Tools for Students Doing Mobile Fieldwork

Mattias Rost
Dept. of Applied IT, Göteborg University
Forskningsgången 6
SE 417 56 Göteborg, Sweden
Tel: +46 704 14 56 64
rost@viktoria.se

Lars Erik Holmquist
Swedish Institute of Computer Science
Box 1263
SE 164 29 Kista, Sweden
Tel: +46 703 55 85 00
leh@sics.se

Abstract

Students are not always sitting at their desk but are also out in the world. In a university course teaching ethnography and design students were out in the field making observations and collecting data. We gave them access to a wiki, which they used to upload field notes and material as a support for collaboration. In this paper we present three tools we built and deployed to aid the students when in field and when collaborating. The first is a mobile tool used to gather data: a program running on the students' mobile phones let them take photos, record video and audio, and write simple text notes, which are automatically uploaded to the wiki. The second is an awareness tool that enables the students to quickly see what the others have done in the wiki. The third is a novel browser for the uploaded data, which relates objects by both time and location. We also talk about the experience from having students using the tools live during the course.

1. Introduction

Students often work at a desk, either reading a book or listening to a lecture. However there are activities where students are actually out in the real world. When being mobile, it is not always very suitable to bring a laptop computer even if they need the capabilities that these devices offer. Instead they inhibit their freedom of movement, and can also serve as an obstacle when interacting with other people at the same time. However, it might be that students are actually out gathering observations and experience about a phenomenon or practice, and therefore need to take notes or capture data which they have to bring back to their desktop for reflection and discussion. This poses various problems.

We report from a course teaching ethnography and design at the IT University of Göteborg, where students work in groups studying a workplace of their choice. They start by getting access to the workplace, and then spend two weeks out in the field. During this time they have to take notes and collect data - taking photos, recording videos or audio. The students themselves have reported that just deciding what kind of notebooks to bring into the field is hard (as it may affect how they are treated by the people they observe) [2], suggesting that using a laptop computer is out of the question. At the end of the day however, they need to get this into their computers to be able to share with their friends in the group to analyze the data. We were therefore interested in building tools to support the students in this endeavor.

In previous years we have experimented with having a *wiki* – easily editable web pages - to support the students. They used the wiki to type in their field notes, put up their work plans, and upload other material gathered in the field, such as photos and drawings. The wiki thus served as a group repository, allowing the individuals to collect their own material as well as get access to their group's material (for more details, see [10]). Having your material in one place was highly beneficial compared to having it spread out on the group members' personal computers. Users could access the material from anywhere as long as they had access to a web browser, and they could link the material directly to individual wiki pages and discuss it. Even if the wiki supported the collaborative aspect of their work it did not support them when they were actually mobile.

They still had to type in their notes when they got home, and upload any photos or videos after getting that data of their cameras. We therefore decided to build a mobile tool to easily take photos, record video and audio, and write short notes, and automatically get these into the wiki.

When we studied the usage of the wiki, it became apparent that the students found it very beneficial to look at each other's texts, and that they would benefit from an increased knowledge about the others' work - what in the field of computer-supported cooperative work is known as *awareness* [5]. We therefore decided to provide an extension to the wiki to provide this awareness, that would tell students at a glance what others had been doing, without forcing them to install any special new software.

In this paper we present three addons for wikis; an awareness extension, a mobile application for capturing data in the field (photo, video, audio, text) and uploading to the wiki, and an extension to the wiki which let you browse through captured material on the wiki.

2. Systems

To setup our wiki, we used the popular wiki engine *TikiWiki* (tikiwiki.org). TikiWiki supports numerous features in addition to the basic wiki functionality, including file and image galleries, blogs, discussion boards, etc. Our configuration had the wiki and the galleries enabled, and allowed comments on wiki pages. Access was restricted so that users had to log in with a username and password in order to read any text, and the only ones given access were the students and the teachers of the course.

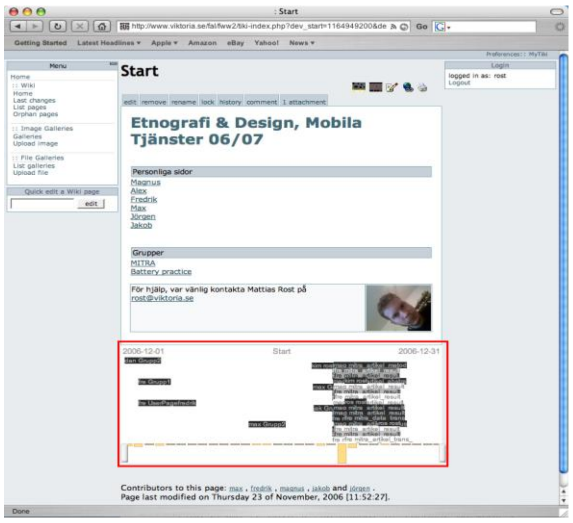
One of the strengths with a wiki is the accessibility of it. As long as you have a web browser you can access whatever is in it from anywhere. We wanted to incorporate this strength as far as we could, and so we wanted to implement the wiki-extensions as simple Rich Internet Applications, running inside the web page using standard APIs (with Ajax techniques [7]).

We will now talk about the tools we built.

2.1 Awareness Tool

In order to support awareness for groups working in our wiki, we wanted to design a visual representation of the activity, which clearly showed what had been added or changed, and by whom at what time. Whenever a person creates or edits a page, or uploads an image or a file, it should be visible to anyone else without being intrusive, when they visit the wiki. In this way users would be able to keep track of each other's work, and follow the progress of the wiki content.

2.1.1 Design. The result is an interactive zooming graphical timeline at the bottom of each wiki page, as shown in Figure 1. The timeline is split horizontally in two parts, providing both overview and detail. The bottom part shows the total number of *events* each day for the last thirty days represented as a histogram. An event here is an action within the wiki, such as creating or editing a page, or uploading an image or a file. The user can zoom in on a time interval by dragging two sliders to choose specific dates. When dragging the sliders to zoom in or out, the visualization is animated in real-time, creating a smooth animation as objects in the upper view gets more spread out or more compact, much like other zoomable interfaces, e.g those created with the zoomable UI toolkit Jazz [3] (not to be confused with the software development system of the same name [9]).



Etnografi & Design, Mobila Tjänster 06/07

Personliga sidor

Magnus
Alex
Fredrik
Max
Jörgen
Jakob

Grupper

MITRA
Battery practice

För hjälp, var vänlig kontakta Mattias Rost på rost@viktoria.se

Contributors to this page: max, fredrik, magnus, jakob and jörgen.
Page last modified on Thursday 23 of November, 2006 [11:52:27].

Figure 1 A collaborative wiki page with our awareness extension visible at the bottom.

The upper part of the timeline shows the detail view, i.e. the events in the chosen time interval. The events are represented by short text strings stating which object (wiki page, image, etc.) is concerned and who caused it. If the same user causes many events on the same object in a short amount of time, they are grouped together. The events are spread out vertically to give more space for events occurring close in time. If the user lets the mouse pointer hover above the text string, a box (similar to a tool-tip) will appear to describe the event in more detail, giving information about exact date, what type of object it is, etc. To go to the corresponding page in the wiki

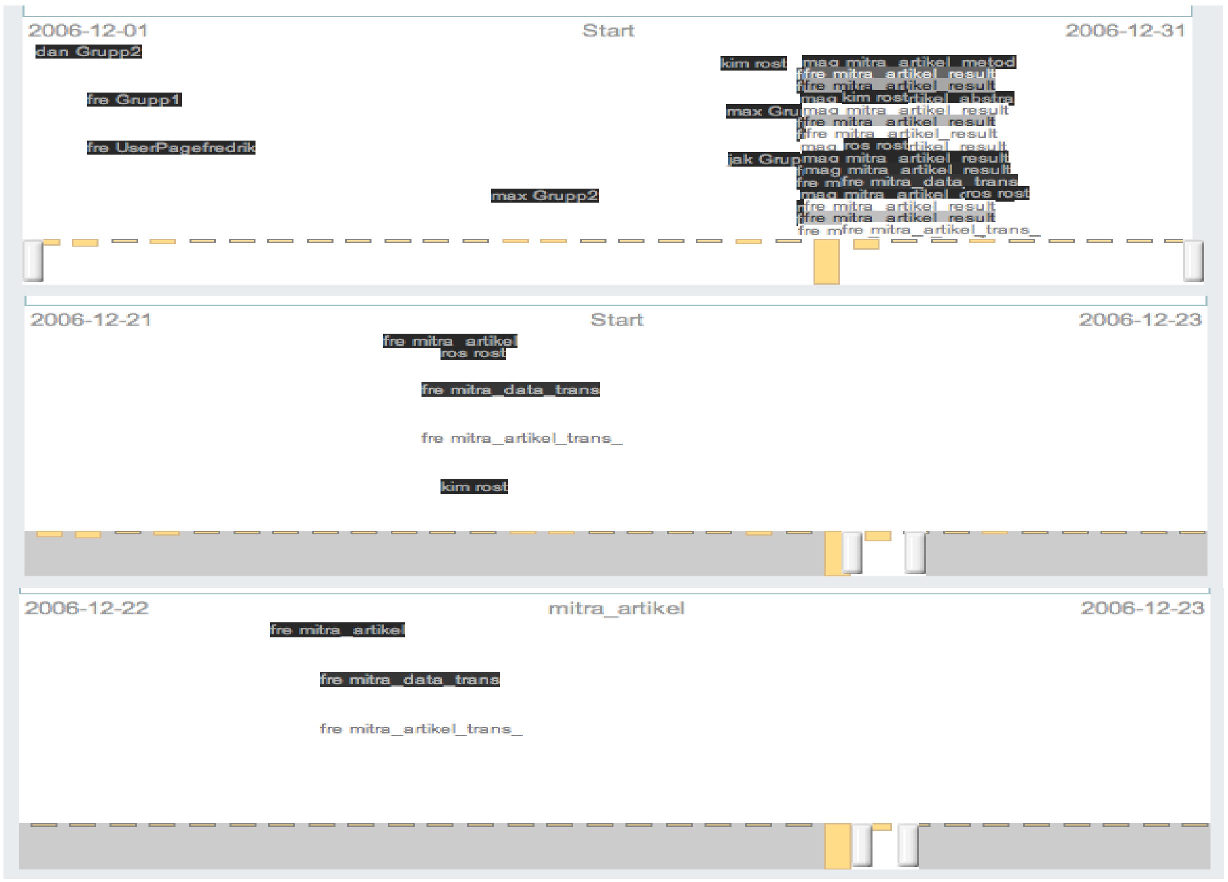


Figure 2 Example views of the awareness extension. (top) View from the start page. A lot of edits have been made around the 21st of December. (middle) Zoomed in view around the dates shown at the top, and in the timeline. (bottom) The view from the 'mitra_artikel' page, showing only those event that belong in that hierarchy.

for the object, the user simply clicks on the text string.

2.1.2 Usage scenario. To illustrate how the users interact and experience this we present the following scenario. When a user first visits the wiki he or she is presented with the start page. This is intended to be the starting point of all pages and there should be no orphan pages (pages not linked to). The integrated awareness view then shows all activities within the last thirty days (Figure 2, top). The user can then zoom in to see what happened on a specific day to get more details and a less cluttered view (Figure 2, middle). By clicking on one of the events, for instance the one called 'mitra_artikel' the browser is redirected to the page for the wiki page named 'mitra_artikel'. The events for pages that are outside the scope of 'mitra_artikel' will then disappear from the awareness view, and only pages accessible from this page are shown (Figure 2, bottom).

2.1.3 Implementation. The interactive timeline was implemented using *Ajax* techniques [7]. This means that javascript on the client side is used to fetch data asynchronously from the server without having to reload the web page. The resulting application runs in the web browser without the need for any special applications or extra plug-ins, such as Java or Flash. This gives a significant advantage for material that is accessed on-line from several different computers and sites, as the only requirement is a web browser.

In order to render the timeline, the client needs data about the events. The data is fetched with an HTTP GET request. The response is XML-formatted data, which is easy to parse on the client. There are two types of data: *histogram data,* and *event data.* The histogram data gives the number of events for the last thirty days. This is typically only fetched once, when the page is being loaded. The event data is a list of all information about the events within a time interval. This data is fetched when the page is loaded, and whenever the user changes the time window. The server part is implemented in PHP to fetch the data.

Page Structure. On a wiki, the structure of all pages and content is usually flat. All pages exist on the same level and they are only connected when an explicit link is made between them. Some wiki software supports *namespaces*, which allow the content creators to arrange content logically. Using namespaces it is possible to organize the content hierarchically (e.g. the namespace Vehicles could have the pages Car and Truck in it), but requires the users to handle this explicitly. TikiWiki does not support namespaces, and therefore the structure we had access to was flat.

In the course, the students were divided into groups, and each group created their own group pages. These group pages would contain links to other pages with more information. Thus the way the wiki was used by the students created a somewhat hierarchical structure. We wanted to use this implicit structure in such a way that when a user viewed a certain page, only the actions on pages "below" this page were visible in the timeline. We therefore had to find a hierarchical structure from the way the pages were linked to each other.

Finding the Hierarchy. The algorithm for finding the hierarchy works by following links in a breadth-first search order and building a tree of pages. Starting from the front page of the wiki, all links on the current page are collected. For each link that has not been visited before (when traversing the tree) a new node, representing the target page, is created as a child to the current page node. This does not necessarily yield the most optimal (or logical) structure, but from experience of how pages are linked together it does give a good enough result.

We used this hierarchy to filter the events shown in the timeline, so that when a user looks at a certain group's page, the only events shown are those that have taken place on this page or pages linked from it and so forth. The result is that users will see only the specific group's activities. But it is also more general such that when looking at a specific report page for instance, a change to any page linked from it (not linked from a page further up the tree) will be shown, recursively.

2.2 Mobile Capturing Tool

We wanted to give the students a simple tool for capturing photos, videos, audio, and text when in the field, and also to be able to easily upload that material directly to the wiki. Since many mobile phones today have all of those capabilities – they have a camera, microphone, and a keypad for text input – and also have the treat of being a communication device – therefore being able to send data to a server – we decided to write an application for mobile phones as our tool.

The mobile application is built for smart phones running Symbian 3rd Edition operating system. The application allows a user to collect photos, videos, texts, and audio recordings (collectively referred to as data or data objects). When the data is saved it is automatically scheduled to be uploaded to a web server. As the target phones have GPRS, UMTS, and WLAN network capability, they are in practice always capable of uploading the data. When the data is collected (photo taken, audio recording saved, etc) the application also stores the last cell IDs observed in the previous thirty minutes. When uploading, besides the actual data object, the cell ID information, timestamp, and originating phone is also sent. The cell ID information is used for organizing the data objects and is described in the next section.

Most current mobile phones already have applications for recording video and audio, taking photos, and writing text. Initially, we intended to use the standard built-in applications, and extend them by for instance monitoring the directories where the applications store data, and when discovered upload the file together with the cell IDs. This would make the development more simple and rapid, and would allow the users to use the programs they already knew. However, this proved to be too inefficient as existing applications were too clumsy and users were often required to wait a considerable amount of time before a needed function was ready for use. Instead, we built an application from scratch containing the four functions of collecting each type of media.

The application has four views, one for each function. The user navigates between the views using left/right on the joystick. The information shown is only the most basic information required, such as how long a video recording is and how much longer it is possible to record before the memory runs out.

Uploading large amounts of data over the cellular phone network can be quite expensive. We therefore wanted to limit this by only automatically uploading data objects below a certain size. We set this size so that photos would always be uploaded instantly, whereas video or audio recordings would only be uploaded if the length is short enough. If the size of the object is too big it will instead be put in a separate queue for large files. In order to have the large files uploaded the user has to manually start the process and select which data connection on the phone to use. The idea here is that the user can upload big files when there is access to WLAN to avoid the cost.

2.2.1 Implementation

The application was implemented in C++ using Carbide.C++. The targeted phone model was the Nokia E70. The phone has a 2 megapixel camera, and a folded keyboard which makes it suitable for writing texts (see Figure 3).

The application actually consists of two programs; one background process which monitors cell movement, i.e. keeps track of when the cell ID of the base station changes, and records this in a database; and one GUI application which gives the core functionality to the user.

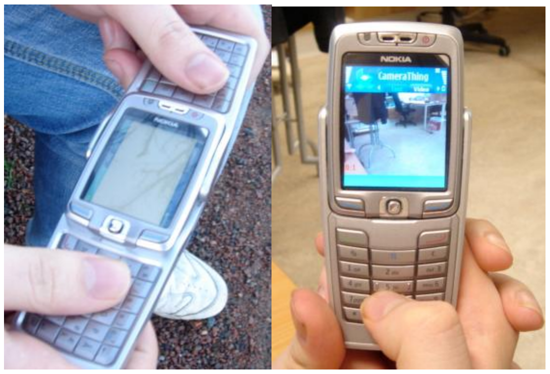


Figure 3 Mobile application. Writing text (left), and taking a photo (right).

A Symbian GUI application is recommended to follow certain architectures, which are supported with different base classes for application logic and GUI components. Our application uses a *view architecture* in which each view or function of an application are separated and easily invoked when needed. This fit well with our intended application, as we wanted four distinct functions.

The background application automatically starts when the phone is powered on, and is always running. It registers a callback to be acknowledged whenever the cell ID changes and stores this. This process also serves as an upload server that attempts to upload anything that is put on queue. The GUI application then issues commands through a custom API to get status information about uploads, and to put data in the queue for uploading.

The two applications communicate using a client/server model where the background application implements the server part and the GUI application issues commands as a client. This is the recommended style for Symbian programming.

2.3 Tool for Browsing Captured Data

Even if the photos and video etc. are uploaded automatically to the wiki, you still need to manage and organize them. The standard way is to have a file gallery where all data is put, and you have to refer to the object in the gallery in some way. The organization within the gallery is also very linear and simplistic showing the filenames and the time of upload, and it can be hard to find what you want. We therefore wanted to create a different mean of managing the data in the wiki, by building a browser which shows how the data objects are related to each other. Again we wanted to implement it as a web application to use the strength of being able to use it from any computer.

The web application lets the user browse objects arranged after either *time* or *location*. The two views differ in principle and are therefore explained separately below. The application is built using Ajax techniques in the same way as the Awareness addon and is run on top of the wiki itself. A button is added to each wiki page which invokes the Browser. The Browser is then brought up on top of the wiki page, occluding the wiki, and can be hidden by hitting a close button to bring back the wiki.

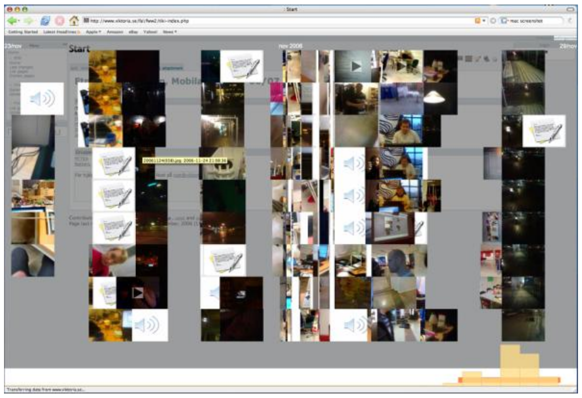


Figure 4. The browser application showing pictures ordered by time.

2.3.1 View by time. The web application that shows objects on a timeline is shown in Figure 4. The bottom of the view shows a number of bars. There are thirty bars representing the last thirty days, where the height indicates the number of objects collected on that day. This part also contains a narrow horizontal bar indicating the time interval for which the objects are shown above. The objects are arranged horizontally according to time, and spread out vertically to increase the visibility. On the very top of the application is a toolbar of options.

The browser is highly interactive and the aim was to make it easy to navigate through objects and quickly get an overview of what the collection contains. This is done by letting the user zoom in and out by simply dragging and clicking with the mouse. In order to change the view, the user can either zoom out (enlarge the time interval) by pressing the right mouse button or a left click while holding the ctrl-button down. In order to zoom in, the user can simply drag the mouse horizontally while holding down the left mouse button and select the time interval to show. Alternatively, by double clicking on one of the bars at the bottom, the view will zoom in on that particular day. The zooming is animated to create a smooth user experience, but the animation can be disabled if desired.

Even though it is very easy to zoom in, it is equally easy to zoom in by mistake or on an incorrect area. To mitigate this, the zoom levels are put on a stack when zooming in. When zooming out, the application will first check if the stack is empty, and if not it will zoom out to the zoom level on top of the stack. Thus, to recover from a zoom mistake the user can easily zoom out to the last zoom level.

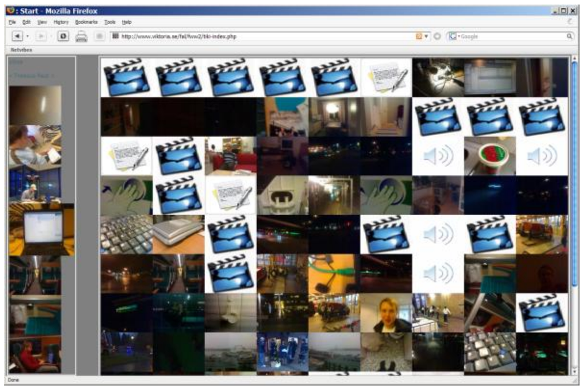


Figure 5. Location view.

2.3.2 View by location. As the collected data objects are tagged with a set of recent cell IDs, they can be related to each other in terms of *location*. In order to find photos from the same location as another object the view can be changed to a location view (shown in Figure 5). In the location view, thumbnails representing the objects are put in chronological order on the left, scrollable by dragging. By choosing an object (clicking the thumbnail), all objects considered to be captured "close" to it are loaded in a view to the right of it.

There are two ways to go to the location view. The user can switch to the view by pressing a link in the top menu bar. Or, in the object view, the user can click a link in the menu going to the location view and automatically highlight the object in the chronological list, and see all objects from the same location below. In this way it is possible to simply locate a photo, and then find all other photos taken at this location, but at the same time find photos related in time (maybe taken by other team members) as well.

2.3.3 Object view. By holding the mouse pointer over an object, some brief information is shown, such as who created the object. By clicking the object representation, the object view is shown (see Figure 6). In the object view, the object is shown to the right and information about it is shown on the left. For video and audio the object is loaded in a QuickTime plug-in (requires QuickTime to be installed on the computer). This allows the object to be examined easily. Furthermore, a link on the left can be used to add the observed object directly to the current wiki page. This allows the users to use the browser to find objects quickly and build wiki pages around them.

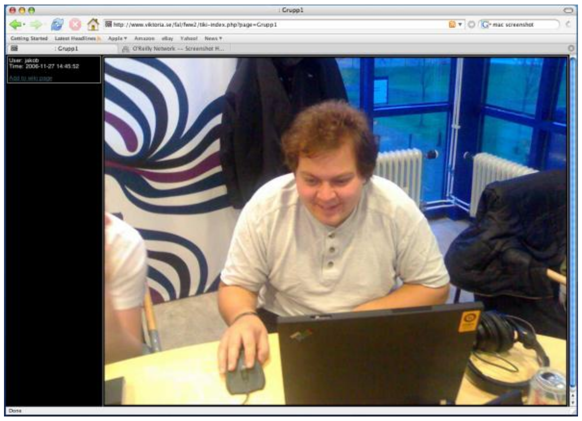


Figure 6. Object view.

2.3.3 Using cell IDs to decide location. For the purpose of the browser, all we need is to be able to decide whether two objects relate to the same location or not. The actual geographical location is of secondary interest and not resolved in our system.

However, to only use cell IDs to decide if they correspond to the same location is difficult. There are basically two reasons for this. First, cell IDs only give coarse-grained details about the location, as cells in the network topology can be rather big. Secondly, different algorithms for jumping between cells are used by network operators and a phone is seldom connected to one base station for too long, even when the phone is stationary. Thus, care must be taken when using the cell IDs to figure out location.

We use a similar algorithm as [11]. By comparing a sequence of the last seen cell IDs for two objects using a similarity measure we get a value for how similar the locations are. Defining a cut-off value thus allows us to make a decision on whether they are the same location or not. Thus, when looking for objects collected at the same place as another object, the similarity is calculated for all objects and the ones above the threshold are presented.

3. User Experience

The awareness visualization was designed based on our studies of the students' work practices during the first year the wiki was used. The second time the course was run (with a new group of students) we installed the awareness visualization and the browser extensions, and gave them a mobile phone each (to be used as their ordinary phone). Two student groups used the wiki and mobile phones to upload ethnographic field notes and other material.

As for the awareness visualization we found that the students mainly used it at the moment when they logged in, from the front page, to quickly get aware of what had happened since they last logged in. However, they rarely used it when deeper inside the wiki. This is probably because the number of pages they created (and also the number of students) was quite low, and their history could be grasped in the timeline on the front page.

They mainly used the phone application for taking photos and some videos, and the browser was used to review, find, and discuss photos and videos that they needed for their analysis. They reported that they found the mobile phone and application a very simple but powerful and helpful tool and appreciated the way material automatically got to the wiki without any extra effort. They also liked how it was visualized in the browser in the wiki, however they did not use the location view at all. The reason for this is most probably because of the way they collected the data, such that the timeline served as an implicit location divider, as they knew where they had been at different times. One important note here is that they always went out in groups, and never individually, meaning that all group members' data were gathered over the same time interval, and at the same location.

4. Related Work

ZoneTag [1] is an application for mobile phones that automatically uploads photos to the photosharing site Flickr (www.flickr.com). It uses cell IDs to tag the photos with location, and to suggest tags that the user might want to use, based on current location and previously used tags. If the location is not known, the user can specify the location on the ZoneTag web site. The location specified will propagate through the network of ZoneTag users so that other photos from the same location (identified by cell ID) will be named. Unlike ZoneTag, our intended use of cell IDs is not to simplify tagging, but rather to simplify the organization of material.

Meneses and Moreira investigated how cell IDs can be used to find a phone's location [11]. Instead of just the current cell ID, their algorithm uses a set of last seen cell IDs and their time stamp. In this way they are able to get a more precise location than by just assigning a cell ID with an area, since cells in the network topology will usually overlap. Furthermore, they use this to determine when a phone is stationary and familiar locations. We use a similar scheme to determine whether two pieces of data (e.g. photos) were created at the same location or not.

Several researchers have explored how people organize and identify photos. Rodden and Wood [13] showed that people find it beneficial to have their collections of digital photos in chronological order, as it is easier to remember when an event occurred relative to other events, than to remember its absolute occurrence. Rodden investigated how visual similarities between images can be used to browse through photos [12]. Cooper et al. used time as a way of clustering the images so that the clusters formed events [4].

A number of systems have been presented that visualize awareness in distributed workgroups. For instance, *AwarenessMaps* [8] supports awareness by visualizing activities in a web-based shared workspace system. The system consists of two parts. The first is *PeopleMap*, which showed the activities of users. The second is *DocumentMap*, which shows the current status of the content of the workspace. AwarenessMaps only shows activities within the last twenty-four hours; when a document is changed its representation is changed for twenty-four hours and is then changed back. Thus it does not give any sense of the history of the document. Another example is *YeTi* [16], an information sharing system for informal digital sharing over distances, which includes a history view for showing when and how information has been accessed by people at different places. The history view is a timeline showing the time and place where a piece of information has been accessed.

In software development, awareness is an important issue. This practice usually has a high degree of cooperation and the need to know the work of others is especially important. Storey et al [14] presented a framework for how to evaluate visualization tools that aim to support awareness in software development. One notable system is *Jazz* [9] (not to be confused with the zooming graphics toolkit of the same name [3]), a software development environment where an existing system, Eclipse (www.eclipse.org) was extended with functions for *contextual collaboration* [6]. The idea was to add functions and tools to the existing environment that the programmers were already using, in order to support collaboration unobtrusively. The added functions included both support for awareness and active communication channels such as chat.

An example of visualizing wiki activity is *history flow visualizations*, which were used to analyze the evolution of pages in WikiPedia [15]. History flow visualizations produce a visual map that shows how a page has been edited and by whom at what time. It was used to analyze the collaboration within WikiPedia and to understand what makes it successful. While history flow visualizations does give a good indication to what has happened to a page historically, it does not convey any information to what is going on in the wiki as a whole or support awareness of what other contributors are doing.

5. Discussion and Conclusions

We have presented an awareness tool, a mobile capturing tool, and a data browser, to aid students who are doing mobile fieldwork. The tools were deployed to students in a course at a local university. A major feature of this work is that the stationary applications actually run inside the web browser using Ajax techniques, and that the capturing tool runs on their mobile phones. It thus does not require any extra effort from the students besides learning to use the tools, rather than having to install and manage special software on their computers, or carrying around extra equipment when in the field – something that can be seen as intrusive by the people they observe.

We believe that there will be a widespread use of mobile devices to create and share collections of digital media. This project represents one approach to organizing such collections, and turned out to be useful in an educational setting, when student created and analyzed field notes. Also, the pain of heavy data costs for phone carriers are now rapidly disappearing with emerging flat rate subscriptions, and the transfer speeds are also increasing with speeds higher than most people had in their homes just a couple of years ago. In the future, we hope to generalize our application to other application areas, to support emerging practices such as mobile photo-blogging and other user-created content.

6. Acknowledgements

Thanks to our colleagues Johan Lundin, Barry Brown and Gustav Lymer. We also want to thank Kim Grahn Dahlberg for assisting in development. This work was part of the iDeas project, a collaboration with Stanford University supported by the Wallenberg Global Learning Network.

7. References

[1] Ahern, S., Davis, M., Eckles, D., King, S., Naaman, M., Nair, R., Spasojevic, M., and Yang, J. ZoneTag: Designing Context-Aware Mobile Media Capture to Increase Participation. In *Proceedings of the PICS workshop at UbiComp 2006* (Irvine, Ca, USA).

[2] Brown, B., Lundin, J., Rost, M., Lymer, G., and Holmquist, L. E. Seeing Ethnographically: Teaching ethnography as part of CSCW. In *Proceedings of ECSCW 2007* (Limerick, Ireland)

[3] Bederson, B., Meyer, J. and Lance, G. Jazz: an extensible zoomable user interface graphics toolkit in Java. In *Proceedings of UIST 2000,* ACM Press.

[4] Cooper, M., Foote, J., Girgensohn, A., and Wilcox, L. Temporal Event Clustering for Digital Photo Collections. *ACM TOMCCAP, Vol. 1, No. 3, Augusti 2005.*

[5] Dourish, P. and Bellotti, P. Awareness and Coordination in Shared Workspaces. In *Proceedings of CSCW 1992* (Toronto, Canada).

[6] Fontana, J. Collaborative Software Ages Slowly. In *Network World Fusion*, January 6, 2003.

[7] Garrett, J.J. *Ajax: A New Approach to Web Applications.* Adaptive Path, February 18, 2005. Available at www.adaptivepath.com/publications/ essays/archives/000385.php

[8] Gross, T., Wirsam, W. and Graether W. AwarenessMaps: Visualizing Awareness in Shared Workspaces. In *Extended abstracts of CHI 2003* (Florida, USA).

[9] Hupfer, S., Cheng, L., Ross, S. and Patterson, S. Introducing Collaboration into an Application Development Environment. In *Proc. of CSCW 2004* (Chicago, USA).

[10] Lymer, G., Lundin, J., Brown, B., Rost, M. and Holmquist, L.E. Web based platforms in co-located practice – The use of a wiki as support for learning and instruction. To appear in *Proc. of CSCL 2007* (New Brunswick, USA).

[11] Meneses, F., and Moriera, A. Using GSM CellID Positiononing for Place Discovering. *First Workshop on Location Based Services for Health Care (LOCARE'06)* (Innsbruck, Austria).

[12] Rodden, K. Evaluating similarity-based visualizations as interfaces for image browsing. *Ph.D. dissertation. University of Cambridge, U.K.*

[13] Rodden, K., and Wood, K. How do people manage their digital photographs? In *Proc. of CHI 2003.* (Fort Lauderdale, Florida, USA)

[14] Storey, M-A., Čubranić, D. and German, D. On the Use of Visualization to Support Awareness of Human Activities in Software Development: A Survey and a Framework. In *Proc. of SoftVis 2005* (St. Louis, USA).

[15] Viégas, F., Wattenberg, M. and Dave, K. Studying Cooperation and Conflict between Authors with *history flow* Visualizations. In *Proc. of CHI 2004* (Vienna, Austria).

[16] Yamada, T., Shingu, J., Churchill, E., Nelson, L., Helfman, J. and Murphy, P. Who Cares?: Reflecting who is reading what on distributed community bulletin boards. In *Proc. of UIST 2004* (Santa Fe, USA).