# POWERVIEW
# Using information links and information views to navigate and visualize information on small displays

Staffan Björk, Johan Redström, Peter Ljungstrand, Lars Erik Holmquist

PLAY: Applied research on art and technology
The Interactive Institute, Box 620, SE-405 30 Göteborg, Sweden
http://www.playresearch.com
{staffan.bjork, johan.redstrom, peter.ljungstrand,
lars.erik.holmquist}@interactiveinstitute.se

**Abstract**. PowerView is a PDA application designed to support people with situational information, primarily during conversations and meetings with other people. PowerView was designed to address a number of issues in interface design concerning both information visualization and interaction on small, mobile devices. In terms of information visualization, the system was required to provide the user with a single integrated information system that enabled quick access to related information once an object of interest had been selected. In terms of interaction, the system was required to enable easy and efficient information retrieval, including single-handed use of the device. These problems were addressed by introducing *Information Links* and *Information Views*. An evaluation of the application against the standard application suite bundle of the PDA, a Casio Cassiopeia E-11, proved the interfaces equivalent in usability even though the PowerView application uses a novel interface paradigm and the test subjects were given no training time with the system.

## 1 Introduction

The popularity of Personal Digital Assistants (PDAs) has increased rapidly in the last few years. One area where PDAs have become especially widespread is among mobile workers, as such devices give users access to digital information while on the move. However, the environment in which PDAs are used is often quite different from a typical office environment, in which many environmental variables can be predicted. In contrast, when designing the user interface for PDAs not only must one assume that the environment may lack comfortable working positions, have bad lighting and be distracting, but also that it may change during a single use session. For instance, using a PDA on a subway means that the lighting changes as the subway car moves between stations. Further, a slight shaking can be expected during the whole ride, and if one hand is occupied with holding a handle in order for the user to keep his or her balance, the use situation becomes even further distanced to the traditional office environment.

In addition to the constraints posed by the physical environment, the small size and form factor of PDAs introduce several new constraints for human computer

interaction design (cf. [19]). PDAs typically have much smaller screens, less computational power and memory, and perhaps most important, limited input facilities. Most PDAs rely primarily on stylus-based input using a touch-sensitive screen, something that demands the use of both hands. Further, the standard components of traditional graphical user interfaces, such as scrollbars, buttons and menus, which on desktop displays only take a small percentage of the available screen space, take up a considerable percentage of the screen space on PDAs, forcing a search for alternatives.

These new situations of computer use do not only create new interface problems regarding user interaction with applications, new requirements are also posed on how information should be visualized on the limited display area. With PowerView, we have explored the design issues of how information visualization techniques can be applied on PDAs, how information navigation can be facilitated in order to enable fast retrieval of information relevant to the situation at hand and how to allow single-handed use of PDA applications. In this paper we describe the PowerView application, the design issues and their respective solutions, as well as how the application works in practice.

## 2 PowerView

PowerView was designed to support the user with relevant information during activities such as conversations and meetings with other people. To do this, we designed an integrated interface to the most common kinds of information stored on PDAs, i.e., contact lists, emails, tasks and meetings. Even though PowerView technically is an application running under Windows CE, it was designed not to use any of the GUI components of Windows CE in its user interface in order to fully explore alternative interaction techniques. In doing so, we tried to avoid or minimize the use of widgets commonly used in large screen GUIs, e.g. buttons, menus, checkboxes and window managing operators (see **Fig. 1** for an typical screenshot of the application). A short description of the system has previously been published [2].

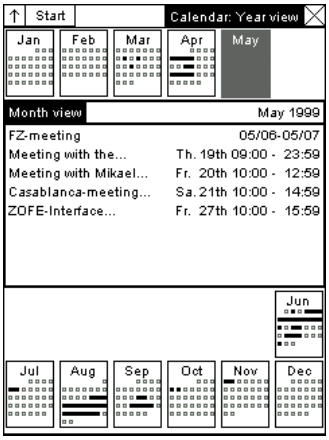


**Fig 1.** Typical screenshot from the PowerView application.

As different variants of PDAs provide different interaction possibilities in terms of display area, computational power, and input methods, the choice of device greatly influenced the design of the application. The PowerView application was implemented on a Casio Cassiopeia E-11 (see **Fig. 2**), being one of the more common PDAs on the market. It is relatively powerful, and it was used as it offered the possibilities of quick prototyping in the high-level object-oriented language Waba (a subset of Java). The device weighs slightly less than 200 grams and can be held by one hand. It is equipped with a touch-sensitive screen capable of displaying 240*320 pixels and 4 shades of gray, a 49 MHz NEC Vr4111 CPU, and 8 Mb RAM. Besides the use of a stylus for input, it has six buttons (excluding the on/off button) of which

two can easily be accessed when programming the device in Waba. One of the two buttons is an exit button, similar to the escape key on a keyboard. The other one is an "Action Control" (see **Fig. 2**) that can be used in three ways: to rotate upward, to rotate downward, or to select by pressing it (inwards). These operations correspond to using the up and down arrow keys and the enter key on a traditional computer keyboard.


**Fig. 2.** The Casio Cassiopeia E-11 with a close-up on the Action Control.

### 2.1 Design Issues

PowerView was designed to address a number of issues in interface design concerning both information visualization and user interaction on PDAs.

**Information Visualization.** Providing a user with as much information as he or she needs to perform a task is an almost ever-present problem when designing computer applications. Many information visualization techniques have been developed and claim to give users efficient views of information (e.g., [5,7,11,13,15]). This problem often becomes more difficult with PDAs, as users often need access to almost the same information as they do on their desktop computers despite having only a fraction of the space of an ordinary display available.

Several information visualization techniques have recently been applied to devices with small screens [4,18]. However, when techniques developed for desktop displays are to be used on PDAs, they often have to be modified to fit the new constraints posed by the small devices. This includes not only the limited display area (which may require designers to abandon ideals such as showing all available information), but also taking into account limited computational powers, memory space and changing networking capabilities.

**Information Navigation.** Regardless of how the information is visualized, PDAs need to use several separate views to present information that could be presented simultaneously on a device with a larger display. Increasing the resolution of PDA displays to show more information does not resolve this problem, since the limited size of the screen would make the presentation too small to be readable. Thus, users need to navigate between several different views on PDAs in order to access the same amount of information as displayed on a single view on a desktop or laptop. Each switch between two views requires the user to re-focus on the new information presented, and also requires the user to relate it to the previous information in order to make sense of it. These transitions adds an overhead cost to the interaction as the user must explicitly choose what view to switch to, make the switch, observe the effects of

the switch and make sense of the new information displayed. This added overhead takes time and concentration from the user's intended activities.

This overhead can partly be mitigated by closely mapping how the information is visualized with how the underlying information is structured. For example, if an address book can not show all entries at once, it is feasible to divide the presentation of contacts into groups where all names in a group starts with the same letter. In this way, if the user must navigate to find a contact, each change of view corresponds to moving from one letter to another.

When the number of items in a group becomes too large, or the number of groups becomes too large to be displayed simultaneously, further divisions are required, creating hierarchical structures of views. These structures require the user not only to switch between different views when moving between items, but also to move between different levels of views, as some views will be used to categorize other views. This creates further overhead, as the user is presented with the same information on several different levels of detail. This becomes most apparent when a user moves from viewing an individual piece of information to viewing another individual piece of information of another information type, as the user has to move from the bottom of a hierarchical structure to the top, switch to viewing another hierarchical structure and navigate to the bottom of that structure. A typical example of this problem is when switching between reading an email from a person to looking for that person's phone number in the contact list application.

While applications like Microsoft Outlook or the Active Desktop on Windows CE devices do integrate a few common types of information, they are still organized in a way that requires the user to explicitly switch between different views or modules in order to obtain the desired information. The PowerView application described in this paper aimed at taking this integration one step further to better support quick retrieval of related information once a certain interest, or focus, had been selected.

**Interaction Constraints.** One of the major differences between the use of desktop computers and PDAs is that many PDAs rely on stylus-based input. This has several implications for interaction design. First, input is slower than using a keyboard, which makes text-based operations, such as searching, less attractive. Second, when the user manipulates an object on the display with the stylus, the stylus and the hand holding the stylus covers parts of the screen. Third, if the user is walking, riding a bus, etc., the PDA is likely to be slightly in motion, which makes high-precision stylus-based manipulation on the screen more difficult to accomplish.

Ethnographical studies have shown that in many work situations, it is not feasible to require that users have both hands available for interacting with a device [10]. For instance, the user might be using a mobile phone with one hand while she is trying to retrieve information from the PDA. Various new input forms have been introduced that allow single-handed control, e.g. by using key cards [17], enhanced track point devices [9], or making a device tilt-sensitive [14]. However, all of these require new, or at least modified, hardware that have yet to become integrated in publicly available PDAs.

Interestingly, many commercially available PDAs already have buttons that can be used by the hand holding the device. Currently, such buttons are only employed to a limited extent, for instance to scroll in menus, as commercially available PDA user

interfaces require point-and-click interaction with the stylus for nearly all operations (an interaction style seemingly inherited from mouse operations on desktop computers). Instead of designing new input devices, we wished to take advantage of these already present buttons to explore single-handed interaction. The use of such buttons for navigation avoids some of the problems associated with stylus-based interaction, but requires that the navigation can be achieved using only a few degrees of freedom.

In order to improve usability in situations where the user only has one hand available for interaction, PowerView was designed to be possible to control solely using the buttons available on the PDA (see **Fig. 2**). This constrained the interface to be possible to control using four degrees of freedom only, corresponding to forwards and backwards, select/enter and exit/up.

**Context of Use.** One difference between stationary and mobile IT support is that while users have to bring their task or problem to the stationary computer, they can bring their mobile devices with them for use when and where they need them (cf. [12]). Since mobile devices such as PDAs rarely have the capabilities of stationary computers, they are not likely to be the complete solution to the users' problems. Instead, they are more of a support in activities where, ideally, the users' main focus is on the activity taking place rather than the technology supporting it. One implication of this is that applications on PDAs should be able to support activities while requiring as little attention (in the form of interaction) as possible, since the user may have focus on an activity outside the device. PowerView is not actively context sensitive in the sense of *context-aware computing* (cf. [16]), but adding sensors providing such functionality is an interesting option for future development.
As an illustration of this difference between stationary and mobile computing, we can think of the how users typically work with text on a PDA in comparison to a stationary computer: on stationary computers users often work with word processors in order to write full texts like this paper; on a PDA short notes during a meeting or a phone call are more likely. Even though both tasks could be accomplished on both platforms, this typical usage illustrates a basic difference in the design objectives of PDAs compared to desktop computers.

## 3 Using PowerView

In order to illustrate how the PowerView interface works, a sequence showing a typical interaction with the system is given in figures **Fig. 3-8**. In this example, the user wishes to find out what meetings are booked with "Mikael Goldstein". To do this, the user locates and selects the name in the Contact list. This creates a *Context View* that provides information linked to the person, including past and future meetings booked with him, email sent to and received from him, and tasks that are related to him. Within this Context View, the user then chooses to obtain more information about meetings, getting a detailed list of all meetings with Mikael Goldstein.

Initially, the user is presented with the *Overview*, in which information from all domains are visible in four different tiles (see **Fig. 3**). By clicking on one of the tiles with the stylus, the system sets that tile as focus tile and changes the visualization accordingly. Clicking on the focus tile with the stylus activates one of the *navigational views*, from which the user can navigate to individual data entries. To find Mikael Goldstein, the user sets the meeting tile as the focus tile (see **Fig. 4**) and activates the navigational view for meetings by selecting it again.

As soon as the meeting tile of the Overview is selected, the system switches to the Address Book View. This navigational view divides all contacts into tiles based on the first letter of the surname, representing the number of contacts in the context tiles as black lines, and the full names in the focus tile (see **Fig. 5**). Similar to the Overview, the user can move the focus between tiles by clicking on them or by using the action control. By moving the focus to the tile containing contacts with surnames starting with G, the contact Mikael Goldstein is identified (see **Fig. 6**). As the individual piece of information is now shown, it can be selected in order to switch to the Context View information view.

The Context View (see **Fig. 7**) visually resembles the Overview as it depicts information from all information domains in four separate tiles. However, the information shown in the context tiles is selected because it is linked with the object in focus. This gives a limited context containing only the information that the user has previously deemed relevant. In this view, the user can decide to look at one of the information objects in detail by moving the focus between the tiles. To examine the meetings associated with Mikael Goldstein, the user simply moves the focus to the Calendar (**Fig. 8**).

It should be noted that as all interaction in the example, and in the application as a whole, can be performed by exclusively using the Action Control (see **Fig. 2**). This ensures that every step can be performed using only the hand that is holding the device.

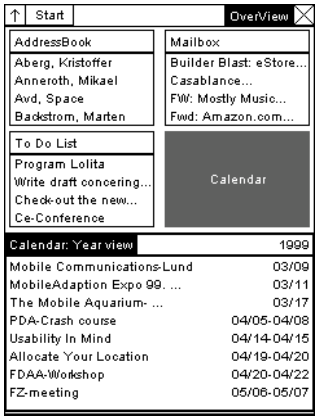


**Fig. 3.** The Overview, the initial view in the PowerView application.

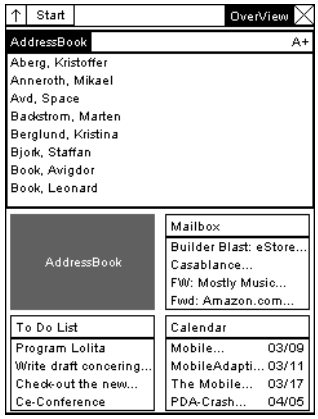


**Fig. 4.** The Overview with focus on the AddressBook.

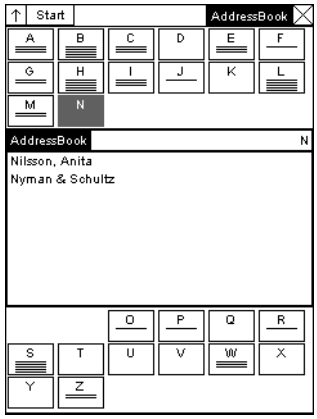


**Fig. 5.** Initial view in the Addressbook.

# 4 Interface Design

In order to create an integrated user interface for the PowerView application, all the design issues discussed previously had to be solved together. We addressed this by using *Information Links* and *Information Views*. By using the two concepts together, a unified presentation was created that at every point focused on supporting the user with information and allowed for a coherent way of navigation for all information types.

## 4.1 Information Links

In order to solve the problem of extensive information navigation to move between individual pieces of related information of different types, it was necessary to find a way of showing several different information types simultaneously. To do this, we needed a system to describe what information was related to a chosen piece of information. *Information links* was introduced to solve this problem.

Information links simply indicate a (semantic) connection between two pieces of information, where the two pieces of information can belong to different information domains. Information links enable the collection of various pieces of information that together form a heterogeneous context to an object (focus) the user selects, while still preserving a homogenous structure for every information domain. Thus, the information links form a semantic layer of relations between objects in the different data types on top of the storage of each data type. Information links differ from hyperlinks in that they are not used to traverse different presentations (e.g. web pages), but rather to define a context for any given piece of information. The strategy of using links or connections between objects to represent semantic properties is frequently used in other research field such as linguistics (e.g., WordNet [6] and other semantic networks).

The type of connection provided by an information link is consciously designed to be explicit, i.e. the user determines if two pieces of information should be linked together and can choose

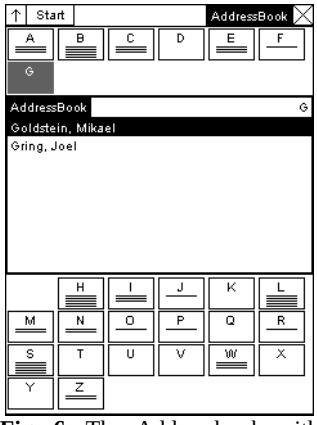


**Fig. 6.** The Addressbook with focus on the letter G.


**Fig. 7.** The Context view with the entry Mikael Goldstein selected.


**Fig. 8.** The Context View with focus on the Calendar.

any criteria for doing so. This makes the system, and the visualization, flexible and enables the user to adapt the visualization in some respects to the environments in which the application is used. While this makes it necessary for the user to perform additional actions in order to establish these links, this effort can be made in advance at the user's leisure.

### 4.2 Information Views

An *information view* is a collection of correlated objects displayed together to help the user with some activity. The objects are active in that they can change their appearance depending on the user's actions and are used to activate changes in the application, including switching to other information views. However, an information view should always be identifiable independently of the current state of the objects in it. The information views can present several different types of information, distinguishing them from (most) 'standard' applications, and are designed to function together with each other to support the user in more complex tasks that traditionally would have required the use of several different applications.

The information presented in an encyclopedia about individual countries can be used as an example of an information view, with the exception that it is a static presentation. Such presentations often give a collection of several different types of information, e.g., a map showing geographical data, a box containing demographical data, and a text body describing history, religion etc. These provide an informative overview of the country in which several different types of information are presented together.

As the PowerView application was designed to support the user with information during meetings and conversations, three categories of tasks, and thus information views, could be identified: showing what information was available on the device, selecting the information of interest, and presenting the selected information. Further, these tasks have to be completed in this order, which made the design of how the information views should be used together easier.

**Focus+Context Visualization.** Based on our experiences of working with the Flip Zooming visualization technique [8], including applying it to small screens [4], we decided to base each information view on a Flip Zooming visualization. Flip Zooming belongs to a class of information visualizations techniques called focus+context visualization. These are characterized by having one central object, the focus, presenting more detailed information, while simultaneously presenting contextual information in the surrounding area. Flip Zooming does this by dividing the information into a number of rectangular tiles of which one is the denoted the focus tile and the remaining context tiles. When creating Information Views using the Flip Zooming technique, each object in the Information View is simply mapped to a tile in the Flip Zooming visualization. The tile selected as focus is given most of the display area and the user can change which tile is in focus using random access methods such as a stylus or mouse. As the tiles are ordered sequentially, it is also possible to "flip" an object into focus by navigating "forward" and "backward". Having a focus tile

allows the user to have a more detailed presentation of information before changing to another information view by selecting it. This enables some exploration at every point without hiding tiles or immediately changing information view.

The Flip Zooming technique allows for hierarchical visualizations [1], i.e., the ability to use one information visualization within another. This allowed for a natural mapping of how the information in the application is structured to how it is visualized, with an information view at each node in the hierarchy. The tasks handled by the different information views had to be presented in a certain order. That order determined the order in the hierarchy, with the information view responsible for presenting the available information at the top and the information view presenting a chosen piece of information at the bottom. The hierarchical Flip Zooming technique is possible to navigate using only four operators, i.e., forwards, backwards, select and exit/up. Thus, it meets the requirement for single-handed use with the Cassiopeia device.

On ordinary desktop displays, it is possible to show all information visualizations used in a hierarchical Flip Zooming visualization simultaneously. For several reasons, this approach was not suitable for PDAs. Firstly, the limited display area of PDAs made it impossible to intelligibly show all information views simultaneously. Secondly, as each information view was designed to help the user with one particular activity, the information given in the surrounding information views would not add information vital for that activity, and thus distract the user and lower the usability of the application as a whole. By limiting the application to show only one information view at any one time, we gave up the idea of having a global context consisting of all information views, in order to have a local context for every view that helps solve the task associated with that view. It should be noted that this limitation differs from most applications, where the presentation of more information of one type is prioritized over the possibility of showing information of several different types throughout the application.

### 4.3 Description of the Information Views used in PowerView

For each category of sub-tasks, one or more information views were created. First, The OverView information view shows a summary of the information available from all four information types. Second, to allow the user to select a piece of information, one or more Navigational views (depending on the number of objects and detail in the information structure) were created to navigate within each of the four information domains. Third, the Context View is used to present the information selected by the user. (See **Fig. 9** for a model of how the information views are related.)

**OverView.** The presentation of the top level of the hierarchy is named the Overview view, as it presents information of all four information types. Here, four objects representing four separate types of information are presented: contacts, email, tasks and meetings. When the user selects one of the objects, PowerView switches to the navigational view corresponding to the information type of the object.

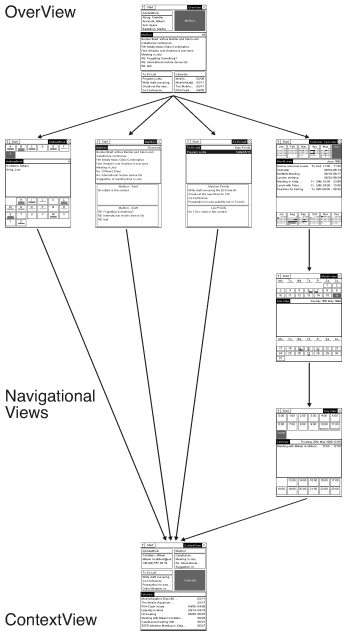


**Fig. 9.** Flow model of interaction with the PowerView application.

**Navigational Views.** After selecting information domain in the OverView, the user is presented with the corresponding navigational view. This view shows all available information in that domain in an abstract form, ordered into objects according to the nature of the information domain. By choosing one of these objects, the user can move to another navigational view that only presents the selected part of the information. Thus, each view helps the user with the task of choosing a region or an item of an information domain, and the navigational views as a whole provide the user with a structured navigation for selecting an individual piece of information.

As PowerView handles four information domains, four groups of navigational views were created. The information domain in the previous usage example was navigated by using only one navigational view. In cases where the information domain is structured into hierarchical structures, the user would have to move to other navigational views before being able to select a piece of information, and then switch to the Context View. For a flow scheme of such an interaction example, without the changing of focus in either the OverView or the Context View, see **Fig. 10**.

Depending on the amount of information and the structure of the information, different numbers of navigational views were required for each domain. In the case of meetings, different navigational views was created for handling years, months, days and hours, while for email only one navigational view separating received, sent and draft email was required. For the purpose of exploring information visualization on PDAs, we did not deem it necessary to create numerous navigation views for each information domain. In use situations where the amount of information requires more navigational views, these can easily be incorporated into the application when needed.

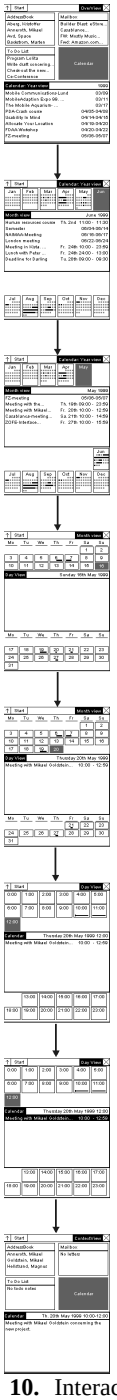

**Fig. 10.** Interaction flow scheme.

**Context View.** When the user selects an individual piece of information at the bottom level in a navigational view, the system switches to the Context View. In this view, the selected piece is displayed together with all information linked to it with information links. For instance, if a meeting is selected, the Context View shows information about people associated with that meeting, as well as email received from or sent to them and tasks that have to be done before the meeting. By mixing the information domains in this fashion, the problem of having to navigate through all the information in the system to move between two related pieces of information of different types was reduced.

As the types of information displayed vary depending on which piece of information the user selects, the Context View was designed to be able to show individual pieces of information from all domains simultaneously. This had the added benefit that the same Context View (with different information) could be used from all navigational views.

## 5 User Evaluation

In order to evaluate the application, the PowerView interface was benchmarked against the standard application bundle in Windows CE. Sixteen paid university students (10 women and 6 men, aged 17-43) were given 7 tasks to be performed on both systems in two different user situations. The main difference between the two situations was that in one the users were told to hold a mobile phone while performing all tasks, thereby inclining the users to use single-handed navigation. None had any prior experience of a PDA but all were familiar with using the Windows operating systems on stationary computers. The experiment was conducted at the Usability Lab at Ericsson Research in Kista, Sweden.

Both systems passed the set usability criterion, completion of 70% of all tasks. In one of the two user situations, 100% of the tasks were solved using the PowerView interface, while only 75% of the tasks were solved using the Windows CE bundle. Although not statistically proven, this indicates that PowerView provides more efficient usability. Further, the evaluation showed that the users perceived that the arrangement of information was significantly better on the PowerView application ($F[1,15]=8.497$, $p=0.011$). Although users received no description of the PowerView interface, and were only allowed six minutes to freely familiarize with the interface before the experiment, no significant difference between task completion time could be found. The main problems identified with the PowerView interface was that users tried to double-tap (similiar to double-clicking in standard graphical window systems), sometimes became confused over which information view they were viewing, and mistook “grayed-out” areas for buttons.

Surprisingly, none of the users utilized the single-handed navigation offered by the action control, despite situations where two-handed navigation led to physical discomfort (e.g., holding a mobile phone by the neck).

## 6 Concluding Remarks

PowerView allows mobile users to access information using an interface designed to work in a supportive rather than attention-demanding fashion. The use of information links and information views offers a new solution to the presentation and navigation of information on devices with small displays, breaking away from the traditional concept of using one application for each information type.

PowerView also supports single-handed navigation and retrieval of information in the entire application due to the restricted degrees of freedom in the interface. Although this was a feature that subjects in the evaluation did not employ, we have argued that single-handed navigation may be necessary in some use contexts since users may have the other hand occupied. Despite the restricted degrees of freedom in the interaction process, the usability evaluation showed that the PowerView application was equivalent in usability for completely novice users in comparison to the Windows CE application bundle.

The use of information views and information links in applications has only briefly been explored in the work described here. Future work is needed to fully validate the generality and usability of these concepts. Considering PowerView, the evaluation identified several possible improvements regarding both interaction and visualization, which will be addressed in future research. Further, PowerView has been identified as a possible basis for an application to organize communication and information together, which will also be explored in the future.

## 7 Acknowledgements

The authors thank Mikael Anneroth, Magnus Hellstrand and Mikael Goldstein at the usability lab at Ericsson Radio Systems AB, who performed the evaluation. This research was funded by SITI, the Swedish Institute for Information Technology, as part of the Effective Display Strategies for Small Screens project within the Mobile Informatics research program.

## 8 References

1. Björk, S. Hierarchical Flip Zooming: Enabling Parallel Exploration of Hierarchical Visualizations. In Proc. of AVI 2000, pp. 232-237, ACM Press, 2000.
2. Björk, S., Holmquist, L.E., Ljungstrand, P., and Redström, J. PowerView: Structured Access to Integrated Information on Small Screens. In Ext. Abstracts of CHI'2000, pp. 265-266, ACM Press, 2000.
3. Björk, S., Holmquist, L.E. and Redström, J. A Framework for Focus+Context Visualization. In Proc. of IEEE Information Visualization '99, pp. 53-57, IEEE Press, 1999.
4. Björk, S., Holmquist, L.E., Redström, J., Bretan, I., Danielsson, R., Karlgren, J., and Franzén, K. WEST: A Web Browser for Small Terminals. In Proc. of ACM UIST '99, pp. 187-196, ACM Press, 1999.

5. Card, S.K., Mackinlay, J.D., and Shneiderman, B. Information Visualization. In Card, S.K., Mackinlay, J.D., and Shneiderman, B. (Eds.) Readings in Information Visualization: Using Vision to Think, pp. 1-34, Morgan Kaufmann Publishers, San Francisco, California, 1999.
6. Fellbaum, C. (ed.). WordNet; An electronic lexical database. MIT Press, 1998.
7. Furnas, G.W. Generalized Fisheye Views. In Proc. of CHI '86, pp. 16-23, ACM Press, 1986.
8. Holmquist, L.E. Focus+Context Visualization with Flip Zooming and the Zoom Browser. In Ext. Abstracts of CHI '97, ACM Press, 1997.
9. Kawachiya, K., and Ishikawa, H. NaviPoint an input device for mobile information browsing. In Proc. of CHI 98, pp. 1-8, ACM Press, 1998.
10. Kristoffersen, S., and Ljungberg, F. "Making Place" to Make IT Work: Empirical Explorations of HCI for Mobile CSCW, In Proc. of Group'99, ACM Press, 1999.
11. Leung, Y.K., and Apperley, M.D. A Review and Taxonomy of Distortion-Oriented Presentation Techniques. In ACM Transactions on Computer-Human Interaction, Vol. 1, No. 2, pp. 126-160, ACM Press, 1994.
12. Norman, D.A., The Invisible Computer, The MIT Press, Cambridge, Mass., USA, 1998.
13. Rao, R., Pedersen, J.O., Hearst, M.A., Mackinlay, J.D, Card, S.K., Masinter, L., Halvorsen, P-K., and Robertson, G.G. Rich Interaction in the Digital Library. In Communications of the ACM, Vol. 38, No. 4, pp. 29-39, ACM Press, 1995.
14. Rekimoto, J. Tilting operations for small screen interfaces. In Proceedings of the ACM symposium on User interface software and technology (UIST) '96, page 167, ACM Press, 1996.
15. Sarkar, M., and Brown, M.H. Graphical Fisheye Views. In Communications of the ACM, Vol. 37, No. 12, pp. 73-84, ACM Press, 1994.
16. Schilit, B., Adams, N., Want, R. Context-Aware Computing Applications. In Proceedings of Workshop on Mobile Computing Systems and Applications, pp. 85-90, Santa Cruz, Ca, U.S, IEEE Computer Society, 1994.
17. Sugimoto, M., and Takahashi, K., SHK single hand key card for mobile devices. In Proc. of CHI '96, pp. 7-8, ACM Press, 1996.
18. Taivalsaari, A. The Event Horizon User Interface Model for Small Devices, Technical Report TR-99-74, Sun Microsystems Laboratories, 1999. Available at http://www.sunlabs.com/technical-reports/1999/
19. Want, R, Schilit, B, Adams, A, Gold, R, Petersen, K, Goldberg, D, Ellis, J. & Weiser, M. The ParcTab Ubiquitous Computing Experiment. Technical Report CSL-95-1, Xerox Palo Alto Research Center, March 1995.