A Context-aware Collaborative Presentation System for Handhelds*

Marcelo G. Malcher
Department of Informatics
PUC-Rio
R. Mq. de S. Vicente, 225
22453-900, Brasil
marcelom@inf.puc-rio.br

Markus Endler
Department of Informatics
PUC-Rio
R. Mq. de S. Vicente, 225
22453-900, Brasil
endler@inf.puc-rio.br

Abstract

Many systems support sharing and co-edition of slides, and thus enable active engagement and collaborative interaction among users of portable devices during lectures or meetings. The majority of these systems, however, run only on larger and more resource-full devices, like notebooks or tablet PCs. Handhelds, such as pocket PCs and smart phones, are becoming cheaper, lighter and with increasing computational power and storage capacity. Therefore, these devices are now also equally eligible for adoption in interactive classes, in the same way as notebooks and tablet PCs are used. In this paper, we present a collaborative presentation system named Interactive Presenter for Handhelds (iPH) which supports sharing and co-edition of slide presentations, and which runs on both tablet PCs and handhelds. We also report the findings of performance tests and usability experiments with iPH on handhelds in an exercise class.

1 Introduction

The ongoing improvement of portable devices and the increasing ubiquity of wireless networks enable the development of services and applications for any-place-any-time collaboration among mobile users in many different environments, such as at home, in public areas, in universities, in companies, among others.

Several research works [9, 13] suggest that the use of portable devices in classrooms improves the student's motivation and engagement by increasing the interaction with the instructor and other students. Such interaction is strengthened by presentation systems that support the learning process during class by changing the student's perspective from a passive participant to an engaged and active role. The traditional teaching model, where the instructor simply gives the lecture based on previously prepared and 'static' slides is being surpassed by a more interactive model, sometimes also called active learning.

A classroom for active learning is a heterogeneous computing environment, since the instructor and the students may use different devices like tablet PCs, notebooks and handhelds. These devices differ in size, computational power, storage capacity, input forms, among others. Handhelds, such as wireless-enabled PDAs or smart-phones are becoming more powerful and more accessible to the end user. These devices are cheaper, smaller and lighter than tablet PCs/notebooks, thus making

*Work supported by Microsoft Research and by Brazilian National Research Council (CNPq), grants 479824/ 2004-5 and 474188/ 2007-8.

them perfectly eligible to use in classrooms by students. A collaborative presentation system to be used in classrooms is expected to be executable on such heterogeneous environment, and to offer a similar user experience to students with handhelds and tablet PCs, i.e. that all students have the same functionalities, independent of the devices they are using. The problem is that most presentation systems are resource hungry and therefore require a full-fledged execution environment, which only runs on heavy and bulky notebooks or tablet PCs. This limits the general usage of these systems in classrooms, since all students must have such a device.

The primary goal of our work was to implement a simple-to-use system for sharing and coedition of slide presentations, which could execute on tablet PCs and handhelds enabled with wireless (WLAN) interfaces. We called this distributed collaborative tool the Interactive Presenter for Handhelds (iPH)[1]. With iPH we intended to extend the benefits of collaborative presentation systems to a heterogeneous set of mobile devices, i.e., allowing iPH to be executed on notebooks, tablet PCs and handhelds. In order to ease and improve the user experience, we made iPH also a context-aware system - it may modify its functionality according to location and computational context information (e.g., the state of system resources and connectivity). For example, iPH uses location information to help users to connect to the classroom-specific collaborative session, or to adapt some of its functionalities according to it, such as enabling or disabling some collaboration capabilities.

Using iPH with tablet PCs and PDAs we tested the performance of some common operations used during classes to measure the differences between the system's responsiveness, and to assess whether iPH on handhelds is in fact adequate for practical use. We also made a classroom experiment where several students with handhelds executing iPH interacted during an exercise class.

The next section summarizes and discusses related collaborative presentation systems. In section 3, we present iPH and its underlying concepts. In section 4, we describe the performance and the usability tests performed and the results obtained. Finally, in section 5 we discuss some current limitations and future work.

2 Related Work

There are several projects doing research on improving the interaction between users holding portable computing devices in a classroom, or in small collaborating groups. These projects have common features like the support for digital ink, and can be split into two different approaches: the classroom (or instructor-based) approach, where the collaboration and interaction among the participants is similar to the one in a classroom with an instructor - who controls the presentation - and the students; and the brainstorming (or symmetric) approach, where all participants share a common virtual space/board and freely interact through it, hence having identical roles (e.g. peer collaborators).

In the first group, we highlight two projects: Classroom Presenter and DyKnow Vision. Classroom Presenter [2] is a distributed presentation system for tablet PCs, where a presentation is synchronously shared across multiple machines. It supports integration of computer-generated slides (e.g. produced with a presentation editor) and digital ink (e.g. manually made texts and drawings), giving the instructors flexibility to present learning content and interact with the audience. Instructors may also initiate polls with students and receive and display their answers. This system was developed at University of Washington in cooperation with other universities, and was extensively used in several undergraduate-level classes with high level of acceptance [1]. However, Classroom

[1] iPH is ready for download and use at http://www.lac.inf.puc-rio.br/iph

Presenter is tailored to resource-rich devices like Tablet PCs and does not run on handhelds. This limitation motivated the development of iPH, which shares a similar collaboration model than the one of Classroom Presenter. DyKnow Vision [3] is nowadays a commercial product, and has many additional functions besides the sharing and co-edition of slides, such as instant messaging and co-navigation of Web pages. Like Classroom Presenter, this system also supports presentation slides and polls to participants, and has other interesting functions like the possibility to check each participant's status, e.g. if a student is following the presentation or is doing other things with his/her device. Like Classroom Presenter, this system does not have a version for resource-poor devices, such as handhelds.

In the second group, i.e. the brainstorming/symmetric approach, there are some collaborative presentation systems which run on handhelds, such as the Peebles project [7], IdeaLink [12] and SharedPad [4]. However, the last two systems were created only as prototypes for evaluation of the underlying middleware used for device interactions. All these systems have limited sharing and synchronous communication capabilities. Finally, it is worth mentioning also Livenotes [5], a system that supports the sharing and co-edition of presentation slides without an explicit master role, but which also executes only on resource-full devices.

3 System Description

The Interactive Presenter for Handhelds (iPH) is a distributed system which supports sharing and co-edition of slide presentations, and which runs on resource-full devices executing Windows XP/Vista and handhelds executing Windows Mobile.

iPH supports a classroom approach where one of the participants assumes the role of an instructor. The instructor is the one who controls the presentation, selecting and broadcasting the slides, and asking the other participants to contribute at particular points of the presentation. For example, the instructor may present a slide with a question to be answered by all students. Each student answers the question making changes to the presented slide and sends to the instructor - iPH supports input both in form of digital ink or typed text, which is printed over the image of a slide. Then, the instructor can evaluate the classroom's answers and choose some of the student's contributions for display to all participants and collective discussion.

It can be used for PowerPoint presentations or any tool that generates JPEG images. When using it for a PowerPoint presentation, it has first to be converted by iDeck, our support tool that converts each slide into a JPEG image and assigns them specific IDs. Hence, the presentation's animations are not preserved during this conversion. With iDeck, it is also possible to add blank slides, set its background color, title and add comments.

Our system comes in the XP and the Mobile versions. Our goal was to create the same user experience for participants holding tablet PCs or handhelds, i.e., not only should every functionality be available in both iPH versions, but also their usage mode should be similar and consistent. Hence, the only difference between the XP and Mobile versions is the layout of graphical user interface (GUI). In the XP version it is possible to simultaneously display the thumbnail-view of slides and the slide-view. In the Mobile version, these views are split into two different screens. Due to the special form and limited size of the handheld display, some buttons are also placed at different positions. Figure 1 shows the iPH XP and iPH Mobile versions, respectively.

iPH on three main capabilities: multicast-based synchronous collaboration, ink-and-text slide overwriting, and context-awareness. For collaborative commands, e.g. commands used to connect to the collaborative session or to send and receive messages containing slides and contributions,

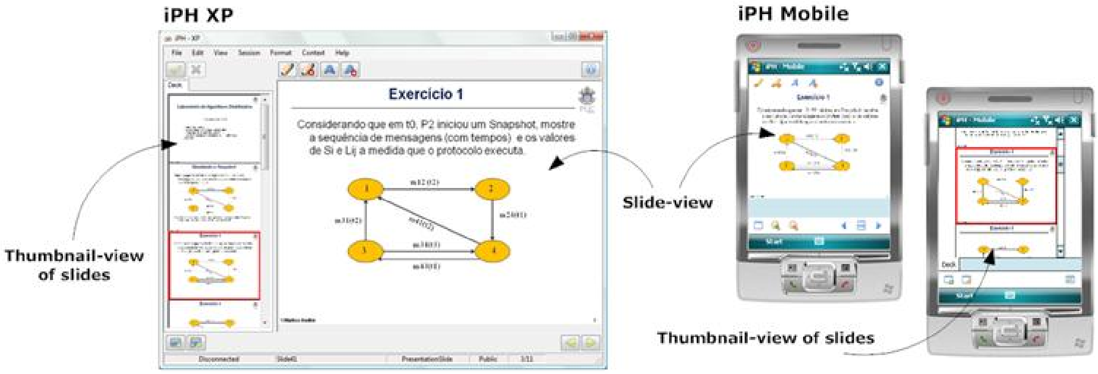

Figure 1. iPH versions

iPH is built on top of the Compact Conference XP (CCXP), our port of most components of the ConferenceXP API [10] to the .NET Compact Framework [6]. The ConferenceXP platform is a middleware for multipoint communication and synchronization that enables developers to create distributed applications without having to deal with low -level communication and connectivity control. For digital ink capture and text insertions we developed an independent component called LAC.Contribs, which controls every contribution made in a visual panel and can be used in both .NET platforms. This component allows formatting the ink color and width and the text font, and also has erasing functionality for both ink and text.

Regarding to context-awareness, iPH requests system context information (e.g., the device's energy level, free memory, quality of the wireless connection, etc.) and symbolic location information (e.g., device entered/exited room/building floor) to MoCA [11], a service-based middleware that supports the development of context-aware collaborative applications.

3.1 Concepts

In this section, we briefly explain the main concepts underlying iPH's design, some of which are similar to the ones of other collaborative presentation systems.

3.1.1 Session

Every user needs to join a collaborative session in order to share and be able to add contributions to the current presentation. Due to the middleware that we chose to build iPH, Compact Conference XP, the concept of session is tightly coupled with a multicast address used to distribute events and data among the devices running iPH. To join a session, every user must inform a multicast address, a port number, a user name and specify his/her role, which defines the functions that will be available for the user. If needed, the user may type a key to protect his/her messages. Every message sent by a user contains the key provided when connecting. It is worth mentioning that the key does not block the user from connecting to a session or to receive/send messages from/to other users. It only ensures that messages containing a key equals to the key used to connect to a session will be processed. Otherwise, these messages will be discarded.

Although the use of multicast communication facilitates the deployment of iPH, i.e. there is no need of any kind of previous configuration, it is not user-friendly to require the user to explicitly enter the multicast address and the port of the desired session. To improve the usability of our system, we designed iPH to make the action of connecting to a session be location-aware. When iPH detects - via its interaction with the symbolic location inference service of MoCA - that the user entered a new region, e.g. a student entered a classroom, it automatically queries a central server in order to discover the multicast address and port associated with the ongoing session in the corresponding region. Then, iPH informs the user that he/she can connect to a collaborative session through the discovered multicast address and port. Figure 2 shows both ways to connect to a collaborative session.

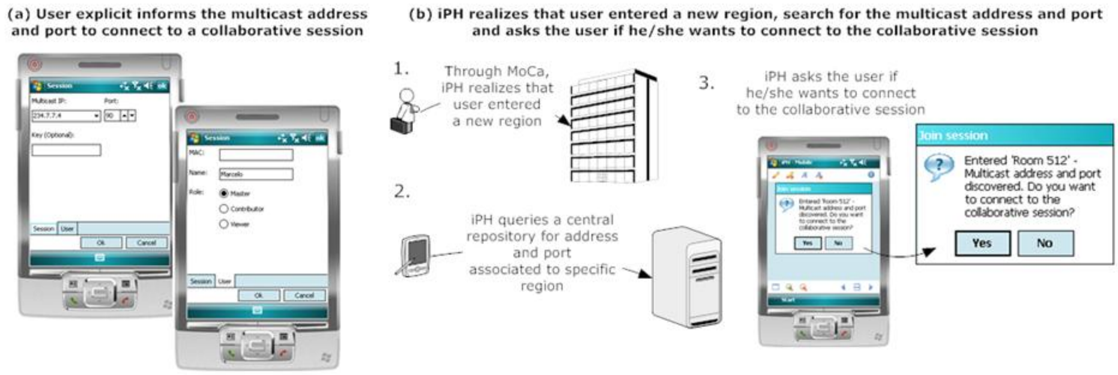

Figure 2. Connecting to a collaborative session

3.1.2 Roles

In iPH, there are three roles: master (instructor), contributor (students) and viewer (computers connected to the video projector).

The *viewer role* is a passive component with no selection and no editing capabilities. It is used only for displaying the instructor's slide-view panel, except for private slides. The *contributor role* enables the user to view the entire set of slides; select/view some slide of the set; add typed text, handwrite or draw (with the pen) on the current slide; send this contribution to the master; switch synchronization ON/OFF with the current master's slide, and display the device's current state of context variables. Finally, the *master role* is the most complete one. In addition to most of the functionality of the contributor role, such as typing or drawing on the current slide, it allows loading and broadcasting the entire presentation (i.e. the original set of slides without the student's contributions) to be shared, displaying contributions submitted by the students, requesting students to re-synchronize with the current slide, creating new public or private slides, and toggling ON/OFF the synchronization between master and viewer devices.

3.1.3 Synchronization

Synchronization control is a tricky issue in any distributed presentation system, as it directly impacts the system's usability. We designed iPH to handle the master-viewer and the master-contributor synchronization controls separately.

The master-viewer synchronization determines when the projector will display the master's screen. Hence it is reasonable to keep this control with the instructor. Therefore, at the GUI of the master device, there is a button that allows the instructor to toggle between synch ON and synch OFF states.

Regarding the master-contributor synchronization, iPH guarantees that, whenever a student starts creating a contribution at a contributor device (e.g. using the pen- or text-based input), this device becomes *out-of-synch* with the master device. This guarantees that a student is not disrupted (and disturbed) during the process of producing a contribution. However, at any time, the student may re-synchronize again with the master. The student also has the option to explicitly de-synchronize his device with the master's device. When master and contributor devices are out-of-synch, the instructor can only request the re-synchronization of the devices. This request translates into a 'synch request icon' at the contributor's GUI, and can be attended or ignored by the students. Figure 3 shows a typical configuration of iPH's usage, where the master device is at the center, and the red arrows suggest that the mastercontrols the synchronization with the viewer, whereas the contributors individually control the synchronization with the master.

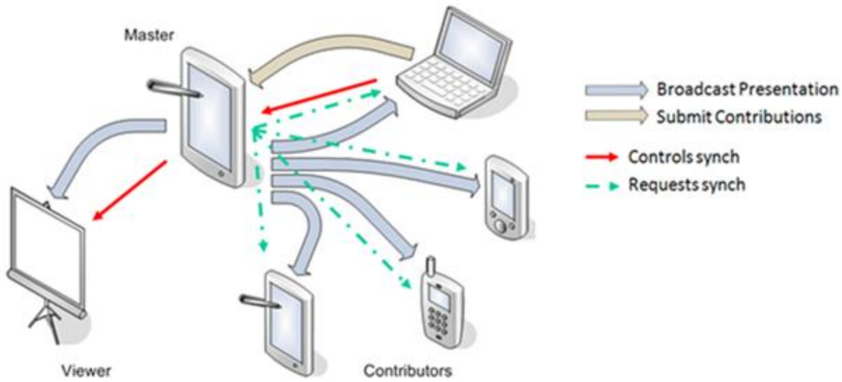

Figure 3. iPH synchronization control between roles

3.1.4 Slide Deck and Contribution

The slide deck is the sequence of slides that is distributed by the master to the other devices. Each slide carries a unique identification and also the information of its position in the sequence. During the collaboration, the instructor can create and introduce new empty slides at any position within his deck.

A contribution is any ink- or text-based input generated by an instructor or a student. Contributions can only be made on the slide-view window over the image of the slide, i.e. there is no extra annotation area around the slide, as in Classroom Presenter. In iPH, the contributions on a slide are recorded and displayed exactly in the order in which they were produced. Whenever a student starts handwriting/drawing or typing on a slide, this becomes a local annotation, until he/she explicitly submits the modified slide to the master. But when the instructor draws a stroke or types in some text, this contribution is automatically and instantaneously distributed to the other devices.

3.1.5 Context-Awareness

Besides using context information to help users to discover and connect to a collaborative session, each participant can access computational context information from its own device (i.e. available resources, such as energy level, connectivity or his symbolic location) at any time. The instructor is also able to access context information from any other device.

We also developed iPH to enable context- and location- specific adaptations of the presentation system, such as dynamically and automatically enabling or disabling some of its functionalities. This is achieved by self-adaptive methods based on rules created by the instructor. For example, the instructor may want to avoid receiving contributions from devices outside the classroom, i.e.

assuming that only the students/devices within the classroom are capable of giving enriching contributions. For this purpose, the instructor would create a rule associating the context state *"device inside room512"* to the iPH function for submitting contributions to the master. Then, whenever a device is detected outside room 512 (by MoCA's [11] symbolic, indoor location inference service, LIS [8]), the submit button at the contributor's GUI of this device would appear gray (disabled), but would return to normal state whenever the device is again detected inside the room 512 (see Figure 4).

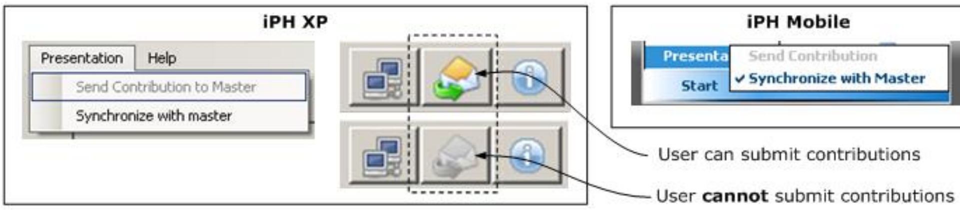

Figure 4. Functionality disabled due to a context rule

4 Tests

The main goal of our tests was to check if it was viable to run iPH on small devices like pocket PCs and smart phones during a class, where the performance of inter-device communication, device's energy consumption and usability of the system are of utmost importance for guaranteeing a seamless (and non distractive) integration of this technology into the classroom activities.

4.1 Performance

We focused our performance tests on three regular operations of the iPH systems: the broadcast of a slide deck by the master; the synch command sent to the contributors for visualizing the current master's slide; and the submission of a set of contributions by a student to the instructor.

In our tests we used one desktop computer connected through Ethernet, as well as two tablet PCs (Toshiba Protegé M400), one pocket PC (HP iPAC HX2400) and one smart phone (HTC Qtek 9100), all wirelessly connected at a WiFi (IEEE 802.11b) Access Point with multicast support.

Figure 5 shows the latency of broadcasting an entire slide deck to the student's devices by an instructor. The results suggest that devices with similar computational power, like desktops and tablet PCs, have also similar performance, independently of whether they are connected through wired or wireless links. This finding is re-enforced by analyzing the broadcasting results for pocket PCs and smart phones. The first ones have more computational resources than the second ones and, therefore, achieved better results. Despite these differences, even the results in the worst case, i.e. the broadcast of a 3MB slide deck (approximately 60 slides) to smart phones, can be considered satisfactory for usage during class, seen that this operation is performed only once during the class (in the beginning, when the instructor sends the slides to all students).

In the second set of tests we estimated the student device's response time to the synch commands sent by the instructor. As shown in Table 1 the response times can be considered quite satisfactory (i.e. only a fraction of a second), and with practically no difference between the devices. This positive result was in line with our expectations, since the message for the synch command only holds the identifier of the slide to be visualized by the students (i.e., the master's current slide).

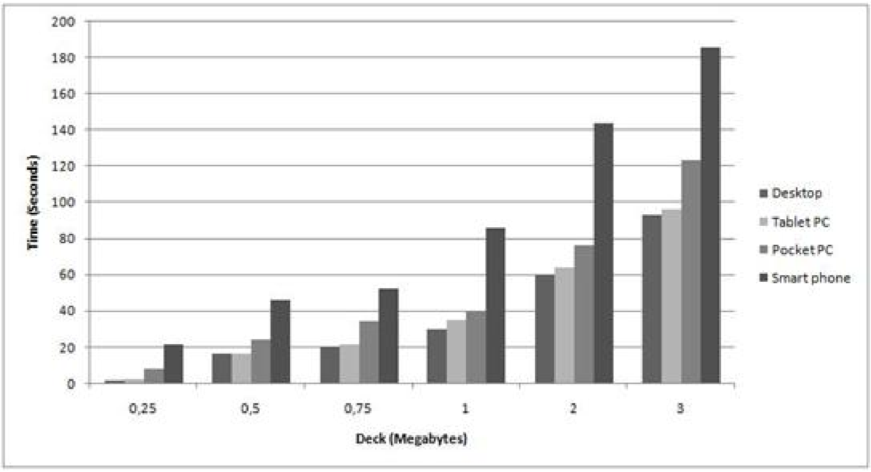

Figure 5. Latency of broadcasting a slide deck of various sizes

Device	Time (seconds)
Desktop	0.14
Tablet PC	0.21
Pocket PC	0.23
Smart phone	0.24

Table 1. Synch command latencies

Finally, the tests with students submission to the instructor showed that computational resources are the determinant for the performance of this operation. As with broadcasting the slide deck, resource-richer devices like desktops and tablet PCs achieved much better results. However, despite taking longer to submit contributions, handhelds can perform this operation in acceptable time, especially when considering that the instructor will most likely be receiving multiple contributions and thus will need some time to check out each of them. Table 2 shows the results of these submission performance tests.

Device	Time (seconds)
Desktop	0.4
Tablet PC	0.6
Pocket PC	3.0
Smart phone	3.9

Table 2. Latency of contribution submissions

4.2 Energy consumption

While performing the tests above, we realized that the devices' energy consumption is also a key factor for the usability of iPH, i.e.it is extremelly important that applications at mobile devices can be used during long periods of time, like classes, without the need to recharge the battery.

Therefore, we performed a test to check how long iPH runs in tablet PCs and pocket PCs until these devices' battery is entirely drained. During the test, the devices periodically executed actions like joining a collaboration session, submiting a contribution, requesting synchronization, among others. As shown in Figure 6, after three hours of execution, the tablet PC's remaining energy level dropped to approximately 10%, while the pocket PC's energy level was around 60%. These values show that iPH is viable in classes that last for approximately three hours.

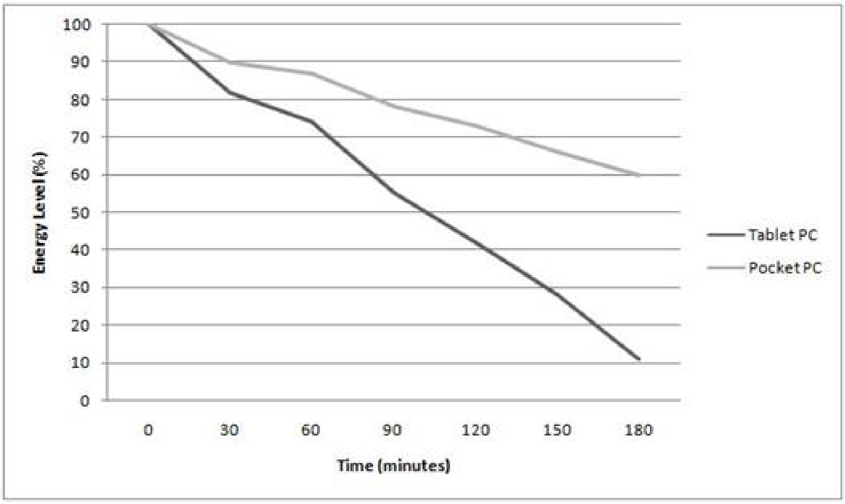

Figure 6. Energy drain on Tablet PCs and Pocket PCs

4.3 Classroom experiment

We evaluated our system in an exercise class (subject: Distributed Algorithms) with 6 postgraduate students using HP iPAC HX2400 PDAs running iPH. In the class we presented to the students several small exercises covering a significant range of topics that had been taught in previous theoretical classes. For each exercise, we gave the students approximately 10-15 minutes to draw or write their solutions over a base slide (e.g., which had some initial drawing like the one shown in Fig. 1) and send them to the instructor. Then, we displayed each of the student's contributions, discussed them in class, and by such reviewed the main concepts and algorithms, and explained some issues that had remained unclear.

Since we knew that the size and the resolution of the PDA displays is significantly smaller than the 15 inch screens of tablet PCs and notebooks, we already prepared the slides taking following precautions, which turned out to be indispensable:

- We used font size of at least 24pt;
- We avoided italic font style, since it is less readable on small resolution screens;
- For the same reason, in diagrams we draw lines with width of at least 2,25pt;
- On the slides, we left enough space around a central figure for the addition of contributions.

Although the experiment was realized with only few students, it was possible to identify some necessary improvements to our system. The main complain was about the display's size of handhelds. Creating easier ways to navigate through slides and to make contributions on them thus seems to be a key factor to improve iPH's usability. Some other interesting suggestions were given, such as the need to create a feedback mechanism to confirm that a contribution has been successfully received by master's device. On several occasions during the experiment the students sent their contributions more than once to the master.

In general, the experiment achieved positive results and was considered a success. All students think that the use of iPH augments the engagement during class, improves the interaction between students and the instructor, and most of them were quite satisfied with the iPH use experience. Table 3 shows the evaluation made by the students about some iPH's aspects.

iPH aspect	Avergage (1-Very bad / 5-Excelent)
Ease to connect to a session	3,8
Slide deck load time	3,8
Synchronization between users	3
Intuitive buttons and commands	3,4
Ease to use	3,4
Functions consistency	3,4
Increase interaction with instructor	4,6

Table 3. Classroom experiment results

5 Conclusion and Future Work

Our preliminary tests have given us with some confidence that running iPH on handhelds, such as pocket PCs or smart phones, for classroom usage is feasible. The performance results showed that the computing limitations of these devices won't have much negative impact on the systems usability, and hence will not disturb the course and pace of a class. It is worth to mention that in our first version of iPH the transfer of medium-size data sets to the handhelds was a problem, e.g. a deck of 1MB took more than 15 minutes to arrive at a handheld! This happened due to the correction algorithm utilized to ensure the correct transmission of messages, the Reed-Solomon polynomial algorithm, which was computed over the entire data set. So, we modified the deck transfer method to send slides one-by-one, instead of sending the whole deck, and which yielded the latencies presented in Figure 5.

Before using the iPH in regular classes, we need to more evaluations of system's usability and maybe redesign some aspects (functionality and interfaces) of iPH to improve it, a study which is currently underway. In addition, it is important that we analyze the usage of iPH on different devices, and for different teaching activities during classes. We expect to improve both iPH's user interface and collaborative functionalities based on these results. For example, in the current version of iPH, when a user looses his/her connection and the presentation is modified (e.g., the inclusion of a new public slide), the instructor has to explicitly re-send the modified deck of slides when this user re-connects. We intend to use context-awareness, i.e. information regarding connectivity status to enable such ”catch-up” automatically.

As future work we also intend to extend iPH's context-awareness and experiment with and explore other context-aware adaptations, e.g. related to other collaboration functionalities, user interfaces, access control policies, content adaptations, etc. Moreover, we also plan to conduct some experiments to investigate the potentials and problems of using context-aware and adaptive collaboration systems in classrooms.

6 Acknowledgments

We thank Microsoft Research for partially funding the development of iPH and CCXP.

References

[1] Richard Anderson, Ruth Anderson, K. M. Davis, Natalie Linnell, Craig Prince, and Valentin Razmov. Supporting active learning and example based instruction with classroom technology. In *SIGCSE '07: Proceedings of the 38th SIGCSE technical symposium on Computer science education*, pages 69–73, New York, NY, USA, 2007. ACM.

[2] Richard Anderson, Ruth Anderson, Beth Simon, Steven A. Wolfman, Tammy VanDeGrift, and Ken Yasuhara. Experiences with a tablet pc based lecture presentation system in computer science courses. In *SIGCSE '04: Proceedings of the 35th SIGCSE technical symposium on Computer science education*, pages 56–60, New York, NY, USA, 2004. ACM.

[3] Dave Berque, Terri Bonebright, and Michael Whitesell. Using pen-based computers across the computer science curriculum. In *SIGCSE '04: Proceedings of the 35th SIGCSE technical symposium on Computer science education*, pages 61–65, New York, NY, USA, 2004. ACM.

[4] Mohammed Waleed Kadous and Claude Sammut. Mica: Pervasive middleware for learning, sharing and talking. In *PERCOMW '04: Proceedings of the Second IEEE Annual Conference on Pervasive Computing and Communications Workshops*, page 176, Washington, DC, USA, 2004. IEEE Computer Society.

[5] Matthew Kam, Jingtao Wang, Alastair Iles, Eric Tse, Jane Chiu, Daniel Glaser, Orna Tarshish, and John Canny. Livenotes: a system for cooperative and augmented note-taking in lectures. In *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 531–540, New York, NY, USA, 2005. ACM.

[6] Microsoft. Microsoft .net compact framework. http://msdn.microsoft.com/en-us/netframework/aa497273.aspx, 2008.

[7] Brad A. Myers, Herb Stiel, and Robert Gargiulo. Collaboration using multiple pdas connected to a pc. In *CSCW '98: Proceedings of the 1998 ACM conference on Computer supported cooperative work*, pages 285–294, New York, NY, USA, 1998. ACM.

[8] Fernando N. Nascimento, Vagner Sacramento, Gustavo Baptista, Hana K. Rubinsztejn, and Markus Endler. Desenvolvimento e avaliação de um serviço de posicionamento baseado em ieee 802.11. In *SBRC 2006: Proceedings of the XXIV Brazilian Symposium on Computer Networks*, 2006. short paper.

[9] L. Nilson and B Weaver. *Enhancing Learning with Laptops in the Classroom*. Jossey-Bass, 2005.

[10] Microsoft Research. Microsoft research conferencexp project. http://www.conferencexp.net, 2008.

[11] Vagner Sacramento, Markus Endler, Hana K. Rubinsztejn, Luciana S. Lima, Kleder Goncalves, Fernando N. Nascimento, and Giulliano A. Bueno. Moca: A middleware for developing collaborative applications for mobile users. *IEEE Distributed Systems Online*, 5(10):2, 2004.

[12] D. Salber, D. P. Siewiorek, and A. Smailagic. Supporting mobile workgroups on a wireless campus. In *Proceedings of Mobile HCI 2001*, 2001.

[13] William M. Waite, Michele H. Jackson, and Amer Diwan. The conversational classroom. In *SIGCSE '03: Proceedings of the 34th SIGCSE technical symposium on Computer science education*, pages 127–131, New York, NY, USA, 2003. ACM.