Seven Challenges for Developers of Mobile Groupware

Jörg Roth

University of Hagen
Department for Computer Science
58084 Hagen, Germany
Joerg.Roth@Fernuni-hagen.de

1 Mobile Groupware

Synchronous groupware applications play a major role for, e.g. shared document editing, co-operative software development or shared workspaces. Groupware applications allow geographically distributed teams to collaborate without significant time delays. Several groupware platforms have been developed to simplify the development of synchronous groupware applications and relieve the developer from standard problems such as communication, synchronisation, coordination or concurrency control.

Currently, there exists a growing market for mobile devices such as PDAs, mobile phones, and electronic pens. Upcoming communication technologies (e.g. wireless local or personal area networks, mobile telephone networks) promise new services for mobile communication. Extending stationary groupware concepts and platforms to mobile environments would offer great potentials. However, too straightforward approaches, e.g. simply running existing groupware platforms on mobile devices, fail due to the different nature of mobile devices and networks.

In the year 2000, we started to investigate, how mobile users and developers of mobile groupware applications could be supported with the help of groupware platforms. In order to examine the specific problems in more detail, we created two platforms: *QuickStep* was especially designed to support groups of handheld users, the second platform *Pocket DreamTeam* was an extension of our stationary groupware framework *DreamTeam*. For both platforms, we had the following design goals:

- We want to support synchronous collaboration. Due to some device and network characteristics, we relaxed the notion of *strict* synchronous collaboration and introduced the term *relaxed synchronous collaboration* [6] for group members who collaborate synchronously, but may be disconnected from the network for short periods.
- We want to support developers of new mobile groupware applications with the help of a development and a runtime environment. The development environment should offer communication and data abstractions as well as an application programming interface (API) for group specific services and widgets. The runtime system should perform standard tasks such as controlling communication links, distribute shared data, manage session and user profiles, perform group rendezvous etc.
- As a technical platform for mobile end-user devices, we decided to use handhelds as shown in fig. 1. We in particular decided not to use notebooks. Due to their size, weight and battery life, mobile working capabilities with notebooks are limited. We especially want to investigate problems related to small mobile devices and want to transfer our solutions to even smaller devices such as electronic pens or mobile phones later.

Fig. 1: Handheld device running a mobile groupware application

Our development and testing system consists of

- handheld devices based on PalmOS equipped with wireless LAN (IEEE 802.11b) adapters,
- a number of stationary workstations (Windows PCs, Solaris workstations) and
- a wireless LAN infrastructure connected with the campus Internet.

Although our testing environment primarily based on wireless LAN, we strictly paid attention to be independent as possible from the network. In principle, our platforms could run on other wireless networks such as IrDA (Infrared), Bluetooth or GSM. Since not all networks support the Internet Protocol (IP) sufficiently, we isolated network related functions in a component we call *Network Kernel Framework* (*NKF*). NKF can roughly be compared to a high-level network driver and offers a uniform interface to the platforms.

In the following, we briefly describe the mobile groupware platforms QuickStep and Pocket DreamTeam before we generalise our experiences.

1.1 Case Study I: QuickStep

The QuickStep platform [3, 6] supports developers of mobility aware collaborative handheld applications. They can use communication, collaboration and dialog primitives provided by the platform and can concentrate on application-specific details. A developer can integrate predefined awareness widgets into an application with a few lines of code. We can summarise the QuickStep approach as follows:

- QuickStep supports applications with well-structured, record oriented data, as being used by built-in software for handheld devices (e.g. for to-do lists, memos, telephone lists). QuickStep was explicitly not designed for supporting multimedia data, graphical oriented applications or continuous data streams.
- QuickStep mainly supports (relaxed) synchronous collaboration. In addition, asynchronous collaboration is possible.
- QuickStep provides awareness widgets for collaboration and context awareness.
- QuickStep comes along with a generic server application, which supports arbitrary client applications without modifying or re-configuring the server.
- Data distribution is based on the database abstraction integrated into most handheld operating systems. The consistency strategy relies on a strong relation between data rows and participating users. To reduce network traffic and to perform as many computations as possible on a server, we developed a combined mirroring and caching mechanism that we designed according to the synchronisation pattern [4].
- Users can form collaborative groups without central administration, e.g. all users inside a meeting room can collaborate spontaneously. As untrusted users can participate, we have to address security problems. QuickStep contains several mechanisms to ensure privacy of individual data. Data rows are, e.g., anonymised before they are transferred across the network.

a) Joerg's handheld b) Stephan's handheld

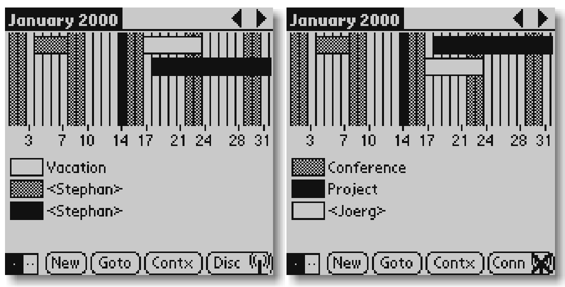


Fig. 2. QuickStep sample application

Fig 2. shows a QuickStep sample application, a collaborative meeting planner. Two or more users can exchange dates and plan further meetings. Each entry, indicated by a bar, is distributed in real-time. The lower half of the window is the legend for the upper half. Foreign entries are anonymised automatically, i.e. the entries are labelled by the user name rather than by the private label (e.g. "Vacation"). Other sample applications are, e.g., a business card collector and a brainstorming tool.

1.2 Case Study II: Pocket DreamTeam

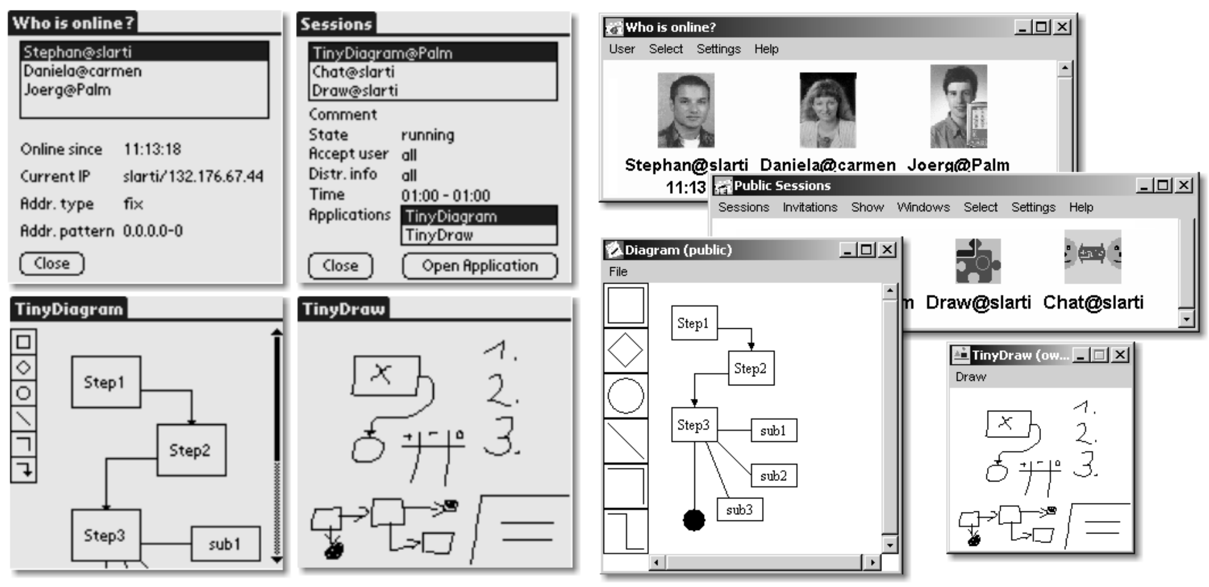
Pocket DreamTeam is the mobile version of our groupware platform DreamTeam [1, 2]. DreamTeam has a fully decentralized architecture without the need for a central server. Shared data are distributed among the group members using an automatic replication mechanism. DreamTeam offers a number of coordination services. It has been successfully used for practical software courses and diploma theses at the University of Hagen. There exists a huge variety of about 20 DreamTeam applications such as distributed sketch, diagram and text editors, a collaborative slide presentation program, a brainstorming tool and a group web browser. After we finished the implementation of the desktop variant of DreamTeam, we planned to develop a handheld version Pocket DreamTeam, which should meet the following requirements:

- Pocket DreamTeam and the desktop version of DreamTeam should run inside the same network. It should be possible to form arbitrary sessions of handheld and desktop users, with e.g., only handheld users, only desktop users or a mixture.
- The original desktop variant of DreamTeam should still run *without any changes*.
- Since handheld devices differ fundamentally from desktop computers, it is not reasonable to follow the desktop usage paradigms based on, e.g., overlapping windows with graphics. We accepted to re-implement some parts of DreamTeam for handheld devices. However, we wanted to keep the amount of new developments as small as possible.

Although DreamTeam has a decentralized architecture, it contains a semi-central coordination mechanism: a session chair creates a session profile for a planned session and sends invitations to possible members. On one hand, this mechanism limits spontaneous collaboration, on the other hand it is much more easy to obtain privacy since the chair has control over the participation process.

Adapting the full replication mechanism to weakly connected devices was a real challenge. The concurrency control of the original DreamTeam uses pessimistic locks, which are not appropriate for mobile users: disconnected users, which hold a lock, can disable an entire session. As a solution, Pocket DreamTeam uses a combined optimistic/pessimistic concurrency control strategy which takes the advantages of both approaches: on the stationary network segment with high reliability and low latency, pessimistic concurrency control is appropriate. On the mobile network segment, optimistic concurrency control leads to much better response times but requires more complex algorithms.

a) Mobile Applications b) Stationary Applications


Who is online?
Stephan@slarti
Daniela@carmen
Joerg@Palm
Online since 11:13:18
Current IP slarti/132.176.67.44
Addr. type fix
Addr. pattern 0.0.0.0-0

Sessions
TinyDiagram@Palm
Chat@slarti
Draw@slarti
Comment
State running
Accept user all
Distr. info all
Time 01:00 - 01:00
Applications TinyDiagram
TinyDraw

Fig 3. DreamTeam applications on mobile device and stationary desktop

Fig. 3 shows Pocket DreamTeam windows (left) and the corresponding desktop DreamTeam windows (right). Corresponding screens on different platforms may be completely different. Overlapping windows, context menus and icons are not useful on small screens. E.g., we replaced icon-based dialogs by simple textual lists.

The users use the upper frames to view contextual information and participate sessions. The *On-line list* shows all users, which are currently on-line, i.e. can participate in a collaborative session. The *Sessions* frame shows all running and planned session. A user can select a running session from a list and join it. From the 20 DreamTeam applications, we selected two applications for mobile extension:

- The *Diagram* tool allows a team to collaboratively create flow charts, entity relation ship or class diagrams.
- With the *Draw* tool, a group can draw and share simple free-hand sketches.

To reduce development costs, a developer can re-use the functional core of the desktop variant in the handheld version. A concept based on *resources* (high-level components, which model the functional core) ensures interoperability across device borders. Device-specific parts however, e.g. dialog frames, have to be re-implemented.

2 Seven Challenges

After we finished the QuickStep and Pocket DreamTeam projects, we tried to abstract from platform details and find a list of general *challenges* for a developer of mobile groupware. We identified the following:

Challenge 1: Communication: Currently the world of mobile communication technologies and protocols is rapidly growing and far from being mastered by developers and end-users. Upcoming technologies like UMTS or Bluetooth promise new mobile communication potentials, but often do not cooperate with existing networks. Mobile and ad-hoc communication leads to issues that affect all communication layers. We have to address issues such as ad-hoc routing, mobility management, service discovery or connecting devices across network borders. Possible solutions (e.g., Mobile IP, I-TCP, SDP, DSDV etc.) are currently not widely integrated into existing networks.

Challenge 2: Architecture: Distributing software and hardware components among the involved sites significantly influences the overall system behaviour. Finding an appropriate architecture for mobile groupware is a highly creative process. Although there exist semi-formal methods to lead system architects through the decision-making process (e.g. [4, 5]), it is difficult to detect all pros and cons of a specific architecture before realisation is finished.

Challenge 3: Coordination: Coordination functions are essential to groupware systems, e.g. to form collaborative groups, publish announcements, coordinate participation to sessions or perform group rendezvous. In contrast to stationary users, mobile users are only loosely connected to the groupware infrastructure, thus require more complex coordination mechanisms. Closely related to coordination is the topic *security* (see challenge 6).

Challenge 4: Data distribution and consistency: Distributing shared data among a group and maintaining consistency is one of the main functions of a groupware system. If mobile users are involved, we have to deal with weakly connected devices and devices which are often turned off, thus not accessible for others. This dramatically increases the problem of data distribution

Table 1. Comparison of QuickStep and Pocket DreamTeam

Challenge	QuickStep	Pocket DreamTeam
Communication	Network Kernel Framework (NKF)	
Architecture	client/server, hierarchical server network	remote proxy, fully decentralized
Coordination	spontaneous groups and sessions, no session chair required	session chair controls participation, limited support for unplanned sessions
Data distribution/ consistency	synchronisation, mirroring/caching, ownership of data	full replication, combined pessimistic/optimistic concurrency control
User interfaces	not primarily addressed, relies on interface creation tools of the target platform, predefined group & mobility widgets exist	
Security/Privacy	anonymising, life-time supervision	relies on security provided on communication level (NKF)
Realisation issues	database abstraction, generic server application	resource abstraction, multi-platform development required

and consistency especially in synchronous groupware systems.

Challenge 5: User interfaces: Mobile devices fundamentally differ from stationary computers [3]. We have to consider small screens, new input devices and different usage paradigms. Applications based on user interface plasticity could save development costs, but are still subject of research. In addition to the actual application interface, mobile groupware has to offer awareness widgets (e.g. to support collaboration or context awareness).

Challenge 6: Security/Privacy: Wireless communication usually based on physical broadcast. In principle, other users can read ongoing transmissions or modify transferred data. Data stored in some mobile end-user devices (e.g. PDAs) have private characteristics. If a team exchanges confidential data, a groupware system has to provide security mechanisms. Unfortunately, increasing security reduces the potentials of spontaneous communication, since we set up a cryptographic framework with, e.g., keys or certificates.

Challenge 7: Realisation issues: These are often neglected. Having mobile devices, we have special development environments, operating systems and programming paradigms. We have to consider hard device limitations, e.g., small memories, slow processors and limited parallel execution capabilities. Often, high-level software engineering approaches as used for desktop environments are not effective for small devices. Multi-platform development environments such as Java ME do not offer a sufficient degree of platform independence and stability, thus development costs are still high.

Table 1 summarises, how our platforms address these challenges. Some problems are still open and have to be addressed in the future. Especially development support for collaborative user interfaces is still rudimentary. Both platforms contain complex mechanisms for data distribution and consistency. Especially the full replication mechanism of Pocket DreamTeam with a combination of pessimistic and optimistic concurrency control is unique. The strength of QuickStep is the mechanisms to simplify spontaneous collaboration and to obtain privacy.

References

1. Lukosch S., Roth J.: Reusing Single-user Applications to Create Multi-user Internet Applications, Innovative Internet Computing Systems (I2CS), Ilmenau, June 21-22, 2001, LNCS 2060, Springer, 79-90
2. Roth J.: DreamTeam - A Platform for Synchronous Collaborative Applications, AI & Society (2000), Vol. 14, No. 1, Special Issue on Computer-Supported Cooperative Work, Springer London, March 2000, 98-119
3. Roth J.: Information sharing with handheld appliances, 8th IFIP Working Conference on Engineering for Human-Computer Interaction (EHCI'01), Toronto, Canada, May 11-13, 2001, LNCS 2254, Springer, 263-279
4. Roth J.: Patterns of mobile interaction, Proceedings of Mobile HCI 2001: Third International Workshop on Human Computer Interaction with Mobile Devices, M. D. Dunlop and S. A. Brewster (eds), IHM-HCI 2001 Lille, France, Sept. 10, 2001, 53-58
5. Roth J., Unger, C.: An extensible classification model for distribution architectures of synchronous groupware, in Dieng R. et al. (eds): Fourth International Conference on the Design of Cooperative Systems (COOP2000), Sophia Antipolis (France), IOS Press, May 23-26, 2000, 113-127
6. Roth J., Unger, C.: Using handheld devices in synchronous collaborative scenarios, Personal and Ubiquitous Computing, Vol. 5, Issue 4, Springer London, Dec. 2001, 243-252