allowing users to construct data flows from modules. D2K modules have inputs and output; they are assembled in a graphical drag-and-drop interface, and connected by data pipes thus specifying the flow of data through the system. Data is then loaded into the front end of the dataflow and a sequence of operations is executed on the data as it passes from one module to the next. The toolkit serves as both an application for end-users to mine data using the modules provided and a platform for developers to build new algorithms by leveraging an existing code base.

Problems arise when the toolkit is used by non-programmers (who are usually experts in the data they wish to model but have no formal programming experience). The graphical interface is designed to make it possible for these users to apply advanced data-mining software easily. The problem is that the interface is just a graphical representation of the underlying programmers' application programming interface (API). The API, intended for programmers, contains

language and descriptions which are targeted at programmers; designed to help them make informed decisions about which modules to use and extend in their application development. Users cannot easily navigate the collection of modules and find ones they might want because they are organized, named and described as they would make sense to a programmer. For example, the machine-learning classifier modules are in different hierarchies for the different authoring organizations (e.g. WEKA vs. NCSA classifiers). This mismatch is indicative of what we feel is a fundamental difference, that abstraction in computer programming languages is different from the kind of abstraction that is helpful to users.

Similarly, the naming and description of the Sorted List brick in ILABS changed with its development. From the perspective of the developer, there was a logical progression; as the tool got more powerful it got more abstract – both conceptually and in name. However, from the perspective of the user, as the tool became more abstract, it became harder to envisage *any* of its different intended uses.

4 Problems and future work

What needs to follow is a more formal and rigorous study of this phenomenon. Much of our observations have been informally gathered through interactions with users and usage log analysis. However, formal comparisons of the appropriability of the various instances of the timeline tool could help in understanding the phenomenon better. We are currently planning user studies designed to measure how significant the abstraction and naming can be in appropriation. A major hurdle is determining how to measure appropriation and appropriability. Even determining an operational definition of appropriation, its different forms and the features that afford it needs more work. We have begun enumerating some of the aspects of technology and software which we feel afford appropriability and might lead to evaluation metrics; this list is by no means complete or necessarily accurate, but has been included as a starting point for further discussion and perhaps to serve as a guidelines for evaluation.

- **At-Handness**: At-hand tools are those which are both physically and cognitively available to the user. At-handness is more than physical availability, because tools may contain features or functionality the user does not know about and thus cannot appropriate. At the same time, there might be a problem with ubiquitous things 'being hidden in plain sight'. Ciborra (1996) describes a similar concept asserting that appropriation happens when a user becomes "intimately familiar with an innovation", ultimately allowing that user to be able to call upon the technology to support day-to-day activities.

- **Granularity**: Clay offers high levels of precision and control in sculpture, but takes a lot of knowledge and skill to mold well. LEGO blocks are a more coarse-grained design resource which can be assembled with lesser skill to produce rough sculptures which approximate the smooth curves attainable with clay. LEGO construction can also be codified precisely such that the exact same sculpture can be reproduced whereas no two clay sculptures can be precisely the same.
- **Playfulness**: The degree to which a tool supports and encourages users to 'play-around', testing variant configurations and learning about how the tool functions. More playful systems could lead to greater discovery of features or generate more ideas about how to use the system in novel ways. This idea is connected more with the notion of serendipitous appropriation as opposed to what we've primarily been discussing – goal-oriented appropriation. We consider serendipitous appropriation to be the uses which arise out of spontaneous creativity – a moment when a user realizes that the tool they have could be used to do something else. This is unlike the goal-oriented appropriation, where a user finds a technology which can help him or her satisfy a need or aid in attaining a specific, defined goal.
- **Sharability**: The degree to which the tool supports sharing customizations and modifications. Tools that have higher sharability would allow users to share appropriations and learn from each other (e.g. Nardi and Miller, 1991).
- **Simplicity**: Tools with complex interfaces might be too difficult to integrate (in the words of Mørch, 1997). Simple things might just be easier for users to understand and learn - thus increasing the at-handness. Also, simple tools might have some atomicity in their functionality, allowing them to be appropriated into ad hoc workflows more easily.

We believe that a consideration of the features that enable appropriation can lead us to the specification of requirements for technologies that can explicitly support appropriation activities by users.

5 References

Alfonseca, M. (1990). Object-Oriented Programming, Tutorial. Conference Proceedings on APL 90: For the Future. May 1990.

Bishop, A. P., Bruce, B. C., Lundsford, K. J., Jones, M. C., Nazarova, M., Linderman, D., Won, M., Heidorn, P. B., Ramprakash, R., Brock, A. (2004) Supporting Community Inquiry with Digital Resources. Journal of Digital Information. 5(3).

Ciborra, C. U. (1996). Introduction: What does Groupware Mean for the Organizations Hosting it? In Ciborra, C. U. (ed.) Groupware and Teamwork: Invisible Aid or Technical Hindrance. Wiley Series in Information Systems. 1-19.

Mørch, A. (1997). Three Levels of End-user Tailoring: Customization, Integration, and Extension. In M. Kyng and L. Mathiassen, editors, Computers and Design in Context. The MIT Press, Cambridge MA.

Nardi, B.A. & Miller, J.R. (1991). Twinkling lights and nested loops: Distributed problem solving and spreadsheet development. International Journal of Man-Machine Studies, 34(2), 161-184

Design for appropriation of ubiquity in information systems

Riad Lemhachheche
Department of Industrial Engineering
Oregon State University
riad@lifetime.oregonstate.edu

Abstract. Computing and information systems have started to move out of the desktop and into the environment. Design requirements are no longer limited to the classical computer-user interface. Interactions are now expected to occur in a wider environment and in an invisible and more natural form. A large amount of research in various field of human – computer interaction has sought to address this evolution. Our research is aimed at helping this evolution by providing a comprehensive vision of the requirements for ubiquity. Once these requirements are more clearly defined, their diffusion and appropriation by researchers and professionals will be made easier. Providing user control is one of the fundamental requirements that we have identified. Computer-supported collaborative technologies are one of the tools available to offer user control, and therefore in the design of a ubiquitous user experience.

1 The search for Ubiquity

As computers become an integral part on most people's lives and provide support to an increased number of human activities, the computers and associated computing devices need to be tightly integrated into people's environment. Ubiquitous computing (Ubicomp), also known as pervasive computing, is the field of research interested in this relationship, seeking to bring a new vision to computers, networks and their applications (Weiser 1999).

Computing has reached a stage of maturity in terms of technology and therefore research should shift to improve the user experience associated to computing activities (Bellotti et al. 2002). This shift implies a complete reconsideration of the relationship between users and computing resources (Norman 1998). New practices and design principles needs to be defined to help the transition from the focus on pure technological improvements to an enhanced user experience of computing anywhere and anytime.

2 The parameters for ubiquity

This research has focused on three components of the ubiquity experience (privacy, context and adaptation) and identified general parameters supporting ubiquity in system and application design.

- **Choice and granularity**: users need to be provided with opportunity to make choices for themselves. The system or application should offer various levels of choices for opportunity to fine tune information disclosure, exchange and retrieval (Lederer 2003).
- **Memory:** users expect the system to remember about past, present and future interactions in an useful way (Salber et al. 1999)
- **Information filtering :** system should filter the information flow between the environment and the user as efficiently as possible (Herlocker et al. 2000)
- **Laws and norms (Kobsa 2001)**
- **User Control (Ackerman et al. 2001)**
- **Interactive learning:** the system will learn from users continuously and will provide users with learning opportunities as well. (Arnstein et al. 2002)

Computer-Supported Cooperative Work technologies have a big role to play in this evolution toward a ubiquitous user experience in information systems. Once the requirements to integrate successfully CSCW technologies in system are defined (Lemhachheche and Porter 2005) , these technologies could support the integration of all of these parameters in the design of successful information systems.

3 References

Ackerman, M., T. Darrell and D. J. Weitzner (2001). 'Privacy in Context.' Human-Computer Interaction 16(2, 3 & 4): 167-176.

Arnstein, L., C.-Y. Hung, R. Franza, Q. H. Zhou, G. Borriello, S. Consolvo and J. Su (2002). 'Labscape: a smart environment for the cell biology laboratory.' Pervasive Computing, IEEE 1(3): 13-21.

Bellotti, V., M. Back, W. K. Edwards, R. E. Grinter, A. Henderson and C. Lopes (2002). Making sense of sensing systems: five questions for designers and researchers. Proceedings of the SIGCHI conference on Human factors in computing systems: Changing our world, changing ourselves. Minneapolis, Minnesota, USA, ACM Press: 415-422.

Herlocker, J., J. Konstan and J. Riedl (2000). 'Explaining collaborative filtering recommendations.' CSCW '00: Proceedings of the 2000 ACM conference on Computer supported cooperative work.

Kobsa, A. (2001). Tailoring Privacy to Users' Needs. Proceedings of the 8th International Conference on User Modeling 2001, Springer-Verlag: 303-313.

Lederer, S. (2003). Designing Disclosure: Interactive Personal privacy at the Dawn of ubiquitous computing. Computer Science Division. Berkeley, University of California, Berkeley.

Lemhachheche, R. and J. D. Porter (2005). 'Requirements for design of collaborative applications and systems.' to appear in ECSCW '05: Proceedings of the 2005 ACM European conference on Computer supported cooperative work, Paris, France.

Norman, D. A. (1998). The invisible computer, MIT Press.

Salber, D., A. K. Dey and G. D. Abowd (1999). The context toolkit: aiding the development of context-enabled applications. Proceedings of the SIGCHI conference on Human factors in computing systems: the CHI is the limit. Pittsburgh, Pennsylvania, United States, ACM Press: 434-441.

Weiser, M. (1999). 'Some computer science issues in ubiquitous computing.' SIGMOBILE Mob. Comput. Commun. Rev. 3(3): 12.

Studying Appropriation in Activity-Centric Collaboration

Michael J. Muller,
Suzanne O. Minassian,[1]
Werner Geyer,
David R Millen,
Elizabeth Brownholtz,
Eric Wilcox
IBM Research
{michael_muller, Minassian, Werner.Geyer, david_r_millen, beth_brownholtz, eric_wilcox} @us.ibm.com

Abstract. We describe a case study of appropriation of a research prototype by a 33-member research community, leading to reinvention of the prototype and a successful transfer to product. Based on those experiences, we propose some lessons learned about designing for appropriation.

1 Introduction

This position paper contains two sections. The major section describes a case study of appropriation in an activity-centric collaboration environment. The second section proposes lessons learned about designing for appropriation in collaborative computing environments.

[1] Now in IBM Software Group,

2 Case Study of Appropriation: ActivityExplorer

We designed ActivityExplorer (AE) as the client portion of our research prototype Instant Collaboration (IC), an experiment in activity-centric collaboration. AE was used by a research community of 33 people during the summer of 2003, and helped us to define a class of computing environments that fall midway between unstructured, ad hoc collaborations (e.g., instant messaging, email) and highly structured, formal collaborations (discussion databases, teamrooms, group decision support systems). The research experiences of 2003 led to a decision in 2004 to include AE as a feature on IBM Workplace Collaboration Services, which was released in 2005 (IBM, 2005). This report goes beyond previous descriptions of the types of activities that emerged (Muller et al., 2004), and the patterns of

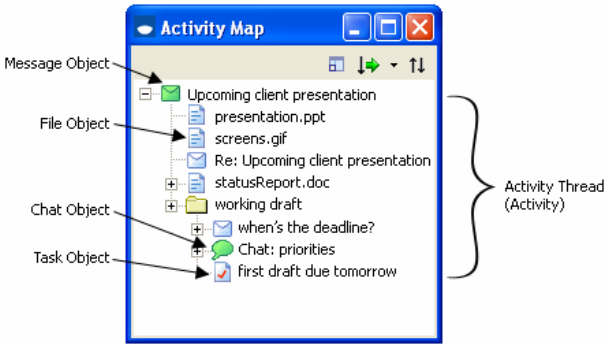


Figure 1. Activity Thread from ActivityExplorer.

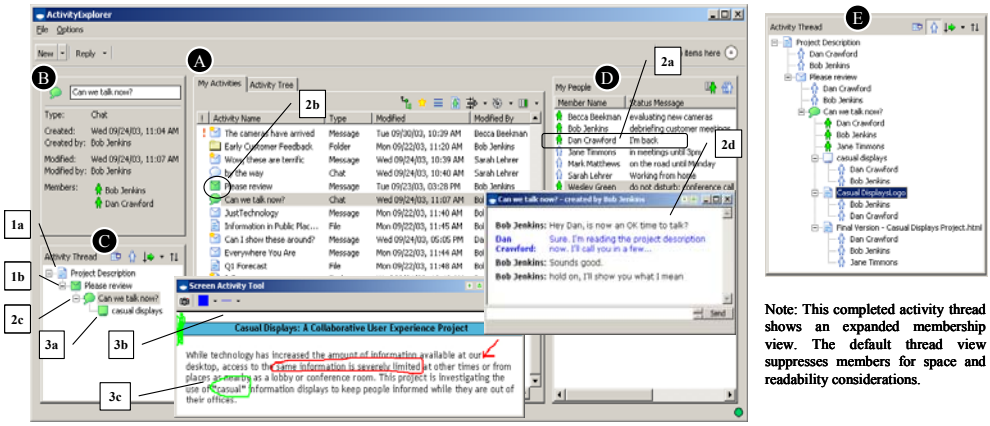


Figure 2: ActivityExplorer. A. Overall list of all shared objects. B. Details of one shared object. C. Activity Thread showing one structured collection of shared objects. D. Enhanced buddy list. E. Another Activity thread. 1a. Activity Thread, beginning with a file object. 1b. Message object, currently accessed by at least one collaborator. 2a. User with "live" online status indicated. 2b. Message object in overall list of objects, currently accessed by at least one collaborator (same object as 1b, but not shown in 2b without Thread context). 2c. Chat object, currently accessed by at least one collaborator. 2d. Chat tool with contents. 3a. Shared screen image, currently being accessed by at least one collaborator. 3b. Shared screen tool displaying shared image, currently being annotated by two collaborators. 3c. Shared screen tool contents.

media usage in the project (Millen et al, 2005), to focus on our experiences with reinvention of the prototype's usage by its users.

Figure 1 shows the basic unit of AE, namely an Activity Thread. An Activity Thread is a structured collection of objects that are shared among a group of users. Available objects in the 2003 research version included messages, files, persistent chats, shared screen images, tasks, and folders.[1] In limiting cases, an Activity thread could include only one object, or could be "shared" by only one person.

As shown in Figure 1, members of an Activity Thread use a hierarchical metaphor to structure their shared "live objects." Each object is "live" in the sense that its icon changes if someone is currently accessing it.[2] A buddy list of other users supports the more common "live names" functionality, such as indicating online presence of a collaborator (Figure 2). As a further extension of "live" functionality, we provide an alerting service. Whenever an Activity Thread was touched by a collaborator (e.g., reading, modifying, or creating new content), all members of that Thread were informed of the action via an alert message in the Windows system tray. Some of these attributes of "liveness" are similar to Dourish's "active properties" in placeless documents (Dourish 2003).

As reported in Muller et al. (2004), we had designed AE to be used by relatively small groups of collaborators, for relatively brief periods, using a handful of objects, in each Activity Thread. We had hypothesized that small, ad hoc collaborations would continue to occur in chat and email, and that large, formal collaborations would continue to occur in discussion databases. Indeed, we found 110 Activity Threads (54%) that corresponded to this pattern (2-14 objects, 1-7 days duration, a small number of collaborators).

2.1 Appropriation

We were surprised by other Activity Threads. The student interns in our group in 2003 took over AE, and made it their home environment. Led by the interns' innovations, multiple groups of researchers also began to use AE in new ways. The result was that the users reinvented AE through use (e.g., Antón and Potts, 2001; Bikson and Eveland, 1996; see also Muller and Gruen, 2005).[3]

The unanticipated usages made some AE Activity Threads into simple chat vehicles -- out of 203 Activity Threads, a total of 71 (35%) contained a single chat object (with an average of 18.92 turns per chat, median 7 turns, range 1-222 turns). Thus, despite the fact that these 71 Threads contained a single object, the single chat object contained evidence (number of chat turns) of extended

[1] The product version does not currently include task objects.

[2] In the research prototype, the liveness was signaled by color changes. In a real product that supports universal usability, the liveness would require a more accessible signal.

[3] Some of the surprising results were clearly due to interns' activities. However, most of the surprising results also involved one or more non-intern research staff members, and fully half of the longest, most surprising Threads were primarily full-time staff collaborations.

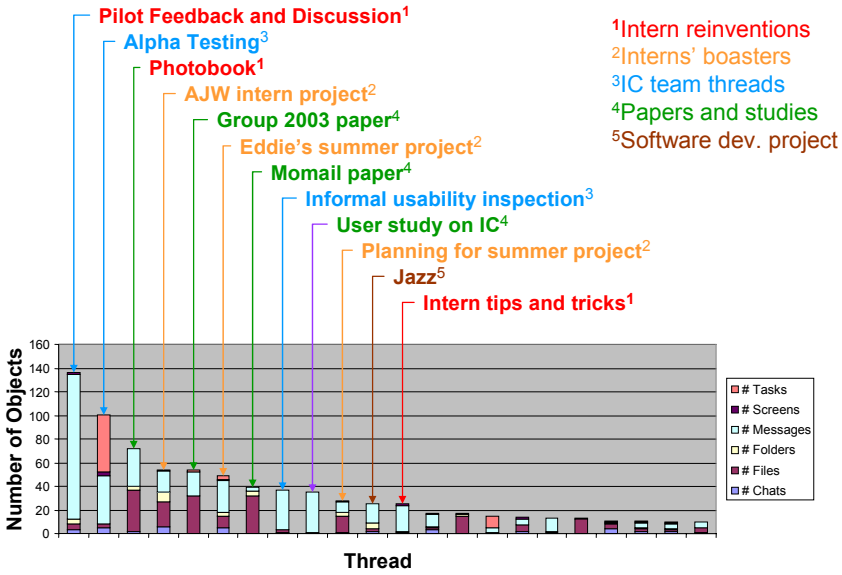


Figure 3. The large Activity Threads, with details about the 12 longest Threads.

collaboration. Other single-item Activity Threads were composed of a message object (24 Threads, 12%), a file object (12 Threads, 6%), a folder (3 Threads, 1%), a task (1 Thread, <1%), and a shared screen (1 Thread, <1%). There is not space in this position paper to analyze these non-chat objects in greater detail; briefly, we hypothesize that these were failed collaborations, in which the intended collaboration partner never responded.

The unanticipated usages also converted some AE Activity Threads into four-month community resources, as well as sites for detailed, extensive development of project contents, such as writing research papers for conferences. Figure 3 presents a summary of the longest Activity Threads. The *Alpha Testing* and *Informal usability inspection* Threads were directly related to the AE project. The *Pilot feedback*, *Photobook* and *Intern tips and tricks* Threads were examples of interns' reinvention of AE for their own community purposes. The *AJW*, *Eddie*, and *Planning* Threads were intern projects that generated large, partially archival sets of materials. The *Group 2003*, *Momail*, and *User study* Threads were researcher activities toward conference papers. The *Jazz* Thread was a researcher exploration of collaborative software development environments.

2.2 Consequences

There were several consequences of these patterns of unanticipated usage. First, AE became the default chat application for many of the interns. Some of them never loaded the IBM-standard instant message product, because their chat needs were met sufficiently through AE. Second, because the AE chats were persistent, users tended to revisit the chat transcripts (the interval between the last *write* into a chat and the last *read* of the chat was 7.55 days); some interns reported that they used the chats as reference materials for mentors' instructions in programming

projects. Third, again because of chat persistence, users tended to put informal communications into email, rather than into chat. The second and third consequences show how a simple change (chat *persistence*) can reverse the previously accepted "outeraction" hypothesis (Nardi et al., 2000) that chat is used in a preliminary and transitory manner to set up more formal, content-filled, and persistent collaborations in other media (see also Muller at al., 2003).

The fourth consequence presented difficulties. As we described in relation to Figure 2, AE supported several aspects of "liveness:" it displayed online status of collaborators (similar to instant messaging buddy lists), access status of objects, and alerts when objects in a relevant Activity Thread were added, updated, or even examined. For small groups of collaborators, the alerting feature appeared to be useful: People were notified immediately of changes to Activity Threads that were relevant to their work. However, for the larger Threads in Figure 3, these alerts became annoying and even burdensome. Many of the larger Threads had membership of all 33 people in the AE community. Any *read* of an object in one of these larger Threads would create an alert for 33 people. These community-wide Threads were not necessarily an essential part of each person's work. The result was that people were receiving apparently high-priority alerts for events that did not directly affect them. Further, we observed cases of "swarming," in which a simple *read* action by one person on an object would generate alerts to other people, who would in turn also read the object, generating further alerts in a feed-forward loop that was limited only by the number of people who were currently online. Users complained strongly about these distractions in the first research prototype of AE.

A fifth consequence will eventually present other difficulties of a more positive nature. As described above, we had anticipated that AE would be used for relatively brief, informal, non-archival collaborations. We learned that people might *begin* an Activity Thread in this manner, but that certain Threads could grow in size and importance until they contained unique project-related content of archival significance. Indeed, part of the strength of AE was that an informal collaboration could begin in a chat or a message, and could grow into a larger body of materials, eventually becoming important project records –*within the unitary AE environment*. By contrast, in more conventional work environments, users might begin the informal collaboration in a chat or email environment, and then *copy* their materials into an intermediate-scale collaboration environment such as a discussion database, and then *copy again* their growing body of resources into a content management system. Thus, the flexibility of AE – its ability to support the reinvention described in this paper – became a major strength and "value proposition." However, the use of AE for potentially archival records paradoxically highlighted its weaknesses in exactly those areas: the research prototype of AE had no mechanism for managing content over long

periods of time. The research prototype of AE had become more valuable than anticipated, and its value had outpaced its ability to manage that value.

The sixth consequence was also a direct outgrowth of the appropriation. The ability of AE to support multiple styles of work – highly informal, semi-structured, as well as archival – led to the adoption of AE as a feature of IBM® Workplace™ Collaboration Services product in its release 2.5, made public during the summer of 2005 (IBM, 2005). The reinvention and appropriation that we experienced contributed to the successful technology transfer of a research prototype into a product feature.

3 Designing for Appropriation

A number of research programmes have focused on the strategic use of ambiguity to foster appropriation and reinvention by users. Our work has learned from that tradition, and has applied some of its lessons in the provision of what might be called open-ended tools for work. Unlike work with cultural probes (Gaver et al., 1999) or related, deliberately ambiguous objects (Boehner et al, 2005a; Kaye et al., 2005; Sengers et al., 2005), our work begins with an anticipated context of use – somewhat more similar to the investigations of Dourish (2003) and Boehner et al., (2005b). Further, our work must have immediate use potential as perceived by its end-users; otherwise, in our organizational environment, there will be little usage and insufficient experience to lead to appropriation or reinvention. While we appreciate the importance for critical reflection on potential techno-centric values that may infiltrate design (Boehner et al., 2005a; Sengers et al., 2005), in our domain we continue to assume at least a subset of those techno-centric values.

In a somewhat more techno-centric manner, Dourish considered diverse experiences with appropriation (1999, 2003). He noted (1999) five aspects of appropriation, which are strongly tied to our results. *Flexibility* of technology was key, and indeed AE provided flexible means for representing relationships among objects. In Dourish's experiences, appropriation generally occurred in a *community*, and as we observed, the interesting aspects of AE arose as a community *incrementally* redefined how it was used, made those redefinitions *visible* to members of the community, and thus redefined what AE "was." Finally, the community's redefinition of AE left visible traces, and the *persistence* of these traces allowed both further reflection and appropriation, and our ability, as researchers, to study the phenomenon. Similarly to Dourish's (2003) description of the Placeless paradigm, AE offered the ability to co-create structures of diverse materials that would have been managed separately by traditional, media-specific applications.

However, our approach provided very little of the multiple levels of taxonomy that Dourish's placeless environment afforded. It appeared to us that the success of AE depended precisely on the co-construction of the *same* shared structure to

organize the shared resources. In this way, or experience differed from some of the attributes that Dourish (2003) found crucial for the success of placeless: multiple dimensions along which to organize information, user control of document behavior, and relaxation of requirements for the *same* organizational structure to achieve mutual intelligibility. When the interns – and later the researchers – transformed AE from a small-scale, transient collaboration environment into a flexible place for small and large, informal and formal co-constructions, their use appeared to depend crucially on having exactly the same representation for all users.

Also in a more techno-centric manner, we look to Pipek's accounts of appropriation in a work context. Pipek (2005) mentions several aspects of tailoring support that resonate with our users' experiences: articulation support (knowledge of who is doing what, and with what completion status), demonstration support (through shared screens). However, Pipek did not observe a crucial aspect of our experiences: the co-construction of structures for articulation support – i.e., the piecemeal, object-by-object joint creation of the structure of shared resources (e.g., as shown in Figure 3). Interestingly, much of what our users accomplished was done without focused discussions (e.g., Pipek's concept of "use discourses"). Because our users were using AE primarily in order to do other work (work that was not "about" AE), it appears to have been an advantage that they could co-construct their resources without needing to engage in a focused dialogue about the tool itself. These experiences are in strong contrast with Pipek's research, and with many of the tailoring and appropriation studies that he surveyed in his thesis.

Based on these comparisons, we offer the follow additions to the preceding advices, observations, and conceptions of design for appropriation.

- **Changing reference systems** – Users tended to move their focus from informal dyads, to small teams, to large communities. They appeared to do so fluidly, without needing to establish or declare the scope of their frame of reference.
- **Graceful adaptation to changing size and membership** – In consequence, AE needed to be able make graceful accommodations to these changing reference systems. AE was successful in providing an informal "growth path" from small, informal, ad hoc, to large, formal, and archival. The research prototype of AE was unsuccessful in accommodating the different notification, liveness, and archiving requirements of these different reference systems.
- **Changing usages and valuations** – We observed distinct changes in the usage (and hence, the meaning) of certain media. The strongest case was the transformation of instant messaging from a transient, assistive technology (the outeraction hypothesis) into a medium that itself

contained valuable records that people referred to an average of a week after they were inscribed.

- **Inversions of importance** – As AE collaborators worked out their new usage patterns, the importance of different media changed. The strongest example is the inversion of importance and formality that occurred between chat (more formal, more public in AE) and email (less formal and more private, by comparison).
- **Increased requirements for user control** – As AE became more a part of collaborators' work, their need to control its attributes increased. We saw this most clearly in users' experience of being overwhelmed by alert notifications, but we also learned of a number of other desirable user control features that we are still working on.

We look forward to comparing our experiences with those of others in the workshop. We hope develop a better understanding of which aspects of appropriation are more important in different settings, depending on technology, environment, and most crucially shared practices. Ultimately, shaping the use and meaning of technology is part of individual and collective democratic practice, and should become one of the formative phenomena for HCI in the 21st century.

4 References

Antón, A.I., and Potts, C. (2001). Functional paleontology: System evolution as the user sees it. *IEEE Conf. on Software Engineering*.

Bikson, T.K., and Eveland, J.D. (1996). Groupware implementation: Reinvention in the sociotechnical frame. *Proc CSCW'96*.

Boehner, K., Shay, D., Kaye, J., and Sengers, P. (2005a). Critical technical practice as a methodology for values in design. Position paper at CHI 2005 workshop, *Quality, Value(s), and Choice: Exploring Deeper Outcomes for HCI Products*.

Boehner, K., Thom-Santelli, J., Gay, G., Sengers, P., and Hancock, J.T. (2005b). Treading uncommon ground: Designing for new shared experiences through appropriation. Position paper at CHI 2005 workshop, *Designing for Community Appropriation*.

Dourish, P. (1999). Evolution in the adoption and use of collaborative technologies. Position paper for ECSCW'99 workshop, Evolving Use of Groupware. Available at http://www.ics.uci.edu/~jpd/publications/ 1999/ecscw99-evolution.pdf (verified 20 June 2005).

Dourish, P. (2003). The appropriation of interactive technologies: Some lessons from placeless documents. *Journal of CSCW* **12**(4), 465-490.

Gaver, B., Dunne, A., and Pacenti, E. (1999). Cultural probes. *Interactions* **6**(1), 21-29.

IBM Corporation (2005). http://www.lotus.com/products/product5.nsf/wdocs/workplacehome (verified 20 June 2005).

Kaye, J., Levitt, M.K., Nevins, J., Golden, J., and Schmidt, V. (2005). Communicating intimacy one bit at a time. *CHI 2005 Extended Abstracts*.

Millen, D.R., Muller, M.J., Geyer, W., Wilcox, E., and Brownholtz, B., (2005). Patterns of media use in an activity-centric environment. *Proc. CHI 2005*.

Muller, M.J., Geyer, W., Brownholtz, B., Wilcox, E., and Millen, D.R. (2004). One hundred days in an activity-centric collaboration environment based on shared objects. *Proc. CHI 2004*.

Muller, M.J., and Gruen, D.M. (2005 in press). Working together inside an emailbox. *Proc. ECSCW 2005*.

Muller, M.J., Raven, M.E., Kogan, S., Millen, D.R., and Carey, K. (2003). Introducing chat into business organizations: Toward an instant messaging maturity model. *Proc, GROUP 2003*.

Nardi, B.A., Whittaker, S., & Bradner, E. (2000). Interaction and outeraction: Instant messaging in action. *Proc. CSCW 2000*.

Pipek, V. (2005). *From tailoring to appropriation support: Negotiating groupware usage*. PhD thesis, Oulu University. Available at http://herkules.oulu.fi/isbn9514276302/ (verified 20 June 2005).

Sengers, P., Boehner, K., David, S., and Kaye, J. (2005 submitted). Reflective design. Submitted to Critical Computing Conference, Århus.

Let's harness IT for our purposes...!

Bettina Törpel
Department of Manufacturing Engineering and Management, Technical University of Denmark, Building 424, DK-2800 Lyngby, +45 4525 6085
bt@ipl.dtu.dk

Abstract. This workshop discussion statement contains a partial embedded outline of the double concept of objectification-appropriation as proposed by Critical Psychology.

1 Introduction

This workshop discussion statement contains an embedded outline of the double concept of objectification-appropriation. The outline is "embedded" in the sense that a number of related concepts from the same theoretical frame are introduced. The theoretical reference frame is Critical Psychology. Critical Psychology, which was mainly developed at the Free University in West Berlin, is a school of thought in the tradition of Activity Theory that was most influential in the 1970s and 1980s (cf. Holzkamp 1983; an introduction in English is provided by Tolman 1994). The version of the double concept presented here

- is a condensed and partial summary of the author's reading/understanding of this double concept in Critical Psychology,
- is meant to serve as discussion impulse for clarifying what we actually refer to when we talk about "appropriation" and
- comes together with an outline of the kind of research and development process the author thinks is both beneficial and sustainable.

For the purposes of this (necessarily extremely reduced) embedded overview, I will only depict particular aspects of the meanings of selected Critical Psychology

concepts, namely agency, (inter-) subjectivity, reasons, meaning, functionality and objectification-appropriation.

2 Agency

Agency (Holzkamp 1983, especially chapters 6.3 and 7; in English sometimes referred to as "action potence") denotes the kind, and extent, of presence and personal availability of, possibilities for acting and influencing phenomena in a way that is relevant for the individual. Each person's agency corresponds with the relation between, on the one hand the general pool of possibilities (and restrictions) that are available and attainable under particular historical-societal circumstances and on the other, the way these given circumstances with their generally available possibilities are subjectively apprehended by the person.

3 Subjectivity, intersubjectivity

For the purposes here I will refer to Critical Psychology's concept of *subjectivity* (ibid, pp. 233 ff) mainly as a concept that denotes differences between individuals, e. g. in their ways of experiencing, interpreting, reasoning and having reasons. Subjectivity (including subjective reasons) is grounded in each person's respective specific socio-material conditions - but cannot be derived from studying them together with the person's movements, expressions etc. Hence, according to Critical psychologists such as Holzkamp (1983), it is never possible for one person (or group of persons) to appropriately grasp, predict or change another person's subjective experiences, reasons or »world«. Rather, this has to be reconstructed and »approximated« from each subjective standpoint, possibly by interleaving several individuals' perspectives in *intersubjective* exchange (ibid, pp. 233ff and chapters 7 and 9).

4 Meaning

In Critical Psychology, the concept of *meaning* (always meaning related to a specific phenomenon; ibid, pp. 172-174 and chapter 6.3) is closely linked to purpose and denotes what »one« can do (with this phenomenon). Two aspects of meaning have to be distinguished: meaning in its generalized aspect and in its specific aspect. *Generalized meaning* refers to the »societal average« meaning, the prevalent, widespread or common meaning of a phenomenon. *Specific meaning* (ibid, chapter 7.5) denotes the societally mediated meaning of a phenomenon in contingency to specific circumstances: location, historically specific constellations and situation as subjectively experienced by specific

individuals or groups. Here, it is central to keep in mind that most phenomena of every-day relevance have been created by humans, exactly for serving specific purposes (or sets of purposes). A purpose is usually a purpose for a person or a group of persons themselves; but often it is also expected that this purpose is generalized in the sense that the purpose will emerge as an issue for other people as well. People then create a new phenomenon (e. g. artifact, work means, computer application) - one that serve this purpose that has not yet been served before - for future application by themselves and possibly by others, and this new phenomenon may later be used by other people who face the same limits and who have not yet had anything available that has served this purpose.

5 Functionality

The concept of *functionality* of means (e. g. computer applications) as used here (and elsewhere, cf. Törpel 2004a, b) has not yet been a focus of Critical Psychology; it is always related to some specific means, in a way that denotes the meaning of the means: what can be done or achieved with a means, what purposes it serves. The central point of this concept of functionality is that what can be done »with« a means, is often not entirely »contained in« the means itself, even though functionality, especially of a computer application, often becomes attributed to the artifact itself (for critiques of attributing distributed functionality to individual artifacts see e. g. Suchman 2000, Latour 1999). Instead, the actual »functionality-in-active-use« that is ascribed to the means emerges in an interplay between the means (e. g. computer application) and other (historically grown, co-developed and differentiated) phenomena, such as

- further devices, work means, artifacts, infrastructures which are actively in use and in which a new means becomes integrated,
- characteristics of the actors (or actor groups) who want to develop, introduce, harness, make use of etc. a specific new means,
- purposes, needs, desires, agendas, objectives of the actors for the development, introduction etc. of the means,
- the actors' specific practices and
- the involved actors' social structures in which they act, e. g. organizations, circles.

In this way, the functionality attributed to computer applications is framed as always being a *distributed and dynamically evolving functionality in use*.

6 Objectification-Appropriation

The double concept of objectification and appropriation becomes relevant in connection with any activity-theoretical consideration of meaning (of phenomena,

artifacts, means, computer applications created by humans; cf. Holzkamp 1983, pp.176-178). Phenomena (as created by humans) and their meanings develop further over time, or rather: humans develop them further. This may be seen both from a generalized and a specific point of view. Any individual human or group of humans does not have an alternative to acting (also: perceiving, experiencing, interpreting etc.) from their subjective standpoints. A phenomenon may have a generalized meaning - but there is no authority (such as a »generalized meaning assessment committee«) that ultimately and objectively can assess, arrive at a conclusion and enforce what this generalized meaning is at one given point in time. Informed guesswork (interpretation) cannot be avoided. Nevertheless, when someone creates a phenomenon (e. g. a means) that serves a specific purpose, helps do a specific job, overcome a limit - after the experience of not yet having been able to serve this purpose, get this job done - this often implies that the creators assume that this limit will be faced by others as well. Yet - nobody can ever be sure that another person's or group's specific circumstances provide in fact »a case« of the kind of circumstances that resulted in the development of the original phenomenon (solution, artifact, means, computer application). This can only be investigated by the new human or group themselves.

The local specific inquiry and the harnessing of the phenomenon (artifact, means, computer application) in its meaning/functionality is what I refer to as *appropriation* (building on Leont'ev 1973, Leontyev 1981, Holzkamp 1983). The altering and codifying aspect of discovering, harnessing and realizing new possibilities, meanings, functionality of phenomena that results in new (versions of) phenomena that serve new generalized purposes is what I refer to as *objectification* (building on Leont'ev 1973, Leontyev 1981, Holzkamp 1983).

7 What is needed...

In my view, the double concept of objectification and appropriation, as further developed by Critical Psychology can provide a powerful reference concept for developing technologies from the standpoints of the affected subjects. What is needed are operationalizations and practical guides for beneficially and responsibly objectifying and appropriating notions related with or attributed to the functionality of computer applications. Another way to put this is that I think we somehow need processes, methods and devices that support reflective and creative development practices of computer applications toward the improvement of the participants' quality of life. An objectifying process could then be interpreted as a process of giving notions, concepts, practices that are in use a more »solid«, durable, generalizeable and pervasive form when they become incorporated in computer applications than when they remain in other forms, such as oral or written forms. Regarding the aspect of appropriation, I think we are in need of processes, methods and devices that support practices of understanding, utilizing,

re-capturing, re-inventing, questioning etc. of existing IT work means (durable, generalizeable and pervasive forms of notions, concepts and practices that are in use) that are in use.

8 How could we proceed in research and development? - The double concept of objectification-appropriation utilized for an emancipating role of research and development

In the view of the author, an appropriate way of approaching research on and development of computer applications could be one that is geared toward finding, exploring, and realizing possibilities of fulfillment and enjoyment that are within and beyond the current possibilities. This includes becoming familiar with, grappling with, and overcoming one's own limits, as well as (and in relation to) the limits of the currently existing conditions of one's life. In most societies this also implies the need to address those relations of power that on the one hand »build a frame« around people's every-day lives and on the other manifest themselves as immediately experienced, concrete, relations of possibilities and restrictions.

Such approaches to research and development would hence primarily be geared towards improving the participants' quality of life by extending spaces of possibilities, in the sense that wider conditions are improved (or at least taken into consideration) as part of the improvement measures. As in some other research programmes (e. g. ethnomethodology), the research programme of Critical Psychology requires professional researchers to be interested in those of their own practices and life conditions that they at least partially share with other people who are not professional researchers, and whose practices and conditions are scrutinized during the research. People who participate in research, but are not professional researchers, must be considered as either already qualified to be co-researchers or to receive support by the professional researchers in their progress towards becoming co-researchers (Holzkamp 1983, chapter 9).

The kind of approach to research and development that is proposed here is very research-oriented, yet not necessarily in the sense of academic research, but in the sense that people are interested in understanding, and possibly overcoming, the current conditions, practices, limits and possibilities of their every-day lives. This implies seeking to understand how current phenomena have evolved and how, given the way they are, they might further develop. It also implies seeking to discover how things could be totally different than they currently are; how oneself, maybe together with fellows (e. g. peers, colleagues, allies, fellow-

sufferers), could influence historical trajectories in a beneficial direction. From this viewpoint, *causes*, *historical trajectories* as well as *reasons* (from the standpoints of the subjects) are all assumed as important constituents of individual and societal reality. In short, such an approach would be inquiring, communicative, alliance-oriented, understanding, improvement oriented, active, activating, fulfillment oriented, and full of respect towards, history, feelings, meanings and reasons (of one's own and others).

9 References

Holzkamp, K. (1983). Grundlegung der Psychologie. Frankfurt a. M.: Campus.

Latour, B. (1999). Pandora's hope: essays on the reality of science studies. Cambridge, MA: Harvard University Press.

Leont'ev, A.N. (1978). Activity. Consciousness. Personality. Englewood Cliffs, NJ: Prentice-Hall.

Leontyev, A.N. (1981). Problems of the Development of Mind. Moscow: Progress.

Suchman, L. (2000). Human/Machine Reconsidered. Published by the Department of Sociology, Lancaster University at: http://www.comp.lancs.ac.uk/sociology/soc040ls.htmlor http://www.comp.lancs.ac.uk/sociology/research/restopic.htm#spatiality. Last accessed: November 26, 2004.

Törpel, B. (2004a). Forming circles and making things happen - Designing functionality that supports cooperative work in fragmented work environments. In: Proceedings of the 27th Information Systems Research Seminar in Scandinavia, IRIS 27 - Learn IT, Know IT, Move IT - August 14-17, 2004 in Falkenberg, Sweden.

Törpel, B. (2004b). Narrative Transformation - Designing work means by telling stories. In: Bertelsen, O. W., Korpela, M. & Mursu, A. (eds.), Proceedings of the First International Workshop on Activity Theory Based Practical Methods for IT Design, 2-3 September, Copenhagen, Denmark. Aarhus, DK: DAIMI Report PB-574, 122-133.

Tolman, C. W. (1994). Psychology, Society, and Subjectivity: An Introduction to German Critical Psychology. London: Routledge.

“Let them use emacs”: the interaction of simplicity and appropriation

Michael B. Twidale,
M. Cameron Jones
Graduate School of Library and Information Science
University of Illinois at Urbana-Champaign
twidale@uiuc.edu, mjones2@uiuc.edu

Abstract. We look at appropriations in a number of contexts, and find that often they involve very simple ways of coordinating functionality that seems to be at odds with much of the approach of computer systems design that emphasizes power and abstraction. We speculate about how we can extend this simplicity in other ways.

1 Introduction

The infamous quotation attributed to Marie Antoinette has been used to illustrate how completely out of touch the *ancien regime* was from the circumstances of the starving peasantry, and consequently how inappropriate to the point of insulting was the proposed solution of eating cake. There are times when the aristocracy (or should that be priesthood?) of computer scientists can, by their proposed design solutions, appear to be equally out of touch from the needs of end users. Designing an incredibly powerful application with thousands of reconfiguration options and its own built in programming language to enable extensibility does not mean that it will be adopted, appropriated or tailored by end users for their own unique and evolving needs. EMACS is not the solution nor is it the ideal design paradigm. This issue persists despite the great progress made in advocating for user centered design and the development of techniques to integrate better

understanding of real workplace needs and practices into systems development. Although at times this recurrent problem can be attributed to persistent developer arrogance or ignorance of real user needs, we suspect that it is also in part due to a fundamental part of the design ethos that most computer scientists absorb both in formal education at universities and in ongoing professional practice. This ethos involves a concentration on functional power, flexibility and abstraction in the development of effective and elegant design solutions. It has enormous merit in nearly all contexts and in no way are we advocating for its erosion. But we do at least fear that it can sometimes get in the way of end user technological appropriation. In this concept paper we attempt to outline some of our thinking about simplicity and appropriation in the light of experiences with two different research projects, and some initial work in trying to characterize the aspects of the appropriation process.

Given that different people use the word 'appropriation' slightly differently, we should clarify that we are not so much interested in the adoption of a technology by types of people who were not expected or even allowed to use it (e.g. Eglash 2004). Rather we want to focus on cases of innovative use of that technology in ways that its developers had not envisaged, planned or explicitly designed the tool to support. That is, we concentrate on appropriation-as-innovation rather than the alternate (equally legitimate) view of appropriation-as-empowerment. In this way the users of technologies can be involved in some way in the co-development of the technology (Fischer 2002), without having to turn themselves into computer scientists in order to be technology developers.

2 Paper Prototyping from Forms into Databases

A study (Twidale & Marty 1999, 2000, Marty 2005a, b) of workflow practices in a museum that was simultaneously undertaking a digitization and a packing process prior to a move to a new building revealed numerous cases of end users appropriating aspects of the available technology to meet their needs. With 40,000 artifacts to pack and move, and relying on significant numbers of keen but relatively unskilled undergraduate workers, the museum staff had to design a system so that no artifact could get lost, even though they knew that many mistakes would inevitably be made. Worse, they had never packed and moved their entire collection before, so the workflow process itself would have to change as they learned by doing. External shocks would inevitably force yet more changes to the process – due to changing budget constraints, building schedule changes, etc. The packing process itself involved various paper forms that were used to record who had done which step of the workflow on which day for any given artifact. This allowed for the tracking of all actions and the rollback and recovery from errors. What is notable is how these forms, especially the packing sheet, evolved over time. People would use the form in ways it had not been

intended for, writing additional information and notes about exceptional issues with a particular artifact in the margin. Some of these ad hoc solutions to a particular problem with a particular artifact gradually evolved into practices that were applied to other related exceptions or to whole categories of artifacts. The forms themselves were revised so that later printouts for subsequent artifacts now had special fields and checklists derived from those handwritten practices. The database itself (created in-house using FILEMAKER PRO) evolved in the light of those changes to provide additional fields and automated checks. Effectively this was a kind of participatory design using paper prototyping. However, unlike conventional participatory design, the paper prototypes were part of the real work process and were actually used, rather than being developed purely to inform the design process.

In a similar way, certain aspects of the database and how it was used were appropriated by people who were not themselves database designers or any kind of programmer to enable them to do their job more effectively. In entering data about an artifact into the database, sometimes a user (particularly an undergraduate) wanted to record their doubts about a particular value. Rather than just guessing and moving on they might enter a value followed by question marks or even type in a comment such as "I don't know". These were messages through the database to other users of that database in the future – maybe the same person as entered the database, but several thousand artifacts later. Such commentary or annotation is very common in paper records (as marginal written comments or addition pieces of paper clipped or attached to a form), but is rarely seen in databases, where the prevailing assumption is that all uses of the database can and should be planned in advance of actual data entry. This might be desirable for optimal design, but we suspect it is rarely achievable in reality. Similarly, the database was used for various checking processes in later stages of packing and unpacking. Sometimes people would deliberately enter a value into the database that they knew would trigger an error for that particular artifact at one of these later checking stages. This allowed them to pass a message forwards in time to do an extra step with that artifact at that time that they knew would otherwise get forgotten about. Again, in the physical world this is entirely unremarkable – scrawled marginal notes, sticky notes and supplemental pages paperclipped to the main paper form are the norm in paper workflows, but exceptional or at least rarely remarked upon when it comes to databases.

This particular workplace study was admittedly an extreme case where the organization really did not know much about what precisely they wanted to do, had no experience with the moving process and yet had to get started, but we believe that the problem of designing for change is one that should pervade CSCW systems development.

3 Copy and Paste as Computational Duct Tape

A study of workplace help-giving in a variety of different settings (Twidale & Ruhleder 2004, Twidale 2005) revealed that in almost all circumstances people were using more than one application at a time. To achieve the work related goal, it was often the case that an element was copied from one application and pasted into another. Documents were frequently attached to emails and passed on to others for comments and further copy and pasting. The use of copy and paste was particularly evident when problems arose and a colleague was asked for help. If a direct solution could not be found quickly, a workaround was developed and again this frequently involved more than one application and copying and pasting to enable the work to be done. Hopefully this kind of interaction is so familiar as to be entirely unsurprising, and yet we believe it to be worthy of more detailed consideration of what it achieves, how it achieves it, and what might be done to improve or extend the approach. Effectively the people studied were creating very complex and sophisticated workflows, but without the use of or benefits from a workflow system. The workflow was mostly mental as they copied and pasted between applications and handed over work elements to others either via email or a shared database. A conventional workflow system might well have helped them optimize their standard actions, but the power of copy-paste was in its ability to deal with exceptions, and especially those whose resolution meant yet another workaround with yet another application, and possibly another person to help out. For example, a report may need some data in a particular form and style. The data might be collected by a number of searches in an internal database, and on the web and pasted into Excel so that all the results could be composed. A graph might be drawn in Excel, but require some tweaking that was done in Paint before pasting it into a Word document and passing it on to a colleague for help. Such a process might be neither elegant nor optimal (the user might be unable to use the report generation facilities in the database and hence be forced to manually compose their data in Excel, and likewise not want to bother with the full power of Excel graphing and so do some tweaking in Paint just because she happens to be more comfortable with that tool), but it does get the job done and can cope with a slightly different job next time.

4 Non-Computational Appropriations

We are accumulating a set of real-life appropriations to help us in trying to understand the aspects of design that seem to afford appropriation. Non-technological appropriations are particularly easy to collect as stories from non computer scientists. The physical nature of these appropriations can also help in thinking about what helped afford them, and also help in thinking about design for

appropriability both in conventional applications and those exploiting aspects of tangibility found in many ubicomp settings. Here are two examples:

4.1 Ad hoc conference calling

The first happened to one of us, amongst a group of very technologically savvy people. It shows two cell phones clustered around an office phone. The problem was to coordinate a synchronous distributed telephonic meeting – hardly a complex activity, and particularly ironic as it was for a CSCW design project. The solution illustrated of putting together two cell phones each with a connection to a remote participant alongside a desktop phone with a speaker option connected to a third remote participant is in many ways inelegant. The sound quality was not perfect, but enough to get the work done. Given various constraints of time and technology (including time to learn about various possible other more elegant solutions) the appropriation of physical affordances illustrated in the photograph was good enough to get the job done, and in many ways faster and better than other experiences we have had with trying to set up synchronous distributed meetings with far more sophisticated technologies. How did it come about? The fact that the nature of the solution is comprehensible to the reader solely from the photograph and without complex explanation of what was done with a CSCW infrastructure is, we believe, an interesting clue. The idea that putting the phones near one another will enable some sort of rough and ready conference call does not seem to require much of a flash of design inspiration, far less in fact than is often needed to figure out how to set up conference calling on most purpose-built phones. In this example the tangibility of the different artifacts and the ease in which different functions are perceived as being associated with different places (microphone, speaker), seem to help in affording appropriation. By contrast, the conventional program with many features all treated as a single complex unit seems to get in the way. We wonder if these aspects of tangibility and separability mean that it will be easier to design for appropriability in a ubiquitous computing context.

4.2 Raffle tickets

This example is interesting because we believe it to be the case where the appropriated use (raffle tickets) is far more prevalent than the originally intended designed use (coat checking). Coat check tickets make ideal raffle tickets because they both need unique tally options, are cheap and easy to separate, and both serve a similar job of separating and bringing together in the future. The idea of using the physical device for a quite different purpose that happens to exploit the affordances outlined above has propagated so much as to dwarf the originally intended use.

5 Conclusion

As our examples have illustrated, many of the successful appropriations that we have observed have involved very simple coordinations of technologies and technological elements that may have relatively limited functionality, and so may be regarded by end users as simple, even if their programmatic execution is complex (but hidden). The simplicity of copy-paste, or writing things into a database field definitely seem to have contributed to the success of the appropriations. But does that mean that simplicity is a necessary attribute of appropriability? Does that mean that we should give up on developing more sophisticated tools, interfaces and functions and just concentrate on these simple, rather mundane-looking ones? We are sure that it is not a good idea to go to the extreme of full power EMACS-like generic toolkits, but we hope that it is possible to develop functionalities that offer slightly more power than copy-paste.

One aspect of the issue seems to be the interaction of granularity and connectivity. Too fine a granularity and one ends up with the power but the associated complexity of EMACS and programming languages. Too coarse a granularity and one has purpose built applications that may have some pre-planned built-in tailoring and customization options, but that do not easily support innovative re-use. Connectivity via widespread provision of copy and paste facilities allow for combining functionalities between applications but without the power and complexity of semi-automation. Automation, iteration and branching get to the heart of computing, both the power and the complexity of programming. No matter how helpful, congenial, benign or graphical the interface, once these features are available, end users are in some way programming and will need to acquire the basic concepts of trying to plan for unexpected and unwanted outcomes such as dead ends, anomalous inputs and outputs and appropriate termination. The manual control of the flow of control seen in our examples looks computationally weak, but paradoxically may be a strength in terms of supporting imaginative leaps, simple experimentation and robust ad hoc usage.

It can be rather disconcerting for computer scientists to consider this aspect of appropriation, should further evidence accumulate to substantiate our claim. The training of computer scientists frequently emphasizes certain design aesthetics. Power and sophisticated functionality are naturally valued. It would seem very odd to extol one system because of how *little* it did compared to another. Even if such a minimalist design aesthetic is adopted (and there is evidence for the Open Source usability debate that that is not a foregone conclusion), there is also the problem of the abstraction and extensibility aesthetic. This is sometimes typified as: 'don't build a thing – build a thing builder'. It means that designers typically aim for generic solutions to problems to maximize power and re-use. This can mean incorporating many tailorability options, configuration settings, and even a programming language within an application so that it can be modified to address a host of future unanticipated needs. The concept of the abstract data type reveals the nature, and the power of the approach. But can such raw power (apparent in many UNIX commands – and of course EMACS), which is so useful to a skilled programmer, actually be a handicap for the end user in appropriating for any purpose? Can even the abstract names given to such functionalities deter appropriation by end users? As our work in this area continues, we plan to systematically explore the different features that support appropriation and to develop illustrator applications to test out those features.

6 References

Eglash, R. (2004). Appropriating Technology: An Introduction. In R. Eglash, J. Crossiant, G. Di Chiro and R. Fouché, (eds.) Appropriating Technology: Vernacular Science and Social Power. University of Minnesota Press.

Fischer, G. (2002). Beyond "Couch Potatoes": From Consumers to Designers and Active Contributors. First Monday 7(12)

Marty, P. (2005). Factors Influencing Error Recovery in Collections Databases: A Museum Case Study. Library Quarterly. In Press.

Marty, P. (2005). Factors Influencing the Co-Evolution of Computer-Mediated Collaborative Practices and Systems: A Museum Case Study. Journal of Computer-Mediated Communication. In Press.

Twidale, M.B. & Marty, P.F. (1999). An Investigation of Data Quality and Collaboration. Technical Report ISRN UIUCLIS--1999/9+CSCW.

Twidale, M.B. & Marty, P.F. (2000). Coping with errors: the importance of process data in robust sociotechnical systems. Proceedings, CSCW'00, Philadelphia, 269-278.

Twidale, M.B. and Ruhleder, K. (2004). Where am I and Who am I? Issues in collaborative technical help. Proceedings, CSCW04. 378-387.

Twidale, M.B. (2005). Over the shoulder learning: supporting brief informal learning. To appear in Computer Supported Cooperative Work: The Journal of Collaborative Computing.

Supporting Configuring as Appropriation Work

Ina Wagner
Vienna University of Technology

Ellen Balka
Simon Fraser University

1 Background

Based on two different but complementary cases – the '*mixed media case'* (architectural design work) and the *'wireless call system case'* (hospital work) – we argue that taking advantage of the potential that configurable systems offer may require attention to several 'qualities of use'. Here we provide further elaboration of the concept of configurability, which we have suggested elsewhere (Balka et.al., 2005).

1.1 Configurability of space and technology relations

Different tasks may require different spatial set-ups – people may need to be able to configure their workspace and the equipment they need with ease (this has implications for the design of the space and the artefacts that populate it). Different tasks may require different configurations of hardware and software, input and output devices, etc. The wireless call system is made up of a combination of telephone console, coloured lights, wireless phones, and 'alarms' that can be triggered in different places (patient bed, bathroom, and so forth), or can be connected to different devices, such as beds or intravenous pumps. It is

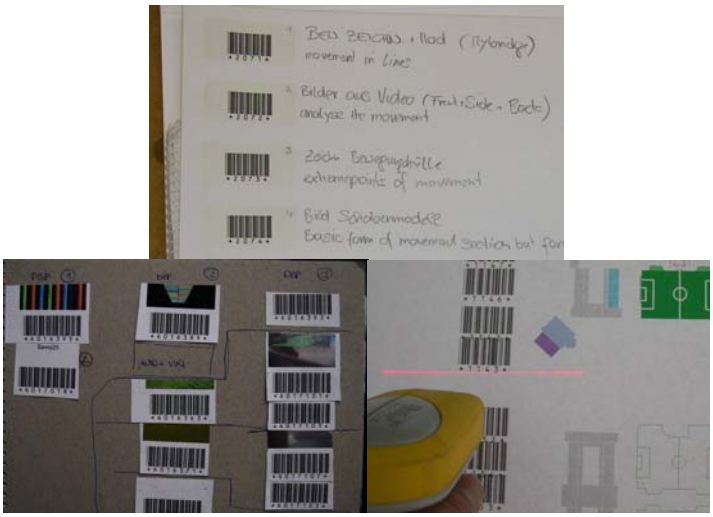
also hoped that the wireless handsets will help quiet the ward, by replacing alarms audible to all with alarm calls heard only by staff. The system (on principle) supports varying physical landscapes of alarms and displays, which, if connected with the mobile phone system, can be accessed from any place.

1.2 *Configurability of connectivity* (of people, places, materials)

Configurability also has to do with the possibility for people to arrange and re-arrange their connections to other people and to particular places, taking account of e.g. a varying spatial organization of activities or of changing patterns of availability. The wireless call system was intended to reduce time and space constraints associated with care delivery in a complex team environment, and in doing so, to improve connectivity—of patients to staff, and staff in varied locations to one another. Configurability also may refer to a capacity for assembling and re-assembling materials (design representations, patient information) so as to shift perspective, gain a particular point of view, support specific activities, and so forth.

1.3 Configuring as direct engagement – transparency and accountability

Designing environments so that users can develop an understanding of their choices, receive feedback about the implications of their interactions with the system, and that their actions are available and understandable to others, is a huge task. In the 'mixed media case', architectural students' direct, bodily engagement with artefacts makes configuring a (publicly) visible, hence accountable activity. Students experimented with changing the properties of a model, by applying colour, inserting movement and context, and varying its dimension in relation to other objects in the physical space. Using barcodes as a single interaction mechanism proved to be a good decision. The barcode technology was easy to understand and transparent. Barcodes were e.g. integrated into CAD drawings, cut out, glued onto posters or models, annotated (students e.g. created their own manuals), distributed in space, used for configuring input (images) – output (projection surfaces) associations and for setting keywords (in a tangible way).



We also could see that the potential of physical interfaces (in this case barcodes on posters, models, and other parts of the physical environment) reaches beyond 'mere embodiment'. They provide people with the means for producing configurations that change spatiality, interactivity, and physical landscape in ways that help experience, explore, present, and perform.

1.4 Configuring as part of technology use

Providing organizational resources for configuring as part of everyday work practice is another challenge. Connected with this are issues like how much work has to be done to configure and re-configure; can it be done by end-users or is this a work of specialized personnel and are these people available, and so forth? In the 'mixed media case', configuring was part of the pedagogy. Students were asked to continuously transform and 're-program' familiar settings. *Configuring* was encouraged, and was hardly distinguishable from proper *use*. Students' configuring their workspace, configuring and selecting textures to be 'painted' onto their models, configuring input and output devices, and so forth happened as part of the design process.

2 Configurability

We see configurability as being intricately linked to the fact that in an evolving environment the boundaries of activities are continuously moving. The introduction of new technologies necessarily involves reconfiguring—often the decision to use new technologies in a particular work setting is undertaken specifically to serve as a catalyst for altering or re-configuring work practices. While this fact may be purposely built into learning situations (such as in our design students case), it may be ignored, hence poorly supported, in other cases (such as the wireless call system case). Here, possibilities for configuration of the

system were shrouded in a lack of transparency about the range of configurable features. Local adaptation and configuration (e.g. that the alarm that signalled when a bed was being unplugged from the wall could be turned off) were at odds with core organizational requirements. Hierarchical organizational relations and the multiplicity of stakeholders deterred configurability. Our conclusion from this is that *configuring organizational relations* is an important resource in supporting highly configurable technologies.

Another conclusion from our study is the need to take a step further and unravel different meanings of configuration and of the contexts in which configuration takes place. In doing this, we are aware that we use different terms without being able to make a clear distinction between them:

- We use the term *configuring* for all activities that allow users to assemble available resources to handle their tasks. Rodden *et al* (2004) discuss configurability in relation to two organizational features of interaction: placement – how to take account of the local spatial organization of activities; and assembly – how to facilitate the configuration and reconfiguration of artefacts and media. Newman et al. (2002) introduce the term *'recombinant computing'*, which builds on tools (protocols and techniques) and interfaces for making components interoperable.
- *Customizing* denotes activities that are necessary to make a device or system function in a particular environment, down to very small details that matter to users. Many systems are designed for a certain degree of customization, offering specific features (Andriessen et al. 2003).
- We talk of *tailoring* in the case that *software* is customized to the needs of users. This often is to do with defining, sharing, and distributing 'standards' (forms, macros) within the organization. It may involve building/modifying buttons, to writing macros, and programming on the PC, manipulating otherwise 'invisible' codes (Trigg and Bødker 1994).
- These activities may involve *appropriation work* – users fitting a (set of) devices, a piece of software to their needs. Bossen and Dalsgaard (2005) distinguish between weak appropriation (flexible appropriation at the level of the artefact) and strong appropriation (changing the technology in ways that go beyond the intention of the designers).

Obviously the architectural students' configuring options differ from those open to the hospital staff. This is partly to do with the nature of design work that makes particular configuring options relevant and attractive, partly with the technologies that were made available to students (at a working prototype level), partly with the fact that their work environment is small, open and experimental.

Configuring the wireless call systems happens within a complex organization and it involves different internal and external (vendor) stakeholders. It may involve different levels – technical (system, component, device) and organizational (ward, IT department and so forth) - and happen at different stages

of the system development and implementation process. Each type of configuring requires a specific set of skills, depending on the activity, and attention to different resources necessary to successfully carry out configuration in relation to each particular type of configuration.

3 Placement - taking account of the local spatial organization of activities

For example, we observed how being able to configure the work environment for a diversity of uses, from solitary work to group discussions, performing, presenting, and building models, is an important aspect of design work. While some spatial set-ups lend themselves to students' quick and easy reconfiguring, others may require high degrees of precision, hence time (such as, for example, fixing the position of beamers and projection surfaces in relation to an artefact). Space on a hospital unit needs to be reconfigured in order to support the use of the wireless call system. A place for storing the devices needed to be created, as well as a place for charging handset batteries, which required the cooperation of building maintenance staff.

4 Assembly – configuring artefacts and media

Students configured input and output devices in support of their design activities and combined technical and spatial components to explore new ways of browsing and searching multimedia database. This is an activity that takes place as part of ongoing work but requires some preparation. At the beginning of each work shift each staff member had to configure the handset they would use for the day. This entailed linking the rooms/ beds they had been assigned to the handset they would carry for the day, and determining which staff member would serve as their backup for calls that went unanswered during breaks and busy times.

5 Customization – appropriation in the use context

Customizing the length of time the wireless handset rings before going to the backup nurse is an example of a restricted set of features to be adjusted by end users; another one would be varying the colour of different types of alarms (associated with the patient bed, the patient's body, the bathroom and so forth). This may happen from time to time to adjust for changes of work practice, staffing level and so forth. In the hierarchical environment of the hospital unit, engagement in this level of customisation was restricted to management staff, who

sought input from representative users. Enacting this particular feature required that a vendor representative alter software logs, and, although this work could have theoretically been undertaken by hospital staff, in the case we observed, it was reserved for the vendor.

6 Customization – fitting the application to a particular setting

Customizing the sign-on and sign-off process to different devices (the unit's computer, its wireless phone system console). This is an activity that takes place at implementation, involving the vendor, technical staff and users. It usually requires some programming (tailoring). In an environment where each implementation integrates a slightly different constellation of devices together with the specifics of what is actually possible in a given setting determined by a myriad of factors, little or no documentation may exist about the exact scope of customization that is possible.

7 Getting an integrated system to work

An example would be getting the wireless network on the ward to interface with the wired phone system. This happens at the pre-pilot or pre-implementation level, with technicians, IT staff and gatekeepers for component systems solving the complexity or messiness of particular implementations.

8 Getting a component system to work as part of an integrated system

Setting up the phone consoles at the unit desks and setting up the wireless LAN. This is part of the initial equipment set up or modification required as multiple components are integrated. Most of this work would be carried out by technicians. Decisions made during component set up may need to be altered when individual components are brought into an integrated network.

9 Elaborating design patterns

In the design setting, users combined technical and spatial components to explore new ways of browsing and searching. Here we talk of creating a *design pattern* - a particular combination of devices and services which may serve as a relevant exemplar from which mixed media environments for other tasks and in other

settings can evolve (Ehn et al. 2004). A design pattern may illustrate what is possible conceptually, however, achieving a specific design pattern in a new setting may rest on resolving configuration issues identified above in new ways. Different configuration activities may be required from one setting to the next in order to achieve the same—or a similar—result.

10 Creating a sustainable structure for implementation, use and configurability

Integrating technological components in new ways can pose challenges to existing roles, and can both alter and introduce new means of responsibility and accountability. In the wireless call system case this would include clarifying who will follow up with problems related to batteries; or defining responsibilities as well as options for staff in case of problems and breakdowns. One example is drawn from a phenomenon we came to refer to as 'phantom calls' that were not taken very seriously by the vendor representative who said they would probably disappear again. But eventually he was forced to check the system's software logs. He found an error in the configuration, which made the system attempt to call the phones up to one hundred times if they had been receiving calls while they were switched off (as happened frequently when staff took breaks). Creating such a sustainable structure primarily involves management.

Configuring systems so that they work together and offer users a seamless use experience requires that configuration occur on numerous levels, which we have attempted to outline above. Each type of configuration involves different constellations of actors, who come together in different groupings, governed perhaps by different interactional norms or relations, which also must be addressed or accounted for in efforts to sustain highly configurable systems.

11 References:

Andriessen, J., Hettinga M. and Wulf V. (2003). Evolving Use of Groupware. Special Issue of Computer Supported Cooperative Work 12 (2).

Balka, E., Wagner I. and Bruun Jensen, C. (2005). Reconfiguring Critical Computing in an Era of Configurability. The Fourth Decennial Aarhus Conference Critical Computing- Between Sense and Sensibility, Aarhus, DK, ACM.

Binder, T., De Michelis, G., Jacucci, G., Matcovic, K., Psik, T.and Wagner, I. (2004). Supporting Configurability in a Tangibly Augmented Environment for Design Students. Personal and Ubiquitous Computing 8, (2004), 310-325.

Bossen C. and Dalsgaard P. (2005). Conceptualization and appropriation: the Evolving Use of a Collaborariuve Knowledge Management System. The Fourth Decennial Aarhus Conference Critical Computing- Between Sense and Sensibility,, Aarhus, DK, ACM.

Ehn, P. Ed (2004). ATELIER IST–2001-33064 Architecture and Technology for Inspirational Learning Environments. Final report. Malmö School of Art and Communication.

Newman, M.W., Sedivy, J. et al. (2002). Designing for Serendipity: Supporting End-User Configuration of Ubiquitous Computing Environments. London, ACM (2002).

Rodden, T., Crabtree, A. et al. (2004) Between the Dazzle of a New Building and Its Eventual Corpse: Assembling the Ubiquitous Home. DIS Cambridge, Mass, ACM (2004).

Trigg, R. and Bødker, S. From Implementation to Design: Tailoring and the Emergence of Systematization in Cscw. CSCW, Chapel Hill, NC, ACM (2004).

Groupware Construction with the Oregon Software Development Process

Till Schümmer
Computer Science Department,
FernUniversität in Hagen, Germany
till.schuemmer@fernuni-hagen.de

Abstract. This position paper describes how appropriation of groupware systems takes place in the Oregon Software Development Process. It calls for a special focus on high-level groupware patterns as communicative and educational means for end-users and developers.

1 Introduction

The development of groupware applications is still a difficult task. One main reason is that both developers and end-users are not aware of possible solutions for supporting group interaction. A second aspect is that group interaction involves many users, which makes the definition of requirements difficult. From that perspective, end-user involvement is one of the most important but much too often neglected issues in groupware development.

Developers and end-users need to be supported in the requirements elicitation and the design of tools that help to satisfy these requirements.

I argue to approach this need from three different perspectives: (1) the abstract view on development that is not bound to the specific nature of the developed artifact, (2) the software development perspective that focuses on processes and tools for the development of software, and (3) the groupware development perspective, which brings together software development aspects with social

aspects and thus addresses the task of groupware development from a socio-technical view.

The basis for my abstract view on design is the holistic view, as it is currently propagated by many designers, especially by Christopher Alexander (Alexander 2003). In his view, a holistic approach to design has to be combined with an evolutionary process and focus on end-user education in order to empower the enduser to play an active role in the development process. Together with Heidegger (Heidegger 1927), Alexander puts special attention on the situatedness of design (Alexander, Silverstein, Angel, Ishikawa & Abrams 1980). Users should reflect on their activities whenever their flow of action is disrupted. This situated reflection (as it was also propagated by (Schön 1983)) provides the best access to the requirements and supports solutions that meet the requirements.

Situatedness requires that the end-user is heavily involved in the design process. To reach this level of control, they have to be educated regarding possible good practices – the patterns – for reshaping their environment. In *A Timeless Way of Building*, Alexander defined a pattern as a morphological law that explains how to design an artifact in order to solve a problem in a specific context (Alexander 1979). The goal of describing best practices with patterns is that end-users are empowered to shape their environment in a professional manner.

These core requirements relate to current software development processes. Especially evolutionary processes (Malotaux 2001), agile methods (Boehm & Turner 2004), and participatory design approaches (Muller & Kuhn 1993) propose an interaction between developers and end-users that gives the control over the built artifact back to the end-users.

Relevant groupware development processes that focus on end-user involvement include the extended eXtreme Programming process (Rittenbruch, McEwan, Ward, Mansfield & Bartenstein. 2002) that focuses on involving the user community in planning and development (instead of just a single customer representative in XP) or an iterative process based on the STEPS process model (Floyd 1993) that puts special attention tailoring during system use (Wulf & Rohde 1995). The latter demands that the end-user adapts the system in order to meet requirements that evolve from reflection in action. Other groupware processes, like SER (Fischer, Grudin, McCall, Ostwald, Redmiles, Reeves & Shipman 2001) put a special focus on sharing of design knowledge. But still they do not help to shape the knowledge in a way that it can be easily used by end-users.

In summary, the field still lacks a groupware development process that supports end-users in a way so that they can learn, plan, implement, modify, and share appropriations based on best practices.

The Oregon Software Development Process presented in this paper tries to fill this gap. It fosters end-user participation, evolutionary growth, and reuse and

exchange of design knowledge. Patterns play an important role in the whole process since they are the means for communicating design knowledge between users, developers, and between users and developers. Expert Involvement 3 8 2 6 11 5 7 9 User Involvement 4 1 Analysis of forces Planning Conflicting forces Design Pattern driven tailoring design - high level patterns - Pattern driven groupware design - low level patterns - Scenarios Implementation Patterns & mockups Pattern driven groupware tailoring Test & Usage Usage with diagnosis & reflection - health map - functional tests conceptional iteration development iteration tailoring iteration Initial forces Groupware development Discussion 10 12

While the complete process has been described before (Schümmer & Slagter 2004), this paper discusses how appropriation work of end-users is supported by the process. After giving a short introduction to the core practices of OSDP, I will focus on tailoring iterations that have the goal of appropriating the groupware system during use.

2 The Oregon Software Development Process

The Oregon Software Development Process (OSDP) intends to foster end-user participation, pattern-oriented transfer of design knowledge, piecemeal growth of the system under development in the form of short iterations, and frequent diagnosis or reflection that lead to an improved application.

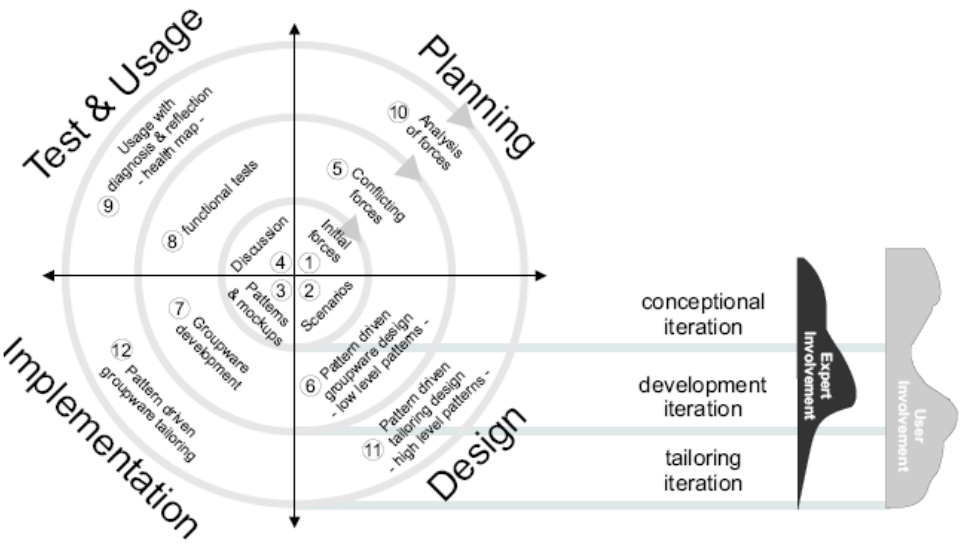


Figure 1: The Oregon Software Development Process.

OSDP structures the development of the application in three kinds of iterations: conceptual iterations, development iterations, and tailoring iterations. Figure 1 shows these iterations denoted by the three circles.

– In conceptual iterations users and developers collaborate in scenario interest groups (SIGs) in order to collect and refine scenarios of system use. They make use of groupware patterns to inform their mapping of social processes to a groupware setting.

– Development iterations focus on the implementation of the scenarios from conceptual iterations. Users and developers collaborate to create task descriptions and to relate these descriptions to groupware patterns. These descriptions are collected and estimated regarding their costs and benefits. The development team continuously focuses on the most important tasks in order to implement the most critical aspects first. Important aspects of the development iteration include that the list of tasks – the backlog – is made public in the user community and that the task wishes of different users or SIGs are ordered and merged.

– In tailoring iterations users reflect on their groupware use and fix conflicting forces by adapting the groupware system. Pattern scouts encourage the users to share their adaptations with other users. If the adaptation matured it is formulated as a pattern and added to the community's pattern language.

While the first two iterations mainly take place before the system is in use, the tailoring iteration describes how the system is appropriated during its use. The following section will thus have a closer look at this kind of iterations.

3 Tailoring Iterations Close Up

In the *tailoring iteration* end-users use the application for the desired purpose. While using the system, end-users with pattern-based design knowledge are encouraged to reflect on their activities whenever they encounter a *breakdown* (9 in fig. 1). A breakdown leads to an entry in the groupware's health map (in the simple case, a note that a specific group need could not be satisfied with the groupware system). In cases where the user does not detect this breakdown (i.e., if the user feels uncomfortable but thinks that this feeling cannot be changed), an evaluation user (as proposed by cf. (Rittenbruch et al. 2002)) can expose the breakdown, discuss it with the user, and initiate a reflection process together with the user.

Users then take a closer look at the detected shortcoming. First, they *analyze the forces* that are in conflict (10). High-level groupware patterns help in this process by describing frequently occurring issues, the various forces, and a proven solution in a way that is appropriate for tailoring end-users.

The solution provided by the high level patterns informs the successive *groupware tailoring design* (11) and the execution of the *tailoring* (12). Tailoring actions can take place at different levels. In content level tailoring, the users change the artifacts that are managed by the groupware system in order to solve the problem. At this level, the pattern assists the user in using a tool. At the functional level, users appropriate the functionality provided by the tool. They activate needed functions and deactivate functions that are in the way. At the component level, the users perform more extensive tailoring actions: they compose functional groupware components in order to create new configurations of applications that support the team in an unanticipated way.

To support tailoring at a group level, a *pattern scout* looks for solutions that work well. As the evaluation user, the pattern scout observes users in the system with the goal of finding recurring successful system use. The found best practice is then discussed with the users and documented in the pattern format. Such new best practices then find their way in the pattern catalogue. Note that these patterns are in most cases very domain specific (e.g. patterns for supporting customer relationship management in the context of a support system). These patterns will, however, be used most frequently in the user community since they are appropriated for the interaction that typically takes place in the community.

4 Experiences

The OSDP has been applied in the CURE project in which a collaborative learning environment for the Distance University of Hagen was developed and installed. The project showed that users and developers ware able to follow the proposed steps and that the patterns play an important role in the communication between all stakeholders. One could observe that users were able to play an active role in all phases of the process. They shaped their environment and developed new best practices (e.g., practices to support literature research). First of these practices were captured as patterns.

The opportunity to tailor motivated users to reflect on their activities. Users created new domain-specific patterns and provided implementations using the tailoring mechanisms of CURE. Some users referred to patterns to support their tailoring, others based their tailoring operations on intuition. This may be one reason why many users reported that tailoring was still difficult.

Propagating the collection of domain-specific patterns to the users is thus still a challenging task in order to educate the end-users and to support better tailoring actions. In cases where patterns were used the forces were also made explicit and discussed between the tailoring users. In other cases, the forces did not play an explicit role.

5 Conclusions

This paper presented parts of the Oregon Software Development Process that focus on the appropriation of groupware systems at runtime. The tailoring iterations of OSDP structure end-user activities in order to support reflection on system use, capturing of best practices, and sharing of expert-user's appropriations.

Most important tools in the OSDP are patterns that help to capture design knowledge in a way that is easy to understand for end-users as well as groupware developers. These patterns evolve during system use, steer the tailoring of the system, and capture evolving best practices of system use.

However, the OSDP is no silver bullet. Patterns with a high level of abstraction can often be implemented by tailoring the application, but low level patterns still require development and design expertise by the involved software developers.

6 References

Alexander, C. (1979), *The timeless way of building*, Oxford University Press, New York.

Alexander, C. (2003), *The phenomenom of life*, Vol. 1 of *The nature of order*, Center for Environmental Structure, Berkeley, California, USA.

Alexander, C., Silverstein, M., Angel, S., Ishikawa, S. & Abrams, D. (1980), *The Oregon Experiment*, Oxford University Press, New York.

Boehm, B. & Turner, R. (2004), *Balancing Agility and Discipline – A Guide for the Perplexed*, Addison Wesley, Boston, MA.

Fischer, G., Grudin, J., McCall, R., Ostwald, J., Redmiles, D., Reeves, B. & Shipman, F. (2001), Seeding, evolutionary growth and reseeding: The incremental development of collaborative design environments, *in* G. Olson, T. Malone & J. Smith, eds, 'Coordination Theory and Collaboration Technology', Lawrence Erlbaum Associates, pp. 447–472. *http://www.ics.uci.edu/ redmiles/publications/B002-FGMcC01.pdf

Floyd, C. (1993), 'Steps – a methodical approach to pd', *Commun. ACM* **36**(6), 83.

Heidegger, M. (1927), *Sein und Zeit*, 17 (1993) edn, Niemeyer, T¨ubingen.

Malotaux, N. (2001), Evolutionary development methods, *in* 'Proceedings of PROGRESS 2001', Technology Foundation (STW), Utrecht, the Netherlands. *http://www.stw.nl/progress2001/proc2001/malotaux.pdf

Muller, M. J. & Kuhn, S. (1993), 'Participatory design', *Communications of the ACM* **36**(6), 24–28.

Rittenbruch, M., McEwan, G., Ward, N., Mansfield, T. & Bartenstein., D. (2002), Extreme participation -moving extreme programming towards participatory design., *in* T. Binder, J. Gregory, &

I. Wagner, eds, 'Participation and Design: Inquiring Into the Poltics, Contexts and Practices of Collaborative Design Work – PDC 2002 Proceedings of the Participatory Design Conference', Malmo, Sweden.

Schön, D. A. (1983), *The Reflective Practictioner: How Professionals Think in Action*, Basic Books, New York.

Schümmer, T. & Slagter, R. (2004), The oregon software development process, *in* 'Proceedings of XP2004'.

Wulf, V. & Rohde, M. (1995), Towards an integrated organization and technology development, *in* 'DIS '95: Proceedings of the conference on Designing interactive systems', ACM Press, pp. 55–64.

Radical Appropriation: The Configurations of Wireless Networking in a Community Group

David W. McDonald
The Information School University of Washington
dwmc@u.washington.edu

Abstract. Forms of appropriating activity range from adopting a technology largely as it is to radically transforming it to make it perform some other function. This field study describes how a wireless community group appropriated commodity wireless networking and transformed it to serve their goals. The study characterizes several configurations which were promoted by community leaders, the varying drawbacks which each led to a new, subsequent, configuration. The results identify a set of design considerations for hardware devices that seem to facilitate appropriation.

1 Introduction

A hack, in the traditional sense of the word, is an appropriation that illustrates some deeper understanding of a technical system. The reconfiguring of a device, making a device specifically designed to do one thing into a device that does something else, has slowly become more commonplace. At this time no one has made an automobile Anti-Lock Braking System play MP3s, but some MP3 players have been reconfigured as Linux computers (Leach, Carne et al. 2003).

Wireless networking, WiFi, 802.11a/b/g, has been fertile ground for a range of appropriation that illustrates creativity and in-depth understanding of a technical system. War driving (Byers and Kormann 2003), packet sniffing, breaking WEP (Wireless Encryption Protocol) (Fluhrer, Mantin et al. 2001; Rager 2001),

represent a range of sophisticated hacking activities in which almost anyone can participate given access to the right software. The software lowers the barrier to participation in a technically sophisticated activity, but someone had to create that software first. Someone, some group, had to take their deep understanding of the technical system and embody it in a software artifact so that less sophisticated participants could benefit.

This study describes appropriation activity in a wireless community group, the Northwest Wireless Group (NWG). The focus is on the radical appropriation of commodity wireless equipment to create a wireless backbone and wireless community access. The fieldwork describes how individuals come together to appropriate wireless technology and solve difficult infrastructure construction and maintenance problems. The results focus on a set of design considerations that seem to facilitate appropriation of hardware devices.

Appropriation is commonly framed as a type of adoption of a piece of software or a complex system. The concept of adoption is described and studied in the computing and information technology literature (e.g. (Markus and Connolly 1990; Francik, Rudman et al. 1991; Orlikowski 1992; Levine and Rossmoore 1993; Kraut, Cool et al. 1994)) and the results can often be reexamined as appropriating activity. Studies of the organizational implementation (deployment) of complex systems like Group Decision Support Systems (GDSS) or Enterprise Resource Planning (ERP), have used 'appropriation' as a term to describe activities of users which are outside a normative model of system usage (Orlikowski and Robey 1991; Galegher and Kraut 1992; DeSanctis 1993; DeSanctis and Poole 1994; Olesen and Myers 1999). Often these are uses outside the normative model of work activity as understood by the system designers. Using appropriation in this way is powerful because the appropriation activities illustrate how users bridge the gap between their actual needs and the needs as implemented in the system.

This paper first describes the primary difference between the default wireless networking context and how the community group reshaped the context by defining a different networking model. This model frames three technical configurations that the community developed over several years. These configurations illustrate a number of difficulties that must be overcome by individuals who appropriate hardware devices.

2 Appropriating a Context: Reconceptualizing WiFi

The wireless industry has largely conceptualized WiFi as a localized service. Generic access points simplify the redistribution of Internet service with built-in software. However, the broader wireless community recognizes that WiFi

technology has the potential to do more than provide simple hotspot service. By reconceptualizing how WiFi technology can be used, a larger network, not just hotspots, can be constructed. A wireless Metropolitan Area Network (MAN) can be developed by connecting nodes of a network through wireless connections (Flickenger 2001), creating a wireless backbone. The Northwest Wireless Group (NWG) is one group that led early efforts to develop infrastructure and software to achieve a wireless MAN.

The insight that a wireless MAN can be constructed from default WiFi components is an appropriation of the context of wireless networking. It illustrates a gap between what users want to be able to do with wireless equipment relative to the default design for wireless networking as promoted by the equipment manufacturers. In this section, we illustrate the difference between the default WiFi context and the context created by NWG members.

2.1 The WiFi Backbone Model

The challenge in reconceptualizing wireless networking is how to connect a large number of standard WiFi nodes without using wires. This requires moving away from the idea that a node in the network is composed of a single piece of equipment with a single WiFi compliant radio. Nodes in an NWG network must be more complex.

A node for the NWG network is composed of at least four pieces of equipment; a computer, two access points and a directional antenna. A small computer serves to route data and monitor the node. These computers often run a version of Linux. In a basic node, one AP is used to provide local (omni-directional) connectivity for wireless devices in the physical vicinity. A second AP provides a directional connection to another node in the wireless network. The directional connection is provided by directional antennas at each end of a link.

A minimal node provides a limited type of connectivity. That is, only supporting a single directional backbone link creates network design problems. Thus the reconceptualized network model supports nodes of differing complexity, with different levels of connectivity to the network. Figure 1 is a diagram of how wireless nodes are connected through directional RF links. This figure illustrates the different levels of connectivity and the general NWG network model.

While the WiFi backbone model is conceptually possible, a fundamental challenge in implementing this network architecture was to develop a model for the basic NWG node. The WiFi standard included and ‘ad-hoc’ or ‘peer-to-peer’ mode which could provide the needed directional link, but in practice different manufacturers implemented this feature slightly differently. As a result, a node model, or node configuration, could help galvanize participation around equipment that would interoperate and simplify how nodes were connected.

3 Radical Appropriation: Finding the Right Configuration

The NWG community explored a number of possible node configurations over the past few years. The four configurations that will be discussed generally overlapped with each other. The community never focused solely on a single configuration. Often configurations were explored and fielded as experiments to test both reliability and to provide service. Each possible configuration was promoted to the general population of participants. In most cases NWG members purchased equipment, explored the capabilities and contributed some way to making a working solution. In two of the cases, severe hardware or software constraints resulted in the configurations being abandoned.

3.1 The Airport/Orinoco RG-1000 Configuration (c. 2001-2002)

In 2000 Apple Computer introduced the Airport wireless access point to the public and initiated the low-cost public acceptance of the 802.11b "WiFi" wireless standard. Wireless networking had been available in various forms prior to the introduction of the Airport, but equipment was expensive and not well supported by the most prevalent computer operating systems. The Airport included a 56K modem that could be configured with NAT (Network Address Translation) and bridging so that several computers on the wireless connection could also use Internet service through the modem. With the introduction of Airport almost anyone could quickly set up a Local Area Network (LAN) in the home or office.

Configuring the Airport[1] to boot a different operating system is not trivial.[2] In particular, the Airport needed support from a second computer running Linux or similar Unix variant. This host machine needed to have a Network File System (NFS) partition created that the Airport would use for a remote boot and for a dedicated file system. On this NFS partition a special '.nbi' (Net Boot Image) would be placed where it would be available for the Airport to read. Lastly, the firmware in the Airport needed to be "flashed" (semi-permanently modified) with new firmware so that the device would boot from the appropriately named .nbi file off the NFS partition of the host machine. A Linux/Unix system administrator with two or more years of experience would find this type of configuration relatively simple. An average user would find this configuration difficult.

---

[1] In the following discussion "Airport" is used in the general sense of any access point that relied on the same internal hardware.

[2] The initial insight about the Airport and the first effort to get Linux to run on the device is credited to Till Straumann. See http://www.slac.stanford.edu/~strauman/pers/airport/airport.html for more details on how to install and configure Linux for an Airport.

The Airport/Orinoco RG-1000 configuration was problematic because it required an additional machine to host the remote boot. But as well, the device had very limited memory. With only 512K of flash memory and 4MB RAM the device was barely able to run Linux. Routing tables, simple network monitoring code and other configuration data all take space in the limited memory. Despite efforts to strip Linux to the bare minimum, memory problems were common, often causing directional links to fail.

3.2 Pebble/Soekris v.1 Configuration (c. 2002-2003)

Frustrated with the installation problems and physical limitations of the Airport/RG-1000 the broader wireless community began exploring solutions using Linux and embedded computers. The Pebble[1] Linux distribution is conceptually similar to that of the Airport Linux; a minimalist distribution designed to be able to run on a range of small computers with limited memory, disk space, and with at least one wireless card.

As NWG became frustrated with the limitations of the Airport/RG-1000 configuration, they began developing nodes around embedded computers. Soekris Engineering developed two computers, the net4511 and the net4521, which were well engineered and highly capable. A single net4521 could potentially support up to four wireless connections as well as an Ethernet link to the Internet. Conceptually, Pebble on a Soekris computer provided an ideal NWG node; potentially a "Class A" node in a single box (see Figure 1).

The Pebble/Soekris configuration was fielded by several participants and problems started to emerge. In general, the Pebble/Soekris configuration was more stable and reliable than the older Airport/RG-1000 configuration. With this reliability, people were more willing to deploy NWG nodes in locations with limited access like roofs and other outdoor locations. NWG participants began exploring how to inexpensively weatherize node equipment. Attempts that used Tupperware, silicone glue, shrink wrap, and shrink tubing all met with varying degrees of success. While the net4521 can support up to three wireless cards, this is not practical because the amount of interference caused by RF bleed from the cards and the pigtail connectors seriously decreases overall throughput.

While the cost of a complete Pebble/Soekris node was less than commercial grade equipment, the combination of cost and complexity of set-up limited the number of NWG participants who would select this equipment. As well, a third alternative, based on an inexpensive consumer grade access point, diverted attention and effort from the Pebble/Soekris configuration.

---

[1] Terry Schmidt, a founder of NYCWireless in New York, initiated and led the Pebble project through several early and critical distributions. The current Pebble distribution is available at http://www.nycwireless.net/pebble/

3.3 WRT54g Configuration (c. 2003-2004)

In 2003 the Linksys Corporation began distribution of a new wireless access point called the WRT54g. Linksys had been an active manufacturer of consumer grade 802.11b equipment that was relatively simple to install and quite popular with the average consumer. The WRT54g was Lynksys' first attempt at an 802.11g access point. As a consumer grade AP the WRT54g was relatively cheap and provided backward compatibility with the 802.11b equipment that was already widely deployed. Thus the WRT54g represented a natural upgrade path from one generation of wireless standard to the next.[1]

Like many APs and routers, the WRTs use a built-in web server and a set of web pages to facilitate set-up and monitoring. Some of the WRT pages collect data from web form fields and pass the data directly to a Linux command line with little or no error checking. Because most command line shells support multiple commands per line, this oversight allowed a user to enter a valid parameter followed by a command separator and a subsequent set of commands. One of the web pages allowed the user to pass parameters to the 'ping' command and provided a large text field area to view the response. With this simple exploit and a way to see the results of a general command response, the community systematically explored the version of Linux and the available command tool set distributed in the WRT. The community quickly realized that the WRTs had some specific limitations, but it had flash memory and that could be changed.

The broader wireless community began efforts to create a custom version of the WRT firmware that would include the tools that were need to remake the WRT into more than a simple AP. Through systematic exploration and some trial and error the community developed wrt54g_tools, a set of software tools specifically for creating a valid firmware image. Despite this success, two problems remained. First, Linksys had used a GPL code base and was reluctant to release the code they had developed. Second, the WRT used a Broadcom wireless card which was designed for the OEM (Original Equipment Manufactuer) marked and Broadcom has never released the drivers for the wireless card.

Linksys eventually released their code to the community, but the WRT54g configuration did not take off. The use of a Broadcom OEM wireless card with proprietary drivers meant that the community was prevented from understanding the underlying hardware well enough to appropriate its latent functionality. Also, equipment manufacturers are constantly looking for ways to simplify their design and produce a product more cheaply. In the case of the WRT, each product revision was like dealing with a new unique piece of equipment. Changes to the hardware, introduced by a manufacturer can be handled in software, such that to an outsider, the product looks the same, the default functionality is the same, the

[1] The relatively common availability of 802.11b and 802.11g equipment is largely a function of the fact that they operate in the same frequency range that simplifies the engineering of the hardware (transmitters, receivers, antennae). The other WiFi standard, 802.11a, has never achieved broad public acceptance.

user interface is the same, while the device hardware could be completely different. In the case of the WRT changes happened frequently and were dramatic enough that small revisions in the underlying hardware made the community based firmware incompatible from one board revision to the next. For the average user, these differences are difficult to explain and an incomplete understanding can result in a modified device that is completely useless.

4 Facilitating Appropriation

The story of NWG is fundamentally a story of collaborative appropriation. It is collaborative in the way the technology is systematically explored and exploited. The collaboration spans the larger wireless community, the activities of the NWG group itself, and the small scale collaboration of individual members as they attempt to build and install network nodes. This study has focused on the technical trajectory of appropriation and the systematic exploration and reconfiguration of WiFi network equipment.

This story illustrates a number of key aspects for the design of technology that facilitates appropriation. In particular, the story highlights how appropriation is possible when devices have latent functionality that is identified and exploited by users. This latent functionality is easier to identify and exploit when users have access to patterns that illustrate aspects of the device design. Lastly, appropriation is possible when there is some configuration stability in the device. This stability allows users to communicate the appropriation to others and know that the modifications will most likely work on another's device.

5 Acknowledgments

This research would not have been possible without the NWG community and the participants' willingness to talk about the project. This research was supported by the University of Washington, Royalty Research Fund Grant 65-2805 and by contributions from Intel Corporation, Intel Research, Seattle.

6 References

Byers, S. and D. Kormann (2003). "802.11b Access Point Mapping." Communications of the ACM 46(5): 41-46.

DeSanctis, G. (1993). Shifting Foundations in Group Support Systems Research. Group Support Systems: New Perspectives.

L. M. Jessup and J. S. Valacich. New York, NY, Macmillan. DeSanctis, G. and M. S. Poole (1994). "Capturing the Complexity in Advanced Technology Use: Adaptive Structuration Theory." Organization Science 5(2): 121-147.

Flickenger, R. (2001). Building Wireless Community Networks, O'Reilly & Associates.

Fluhrer, S., I. Mantin, et al. (2001). Weaknesses in the Key Scheduling Algorithm of RC4. Lecture Notes in Computer Science, Springer-Verlag. 2259: 1-24.

Francik, E., S. E. Rudman, et al. (1991). "Putting Innovation to Work: Adoption Strategies for Multimedia Communication Systems." Communications of the ACM 34(12): 53 - 63.

Galegher, J. and R. E. Kraut (1992). Computer-Mediated Communication and Collaborative Writing: Media Influence and Adaptation to Communication Constraints. Proceedings of the 1992 ACM Conference on Computer-Supported Cooperative Work (CSCW '92), ACM Press.

Kraut, R. E., C. Cool, et al. (1994). Life and Death of New Technology: Task, Utility and Social Influences on the Use of a Communication Medium. The 1994 ACM Conference on Computer-Supported Cooperative Work (CSCW '94), Chapel Hill, NC, ACM Press.

Leach, B., D. Carne, et al. (2003). iPodLinux, iPodLinux Project.

Levine, H. G. and D. Rossmoore (1993). "Diagnosing the Human Threats to Information Technology Implementation: A Missing Factor in Systems Analysis Illustrated in a Case Study." Journal of Management Information Systems 10(2): 55-73.

Markus, M. L. and T. Connolly (1990). Why CSCW Applications Fail: Problems in the Adoption of Interdependent Work Tools. The 1990 ACM Conference on Computer-Supported Cooperative Work (CSCW '90).

Olesen, K. and M. D. Myers (1999). "Trying to Improve Communication and Collaboration with Information Technology: An Action Research Project which Failed." Information, Technology and People 12(4): 317-332.

Orlikowski, W. J. (1992). Learning from Notes: Organizational Issues in Groupware Implementation. The 1994 ACM Conference on Computer-Supported Cooperative Work (CSCW '92).

Orlikowski, W. J. and D. Robey (1991). "Information Technology and the Structuring of Organizations." Information Systems Research 2(2): 143-169.

Rager, A. (2001). WEPCrack.

‘Reflective User’ in Practice: Explorations from two cases

Samuli Pekkola
Department of Computer Science and Information Systems
PO Box 35 (Agora), 40014 University of Jyväskylä, Finland
samuli@cc.jyu.fi

1 Introduction

Information systems development (ISD) methodologies are numerous (Iivari et al. 2001). Yet, they do not address the change in the information system itself when it is introduced into an organization, or when the organization or its environment changes (Lyytinen 1986; de Michelis et al. 1998). It can even be said that the information systems development begins when it is introduced into an organization (Nurminen and Forsman 1994).

One reason for the deficiencies in ISD methods and obscurities in systems development is the difficulty of anticipating its use in the working environment (c.f. (Robinson 1993). As a consequence, it is very difficult for systems developers to create complete use cases or make appropriate design decisions. Instead, they have to rely on end-users and consider them as the sources of information and most important factors in successful systems development (c.f. Lynch and Gregor 2004). In other words, an input from the user is used to validate the appropriateness of the design decisions.

But this is not an easy task. Evaluating the appropriateness is highly subjective – the same design can be perceived correct for one, and incorrect for the other. The situation is even more difficult if the users do not know when, how, in which context, and with whom they would use the system – which is often the case with research prototypes. In this position paper, I present explorations from two different cases where the users’ suggestions for the features of CSCW systems, and the

feedback for the appropriateness of the design decisions are found problematic to identify and articulate. The first case is composed of two research projects where CSCW applications were developed to support communication and cooperation within an organization. The second case is about a research project where tools for inter-organizational cooperation and communication were investigated.

2 Case 1: Support for intra-organizational activities

The support for intra-organizational communication and collaboration was studied in two consecutive research projects: VIVA (1998-2001) and MADE (2002-2003). In both projects, the aim was to develop a system to support the work of mechanical engineers of a manufacturing company. The engineers worked with others that were located in another site of the company, in their customer's site, or were traveling between them. Meanwhile, because of the context and the risks for enormous financial losses, it was essential that appropriate engineers can be contacted and information shared regardless of their location. The engineers had worked this way for years sharing information through email and telephone and, importantly to the developers, recognizing the problems in real-time communication. Consequently they had a lot of expectations from the future system, which they expected to be able to solve many problems of email and telephone. The need was very practical and very concrete – to share information and communicate with others in real-time. The projects were established for answer this need; first VIVA for PC's, and then MADE for mobile terminals.

For the applications, there was no particular purpose or context for which they were targeted (apart from information sharing and communication). The idea behind there was to offer multiple communication and collaboration tools, e.g. text chat, audio, shared whiteboard, shared text editor, file transfer and short messages so that the users could choose the device, or a set of many that they found the most appropriate at any particular moment (c.f. Pekkola 2003; Pekkola et al. 2003). Both VIVA and MADE systems succeeded in term of achieving the goal.

The systems development lifecycles followed evolutionary prototyping approach (McConnell 1996). Seven different prototype versions were designed, implemented and evaluated in the company. Consequently, the users' reflections on the appropriateness of the design were concerned. These occurred through joint design workshops, researchers observing the work situations, making interviews, performing paper prototype evaluations and log-file analysis, and getting direct feedback to the developers. Different methods had dissimilar benefits: prototypes concretized the design ideas; workshops provided a method to establish a 'common language' and to commonly understand the work processes and general requirements; work situation observations revealed some unspoken issues of work;

interviews gave detailed requirements and design suggestions; paper prototypes validated UI designs; and log-file analysis of usage and direct feedback grounded the comments. During the development process, when the users began to see the benefits of the system for their work, they started to propose further improvements, which could make the system even better. In fact, their number increased with the quality (i.e. lesser errors) – just as Prinz et al. (1998) proposed. The value of the systems can be expressed in the following quote from one of the users of VIVA:

> "We could work with text chat, since it has been used and tested so much that we know it thoroughly, and know how to apply it to different situations. And with [tele]phone, which does the same thing [as text chat] but only quicker. Audio [in VIVA], however, is good if there are three participants. […]. But after all, I think the added-value with VIVA is the combination of different media." (MT, autumn 2001)

3 Case 2: Support for inter-organizational activities

Another project, TechMedia (2003), focuses on inter-organizational communication and information sharing. The project aims at developing ICT solutions to support networked business operations between a manufacturer and their customer on a factory floor level. In other words, the project tries to support cooperation between groups of experts in two organizations with different objectives, strategies, cultures, operations, practices, and technologies, among other things.

The manufacturer is the same as in the Case 1. However, now the activities are not as time critical as earlier, since the manufacturer is offering only support services to the customers. These include, for instance, maintenance and repairing services, analysis of problems or potential problems, and fine-tuning and modifications for improved performance of the machine they produce. These services are exploited in routine maintenance and minor problem-fixing operations as well as when planning larger maintenance maneuvers.

But this is still to come. Currently, the manufacturer has no means to monitor how their suggestions and services are considered. They can monitor a part of the customer's information systems, but not the whole, making it difficult for the manufacturer to understand the context and identify causal-relationships. These make it difficult to distinguish the benefits of the services so that they can be improved, and more importantly, so that they can be sold to others. On the other hand, the customer does not know how to make the most of the service – their processes and information systems do not meet this objective. They have done their industrial business without such services for dozens of years. Nevertheless, both the manufacturer and their customer agree that there is a desperate need for these services to keep them on the market. (c.f. Heikkilä et al. 2005).

In TechMedia project, the objective is to support and encourage knowledge management between the manufacturer's and customer's employees at the factory floor level. This is approached by implementing a shared workspace where the reports on monitoring and analysis are uploaded, their statuses are monitored, and related discussions held.

The development process was far from straightforward. At the beginning, none of the parties; manufacturer or their customer, had any concrete idea what the outcome would be. This was because no one knew the (business) process to be supported, the future users, the infrastructure where the system would operate, or what information was needed to be shared. Even the need for such a system was initially questioned by the customer. In other words, all traditional points of reference were missing. However, after organizing numerous interviews and workshops with different parties, it became possible to create a mutual understanding of the problems, challenges, and solutions of a new business model.

In contrary to the Case 1, here the user is less reflective but more participative. The appropriateness of the design solutions are still to be seen as the work is in progress, and for instance, the (business) process is not fixed yet. Still, because the users participated in the design and even acted as system designers, it is expected that the results are to be validated and approved, as Lynch and Gregor discussed (2004). It is expected that overall design is appropriate, although some details might get changed in the future.

4 Discussions

Here I have presented explorations from two different cases where the users were involved as 'reflectors' for the appropriateness of design decisions. Can a 'reflective' user be defined accordingly?

In the first case, the research was closer to systems development. This means that unknown variables are few. The users know what they want (on a large scale), they are known, there is a certain process to be supported no matter how vaguely it is defined, information flows are more or less defined, and underlying infrastructure is known. The second case is further away from development as the situation is more complex – basically nothing is known. There the user cannot evaluate the design precisely. So, for the users to be 'reflective', the development project must be closer to 'development' rather than 'research'. Users, conditions, processes and environment have to be simply known.

Both cases (and all the projects) followed action research approach where the researchers attempt to alter the object of the study. In Case 1, the researchers were more distant to the users while in Case 2 they cooperated very tightly with the users. To compare these modes of cooperation from the reflective user point of view, in Case 1 the users were equally active but more reflective while in Case 2 they were more participative. In other words, for users to be reflective, they cannot be too engaged with the developers. This minimizes their reciprocal

influence and provides a ground for objective evaluations of the appropriateness of the design decisions. If the users are 'too' engaged, their role will change to as participants influencing, and not validating, the design.

In Case 2, the number of unknown factors was extensive. Business process, the users, technological infrastructure and information-flows were all unknown and undefined (in fact some of them are still so after two-thirds of the project). This led to tight collaboration with potential users. However, regardless of this kind of intensive cooperation, one can question whether the users can really be relied on. Each user, being reflective or not, looks at the situation from his/her perspective, with his/her own experiences, knowledge, education, history, and tasks. This personal background is influenced by the organizational issues such as organization's objectives, strategies, cultures, operations, practices, and technologies, among others. Hence, if the user can provide comments about the appropriateness of the design, they are more likely to be biased especially in unfamiliar cases and situations. In Case 1, the users were able to do so as they were familiar with the objectives. In Case 2, this is less likely to happen as the number of variables is much greater.

5 References

de Michelis, G., E. Dubois, M. Jarke, F. Matthes, J. Mylopoulos, J. W. Schmidt, C. Woo & E. Yu (1998). 'A Three-faceted View of Information Systems.' *CACM* 41(12): 64-70.

Heikkilä, J., M. Heikkilä, J. Lehmonen & S. Pekkola (2005). 'Smart ICT support for business networks'. Vervest, van Heck, Preiss & Pau (eds) *Smart business networks*. Springer: 389-403.

Iivari, J., R. Hirscheim & H. K. Klein (2001). 'A Dynamic Framework for Classifying Information Systems Development Methodologies and Approaches.' *JMIS* 17(3): 179-218.

Lynch, T. & S. Gregor (2004). 'User Participation in Decision Support Systems Development: Influencing system outcomes.' *European Journal of Information Systems* 13: 286-301.

Lyytinen, K. (1986). *Information Systems Development as Social Action: Framework and Critica*l *Implications*. Jyväskylä, University of Jyväskylä. McConnell, S. (1996). *Rapid Development*, Microsoft Press. Nurminen, M. I. & U. Forsman (1994). 'Reversed Quality Life Cycle Model'. Bradley & Hendrick (eds.) *Human Factors in Organizational Design and Management IV*. Elsevier: 393-398.

Pekkola, S. (2003). *Multiple media in group work: emphasising individual users in distributed and real-time CSCW systems*. Jyväskylä Studies in Computing. Jyväskylä.

Pekkola, S., A. Rikalainen, T. Toivonen, S. Hujala, N. Kaarilahti, R. Lintinen & P. Pohjola (2003). 'MADE - A Groupware Application to Support Real-time Activities of Distributed and Co-operating Communities'. *Groupware: Design, Implementation, and Use -9th International Workshop (CRIWG 2003)*, Springer Lecture Notes in Computer Science 2806.

Prinz, W., G. Mark & U. Pankoke-Babatz (1998). 'Designing Groupware for Congruency in Use'. In: *CSCW'98*, ACM Press. Robinson, M. (1993). 'Design for unanticipated use ....". In: *ECSCW'93*, Kluwer Academic Publishers.

Discussion Report: Simplicity and Appropriation

Paul Dourish,
Thomas Herrmann,
Wendy A. Kellogg,
and Gabriele Kunau
UC Irvine, Ruhr-University of Bochum, IBM T.J. Watson Research Center, and Ruhr-University of Bochum
jpd@ics.uci.edu, thomas.herrmann@rub.de, wkellogg@us.ibm.com, Gabriele.Kunau@iaw.ruhr-uni-bochum.de

Abstract. Going beyond the traditional view of appropriation as the customization and tailorability of technical systems, we consider a view of appropriation as a set of emergent socially constructed meanings enacted among a community of users – that is, appropriation as a sociotechnical phenomenon (Dourish 2001a, 2001b). We lay out the rationale behind such an approach, the key role of simplicity in supporting collective appropriation, and consider how we might design for this view of appropriation. We argue that simplicity and a design stance of "less is more" are key elements in supporting appropriation.

1 Starting Points

Our discussion began with the realization that appropriation has several distinct meanings, all of which were in play at the workshop. Perhaps the canonical interpretation of appropriation is that of customization and tailoring by users. Yet we felt that other interpretations, such as unexpected use of technology,[1] or the

[1] That is, unexpected by system designers and developers.

socially constructed meanings around technology and its use that grow out of users' practices, were equally interesting and worthy of consideration.

We realized that there was a set of assumptions underlying our conversation. First, we believe that appropriation is "always already social;" this is obvious in the case of appropriation as socially constructed meaning, but even individual acts of appropriation are embedded in a social context, if only by virtue of the fact that an individual who, say, customizes their software can tell others who may then similarly customize their software. Second, we assume that users know what they are doing, what they are trying to accomplish – in fact, they know more than designers do about their contexts of technology use. Third, we assert that users' understanding of a technical system and its capabilities is the basis for any creative practice utilizing it; this is the key contribution of simplicity. Fourth, appropriation pragmatically is a collective activity, which raises the question of the appropriate unit of analysis for research. From this perspective, analysis must include both the technical system and the practices of a community of users embedded in a social context. This emphasizes the central role of communication channels among users, and reminds us that technology itself can function as a communication channel, either by supporting communication directly, or indirectly by making users and their use of the technology visible to others and thus a source of social dynamics (e.g., imitation, peer pressure, etc.).

2 Simplicity and Appropriation

Given this view of appropriation as enacted by a community of users, what is meant by simplicity? Simplicity on this view is a relational property that emerges from the interplay of a technology with users' intentions and the social structures in which use is embedded. Although a simpler technology may be easier for users to understand, understanding in and of itself is only prerequisite to appropriation. The community of users must also have a means of collectively instantiating and evaluating adaptations of a technology. Such means may include a variety of meta-capabilities, for example, the ability to communicate about the technology itself, the ability to see the actions of others, the ability to understand how others will see one's own actions, etc. This also suggests that concepts such as self-description (Kunau et al., 2005), self-regulation (Kellogg & Erickson, 2005), and self-reference are key for understanding how to facilitate appropriation.

The consideration of how a community could possibly appropriate technology leads to the notion of a continuum of appropriation. In its simplest form, appropriation may simply evolve over the course of use, without explicit management on the part of the community. At the other end of the spectrum, appropriation may occur through a deliberate effort by the community to reflect on how a technology can or should be used to best achieve a variety of collective intentions and to learn from experience (e.g., by establishing norms or locally-

adapted use practices). Of course, there is also room for a variety of appropriation mechanisms between these extremes.

Considering how use might spread through a community of users, we also realized that appropriation is not necessarily always "good" or appropriate. The practice of responding to warnings promulgated through email is encouraged when unsuspecting users forward such emails to their trusted correspondents; when these are in reality virally-spreading phishing schemes, as they often are, this is not good. Advice to set your file sharing permissions a certain way to facilitate music sharing that in fact exposes personal information on one's hard disk is not a good adaptation to make. Thus, the easier it gets for practices and adaptations to spread, the more critical become issues such as bounding and controlling evolving use, or having a means of establishing trustworthy role models and leadership to advise and protect users.

3 Designing for Appropriation

We next turned our attention to the issue of designing for appropriation; what would it mean to do so on this view? This is an open area of research, but we articulated four areas where design might be expected to impact appropriation: first and foremost, enabling users to see the consequences of their own and others' actions. Second, progressive disclosure of function may help, which again speaks to simplicity. Third, as discussed previously communication channels are critical. Fourth, deixis (literally, the ability to point to a part of the technology in use) and reference are necessary.

There are also design consequences for viewing appropriation as a collective rather than an individual phenomenon. There is a difference between a collective practice – for example the kinds of norms established by Babble users (see Kellogg & Erickson, 2005) – and the case of many individuals who may "do" the same thing or "have" the same customization. The latter is not a collective appropriation, but at the extreme a kind of "convergent evolution" (many individuals expressing the same adaptation in response to similar 'evolutionary pressures,' but independently of each other). A primary design goal, then, is to discover and then support the social processes that can enable and shape collective action. The emphasis shifts from customization to negotiation; the notion of simplicity shifts from making easier the user's choice among a vast array of customization options to making it easier for a community of users to propose, try out, and reflect on various ways of using a technology.

4 Research Issues

We can summarize the broad research issue at stake here as "how can we do more with less?" Rather than focusing on an expanding set of "cool" customization features, we ask how little can we get away with? How can we reduce the complexity of the technology, get it out of the way, while increasing and enhancing the ways in which individual users can profit from each other's experience, or that collectives of users can negotiate ever more optimal and suitable adaptations?

Finally, there is still much to sort out in the relationship between social meaning and individual action and in how appropriation can be managed. Issues here range from support for leadership and role models among community members, to how to enhance users' ability to self-describe and self-regulate, to basic issues of how to support the emergence of norms in online environments.

5 References

Dourish, P. (2001a). *Where the action is*. Cambridge, MA: MIT Press.

Dourish, P. (2001b). Seeking a foundation for context-aware computing. *Human Computer Interaction*, **16(2-4)**, pp. 229-241.

Kellogg, W.A. and Erickson, T. (2005). Supporting appropriation work with social translucence, collective sensemaking, and social scaffolding. Workshop on Supporting Appropriation Work: Approaches for the 'Reflective' User. European Computer-Supported Cooperative Work, September 18-22, 2005, Paris, France.

Kunau, G., Herrmann, T., and Loser, K-U (2005). Socio-technical self-descriptions as a means for appropriation. Workshop on Supporting Appropriation Work: Approaches for the 'Reflective' User. European Computer-Supported Cooperative Work, September 18-22, 2005, Paris, France.

Biographies

Ellen Balka is Professor with the School of Communication, Simon Fraser University Burnaby, British Columbia.

Arthur B. Baskin works as a researcher with Intelligence IT, Indianapolis, Indiana (USA).

Marek Bell is currently studying towards a Ph.D. at the Department of Computing Science of the University of Glasgow. His research interests lie in models of user-interaction and awareness and his supervisors are Dr. Matthew Chalmers and Phil Gray. He belongs to the Equator IRC which is funded by the EPSRC.

Alan Borning is a professor with the Department of Computer Science & Engineering at the University of Washington, and also adjunct professor with the Information School. His current research interests are in human-computer interaction and the interactions between human values and computer technology, particularly as applied to simulations of urban development.

Elizabeth Brownholtz works with the IBM Watson Research Center. Her Research Interests are: Synchronous collaboration, Attention management, Software patterns and Collaborative Development Environments. Since November 2003, Beth has been working on the Activity Explorer (AE) project. AE enables users to combine ad hoc, real-time communications with the rich collaboration features of shared workspaces. It helps people share their work at the document/object level and combine these shared documents/objects into hierarchical structured collections, or activity threads that capture complete tasks and activities.

Diego Calzà works with the department of sociology and social research, University of Trento, Italy.

Matthew Chalmers is Reader in Computer Science at the University of Glasgow. His current research aims to take account of social and perceptual issues both in the design of computer systems, in visualisation, recommender systems and ubiquitous computing, and in the theory of computer science, relating contemporary semiology/philosophy to computational representation. In practice that generally means tracking and logging systems that show a lot of information, and then feeding that logged data back to people and into adaptive system infrastructure.

Vincenzo D'Andrea works as an Associate Professor with the University of Trento. A short Description of his work is only available in Italian:
Professore Associato della Facoltà di Sociologia dell'Università degli Studi di Trento. Membro del Dipartimento di Informatica e Telecomunicazioni. Collaboratore del Laboratorio di Ingegneria Informatica.

Janet Davis works as a doctoral candidate with the Department of Computer Science and Engineering at the University of Washington. Janet's focus is on design to enhance civic engagement and support the legitimacy of complex systems such as UrbanSim in democratic decision making. Other interests include the study of value issues (e.g., privacy) throughout the undergraduate computer science curriculum.

Paul Dourish is an Associate Professor of Informatics and Computer Science at UC Irvine, and Associate Director for Research of the Irvine Division of the California Institute for Telecommunications and Information Technology (Cal-(IT)2). His principal research interests are in Ubiquitous Computing, Computer-Supported Cooperative Work, Human Computer Interaction, and Social Studies of Science and Technology.

Thomas Erickson is a Research Staff Member at the IBM T.J. Watson Research Center in New York where he works on designing systems that support network mediated group interaction. His research involves exploring the design and use of social proxies, minimalist graphical visualizations of people and their activities in online environments. Originally trained in cognitive psychology, Tom gradually morphed into an interaction designer and researcher via stints in a small startup (5 years) and Apple's Advanced Technology Group (9 years); he has been at IBM since 1997. His approach to systems design is shaped by work in sociology, rhetoric, architecture and urban design. He has contributed to the design of many products, and authored about 50 publications on topics ranging from personal electronic notebooks and information retrieval systems to pattern languages and virtual community.

Werner Geyer is Research Staff Member, IBM T.J. Watson Research Center, Cambridge, USA. His main interests are Computer-Supported Cooperative Work, Human Computer Interaction, Group Communications, Groupware, Collaboration Architectures, Ubiquitous Computing, Personal Information Management, Team Productivity, Electronic Meeting Support, Networked Multimedia, Communication Protocols, Distance Education.

Malcolm Hall is currently pursuing a PhD in the area of adaptive mobile recommendation systems at the University of Glasgow.

Thomas Herrmann is professor with the Ruhr-University Bochum, Institute of Applied Workscience, Department for Information and Technology Management. His main research-interests are evaluation and design of socio-technical systems in accordance with human needs and organizational structures.

Gianni Jacucci works as a Professor with the University of Trento, Department of Computer and Management Sciences (D.I.S.A.), Laboratory of Information and Communication Technologies.

M. Cameron Jones works with the University of Illinois as a Graduate Research Assistant.

Wendy A. Kellogg is Manager of Social Computing at IBM's T. J. Watson Research Center. Her current work involves designing and studying systems for supporting computer-mediated communication (CMC) in groups and organizations. Wendy's past work in human-computer interaction (HCI) has included theory, evaluation methods, design, and development. She holds a Ph.D. in Cognitive Psychology from the University of Oregon. She has authored and edited papers in the fields of Cognitive Psychology, HCI, and CSCW. She served as Technical Papers Co-Chair for CHI 2005, as Technical Program Co-Chair for ACM's DIS 2000 ("Designing Interactive Systems") conference, and as General Co-Chair for ACM's CSCW 2000 ("Computer-Supported Cooperative Work") and ACM's CHI 1994 ("Human Factors in Computing Systems") conferences. Wendy is a past member of the National Academies of Science Computer Science and Telecommunications Board and was elected ACM Fellow in 2002.

Gabriele Kunau works as a researcher with the Ruhr-Universität Bochum, Institute of Applied Workscience. Her research focus lies in the integrated support of software development and organizational change.

Riad Lemhachheche works with the Oregon State University. His interests are Information Architecture, specifically Telecommunication Services, Computer Supported Collaborative Work and Electronic Privacy. A second point of interest is Mobile Networking, specifically Ubiquitous Computing, Context Awareness, Interaction Design and Human - Computer Interaction.

Kai-Uwe Loser is data protection officer at the Ruhr-University Bochum.

David W. McDonald works as a researcher with the Information School University of Washington. His research interests are computer-supported cooperative work (CSCW), human-computer interaction (HCI), system design, software architecture, software engineering, ethnographic study, and the social analysis of technology.

David R. Millen is a group manager in the Collaborative User Experience group at IBM T J Watson Research in Cambridge, MA. His group studies the social and technological implications of on-line communities and large-scale collaboration. Through field study and prototype applications, his group explores how to create and support distributed teams and on-line communities. Many of these applications include interactive visualizations to enhance the user experience and help make large amounts of information accessible to the community. Prior to joining IBM, David worked at AT&T Labs, where he explored how new technologies changed employee work activities, organizational roles, and patterns of communication. His research keywords are: Online communities, Computer-mediated communication, Computer-Supported Cooperative Work (CSCW).

Suzanne O. Minassian. As the Domain Engineer for IBM Workplace for Business Strategy Execution (IWBSE), Suzanne works to define the overall product function and development strategy. She also collaborates with customers to evaluate how IWBSE should be designed for their business needs from a functional and interface perspective. Having come from IBM Research, she is particularly interested in collaboration technologies, personal and business productivity tools, and planning, conducting, and performing user studies and research. Suzanne graduated from Bentley College with an MBA, concentrating on Human Factors in Information Design.

Michael J. Muller is Research Scientist/Design Researcher at IBM Watson Research Center. His Research Interests are: Participatory design, Communities of practice, Methods for analysis, design, and assessment, Ethnocritical analyses of the roles of people in human-computer interaction, Human-computer interaction work as cultural and linguistic translation.

Samuli Pekkola PhD, is currently Professor and Vice head of the department in the Department of Computer Science and Information Systems at the University of Jyväskylä, Finland. His research evolves around the support for office and group work through different devices and around participatory information systems development methods.

Volkmar Pipek studied Computer Science and Economics at the University of Kaiserslautern, focussing on Database Systems and Artificial Intelligence. His research interest into interdisciplinary, more application-oriented computer science leads him to the Research Group on HCI and CSCW (ProSEC) at the Institute for Computer Science III at the University of Bonn. He worked from July 1997 to December 1998 in the project POLITeam on "awareness" issues and organizational aspects of introduction and maintenance of groupware applications. 1999 he worked in several smaller projects on Knowledge Management and Distance Learning. From April 2000 to March 2003 he was coordinating the project OlViO, a project on the use of IT in Organisational Learning. Currently he is a guest researcher at the Laboratory of HCI and Group Technology at the University of Oulu, Finland. He belongs to the board of trustees of the International Institute for Socio-Informatics (IISI).
http://members.iisi.de/pipek/

Markus Rohde, Dipl.-Psych., studied psychology and sociology at the University of Bonn and is one of the founders of the International Institute for Socio-Informatics (IISI). At the time being he is working as project manager for IISI and as research associate at the Institute for Information Systems at the University of Siegen. Moreover he is editor of the political science journal "Forschungsjournal Neue Soziale Bewegungen" (New Social Movements). Since 1991 his research focuses on usability engineering of network systems, on virtual organizations, and on "organization and technology development". Since 1994 he is working as a consultant for medium-sized enterprises and for nonprofit-organizations. From 1997 until 2001 he worked as CEO of AGENDA CONSULT GmbH. In 2004 he was research associate for Fraunhofer Institute for Applied Information Technology (Fhg-FIT) in Sankt Augustin. His main research interests are human-computer interaction, computer supported cooperative work (CSCW), expertise management and blended learning, virtual organizations, non-governmental organizations and (new) social movements.
http://members.iisi.de/rohde/

Till Schümmer works as a researcher with the FernUniversität Hagen. His research interest are CSCW, CSCL, distributed systems, software engineering and agile methods, design patterns and cooperative games.

Bettina Törpel is an Assistant Prof. with Technical University of Denmark in Lyngby, Copenhagen.

Michael B. Twidale works with the Graduate School of Library and Information Science, University of Illinois His activities are Computers: specifically computer-supported cooperative work, online information systems, digital libraries, user interface design, museum informatics, designing interfaces to support informal workplace incremental learning of computer systems.

Ina Wagner is Professor for Multidisciplinary Systems Design and Computer-Supported Co-operative Work (CSCW) and Head of the Institute for Technology Assessment and Design. She has edited and written numerous books and authored over 100 papers on a variety of technology-related issues, amongst them a feminist perspective in science and technology, ethical and political issues in systems design, computer-support of hospital work and of architectural design and planning, CSCW and networking. One of her main research interests is creative design work.

Eric Wilcox is a designer researcher working in the Collaborative User Experience Group at IBM. He has led innovative research and forged relationships between research and product divisions through his use of design. Eric has lectured and taught classes in time based media, and currently chairs the Boston AIGA Experience Design Community.