Activity Based Client for iOS
Bachelor Thesis

Hildur Uffe Flemberg (270689-3043, hufl@itu.dk),
Niklas Schalck Johansson (121288-2727, nsjo@itu.dk)

Supervised by
Jakob E. Bardram (bardram@itu.dk)

IT University of Copenhagen

Wednesday May 23th

Abstract

Abstract goes here...

Acknowledgements

We would like to thank Professor Jakob E. Bardram, for making this project possible and for his help and guidance during the writing of this thesis. We would also like to thank ABC research team member and PhD student Steven Houben for his help during the entire process, providing valuable feedback, both for the thesis, as well as the development of the proof of concept system.

Finally we want to thank our test users Søren Engel, Nikolaj Aaes, Ida Haas, Julie Hansen, Nicklas Warberg, Pernille Kjeldsen, Christian Vestergaard and Christian Harrington for their engagement in the testing phase. We really appreciate your help, and all the feedback that you have given us.

Contents

List of Figures

List of Tables

1 Introduction

In this chapter, we will introduce the bachelor thesis and explain it's relevance. We will introduce the concept of Activity Based Computing and provide enough background information for the uninformed reader to understand it's purpose. At the same time, we will state an exact problem definition, explain how we approached it and briefly summarise the results. An extensive explanation of Activity Based Computing and the results of the thesis is deferred to subsequent chapters.

1.1 Context and Motivation

Today, work practices are often complex entities with respect to the number of resources and work flows involved. Take for instance a university student who follows several courses at a time. Each course might include literature to read, lectures to attend and exercises to solve. In order for a student to get through an average day, he would need to keep track of his schedule to know which course is taught that day and he would have to solve the exercises which requires resources like the document that defines them, the course web page etc. Furthermore, if he has several lectures the same day or the lecture and the exercises are held in different rooms then he would need to keep track of this and prepare a different set of documents for each activity.

As argued by Bardram [2011], moving away from the typical office desk environment has a negative impact on the challenges that arise from management of parallel activities. Users are left with their own organizing skills for tools when trying to prepare and share their resources before the activity is actually happening. In modern times, computers in various shapes have been utilized to meet and optimize those challenges but they all still provide support on the lowest level only, i.e. allowing the user to browse the file system, open and close applications, manually distribute their documents to collaborators by e-mail etc.

Activity Based Computing (ABC) is a new higher level paradigm originally developed for use within hospitals. It aims at aiding users in managing parallel activities at a higher level by introducing the notion of an "Activity" that acts as the basic computational unit and facilitates easy distribution, suspension and resumption.

In this thesis we would like to investigate how the Apple iPad can be used as an ABC enabled device to aid university students in managing their daily activities. Earlier research within ABC, Bardram [2009] has concluded that tablets were less useful as ABC devices in hospital settings. We find investigations of the usefulness of the iPad in university settings interesting for two reasons. First, university settings resemble hospital settings by being collaborative, non-office like and attached to multiple locations which imposes amplified activity management challenges as argued by Bardram. Second, the last couple of years, tablets have undergone an evolution which improves some of their properties that have earlier been subject to criticism in Bardram's research.

1.2 Background

The ABC project started in 2002 and took it's outset in pervasive computing designed for mobile, collaborative and time critical work for clinicians in hospitals. Since 2002 a lot of experiments have been carried out with different families of devices as well as technical implementations at different software levels. The current implementation is a java peer-to-peer system that is based on the AEXO infrastructure and targeted for non-PC devices.

Devices that have been used in ABC includes desktops, laptops, tablets and wall screen displays. Tablets have received a lot of criticism due to their size and weight which made them difficult to handle in front of patients. Bardram [2009] even explains how nurses were forced to drop the device in the patient's lap in order to interact properly with it.

Despite the fact that ABC was designed for use in hospitals it is introduced as a paradigm for ubiquitous computing that has fostered 6 general principles (which we will elaborate on in chapter 2) and these can be applied in any context. Lately, experiments with ABC has been carried out within the field of biology.

The ABC research team is financially funded by the Danish Council for Strategic Research. The team is located at the IT University of Copenhagen and led by Jakob Bardram.

1.3 Goals

The goal is to establish the ABC related usefulness of a modern tablet computer such as the Apple iPad in a university context. We will limit our investigation to include the following ABC principles

- Activity-Centered
- Activity Roaming
- Activity Adaption
- Activity Awareness

1.4 Approach

In order to achieve the above mentioned goals we will follow the approaches listed below.

- **Investigation**
  We will study the research that has already been done on ABC within other domains and take the results that have relevance to a university context into consideration when defining the features of the app. We will also familiarize ourselves with the latest version of the iOS platform and learn Objective C.
- **Implement a proof-of-concept solution**
  We will implement an iPad app to serve as a proof-of-concept solution. The app will serve as a client of the ABC framework developed by the ITU research team and implement the 4 ABC principles mentioned in the section above. The argument for this choice of principles is deferred to a later chapter.
- **Evaluation**
  We will make user tests to establish the usefulness of our product and argue for the feasibility of our goals. We will define a set of user scenarios and have genuine users from the target domain carry them out. Next, we will record their feedback, share some thoughts of our own and present the results in the report in a readable format as a collective evaluation.

1.5 Results

We developed an iPad app that supports the 4 principles we decided to focus on. We defined a set of features that we considered likely to contribute to the overall usefulness of the product and designed our user scenarios around those features. Next, we had users go through the scenarios and answer surveys with questions about the experience that could be rated on a scale from 1 to 5. The evaluation clearly showed that the majority of the users were happy with the product and felt that it would be of valuable use to them in their everyday work as students. The surveys also brought results to show that some features were clearly better than others. Equally important, most users didn't leave the room without suggestions to things that they thought should be improved or added to the solution. The feature that got the highest score of 4.9 (average) was the ability to open resources in other apps whereas the overview of activities were rated with 3.9 (average).

2 Activity Based Computing

In this chapter we will further explain the activity-based paradigm. We will explain why adaptation and awareness is important, and introduce the activity based computing framework that will help us create an activity based client.

2.1 Background

Activity based computing is a computer paradigm that moves focus from application based computer interaction into a higher level computational support for human activities. The paradigm has its outset in clinical work on hospitals, and seeks to aggregate resources to activities, instead of specific applications. An example of such an activity could be the development of our proof of concept. Opening an activity will cause the relevant services and resources to become available to the user, and allow to user to more easily switch between activities and all their associated services and resources. Figure 2.1 shows an example of the activity "Proof-of-concept development", its associated services and their resources.

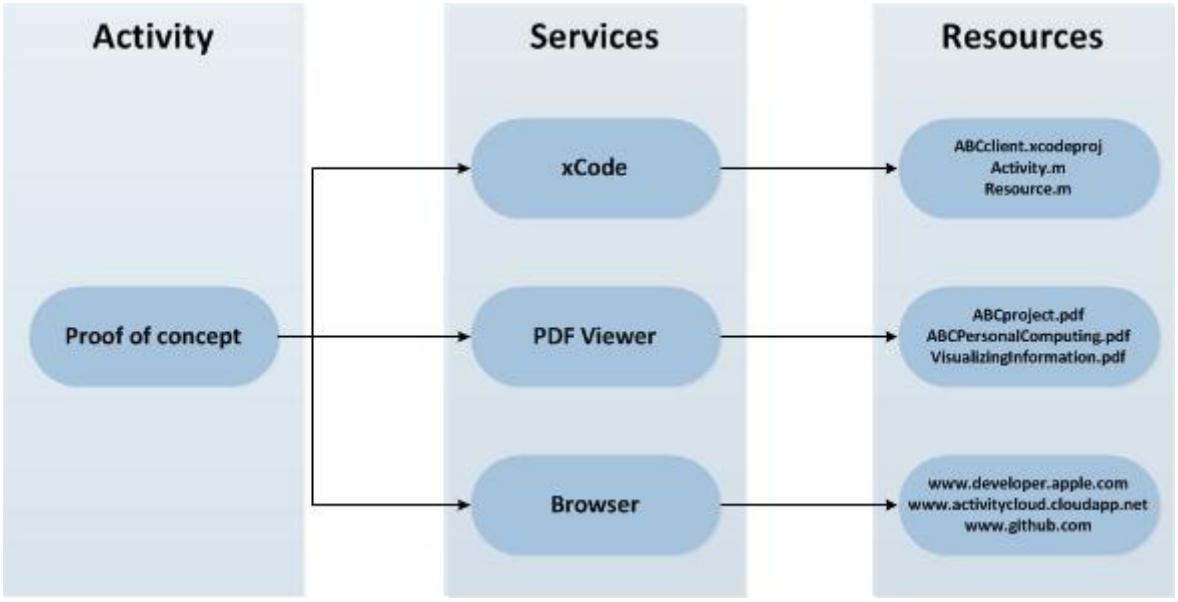

Figure 2.1: *The "Proof of Concept" activity. Illustrates how an activity encapsulates its services and resources*

Bardram [2011] identifies six principles that forms the basis of the activity based computing paradigm, being; *Activity Centered*, *Activity Suspend and Resume*, *Activity Roaming*, *Acitivty Adaptation*, *Activity Sharing* and *Activity Awareness*. Each of these will be further described in the following.

1. Activity Centered

An Activity is a computational unit that encapsulates a set of services and their relevant resources. An activity therefore encapsulate digital software and data necessary for a user to carry out their work (activity).

2. **Activity Suspend and Resume**
   This allows a user to alternate between several activities and support interruptions that requires the user to perform another task. This is done by suspending the current activity and resuming another.

3. **Activity Roaming**
   This principle enables activities to be stored on an infrastructure, like a server, and allows for activities to be suspended on one device, and then resumed on another, to better support user mobility.

4. **Activity Adaptation**
   An activity can be displayed and handled on very different devices, and should adapt to the resources available at the resumed devices. In this case resources could be CPU power, screen size and network bandwidth.

5. **Activity Sharing**
   Focuses and deals with the collaborative aspect of activities. This principle states that activities are shared among collaborators that appear on a list of participants for any particular activity. If two or more participants engage resumes the same activity they will both be notified and engage in video and audio chat if possible.

6. **Activity Awareness**
   Allows for an activity to be context aware, such that it adapts itself to its current environment and work context. This could be to i.e. adapting the user interface or changing activities and services based on where the device is located.

Implementing all of these features enables computational devices to better support human activities, and allow users to move away from the traditional document -and application centered model on a desktop computer. In this thesis we will focus on how to display activities on the iPad and how to adapt the interface to the orientation of the iPad, how to store activities in an infrastructure and how the iPad can be aware of its surroundings. In this chapter we will further explain the principle of adaptation and awareness.

2.2 Activity Adaptation and Awareness

In many work places it is normal for users to carry out their work in several different locations. As an example it would be natural to consider a plausible scenario for a professor during his day at a university:

> *A professor got a day full of meetings at the University. In the beginning of the day he will have a meeting with the head of a study programme at which he teaches a course. They will discuss several course related material, and course goals, and will require the use of the course website, a document with the goals of the study programme, the exercises used in the course. Later he will have a lunch meeting with a fellow colleague in the cantina, to discuss an idea for a project. During preparation he have found several online resources on the matter, and have made a few designs and diagrams he wants to share. After lunch he got a meeting in his office with a couple of students regarding a bachelor thesis, and needs to review some code written, which includes looking over several source code files, as well as a generated documentation file. Afterwards he got a meeting with a PhD student in his office, regarding his thesis that needs review, and also to discuss a certain article found online. During the meeting the search through an article database located online, and find a couple of interesting articles that they save for later use.*

This is a thought scenario but it clearly illustrates two things: first of all there is a need to aggregate resources to certain activities during the day for easy access, and second he only need certain activities at certain locations.

The first issue can be handled by using the activity based paradigm as explained earlier, in order to encapsulate resources with different activities. Now the second issue can be handled by filtering the available activities based on where the user is as illustrated in figure 2.2. Now in this case, only four activities have been mentioned but there might be many more than those. There might be activities planned for the rest of the week, and there might even be activities that are not work related. This could potentially sum up to quite a lot of activities and most of them are only relevant when you are in a specific location.

This is where the principle of activity awareness becomes important. Many wireless technologies exists today as explained by WirelessWikipedia [2012], which enables devices to communicate with nodes placed in a building. Using these technologies a device can communicate with these wireless nodes, and get information on its current whereabouts.

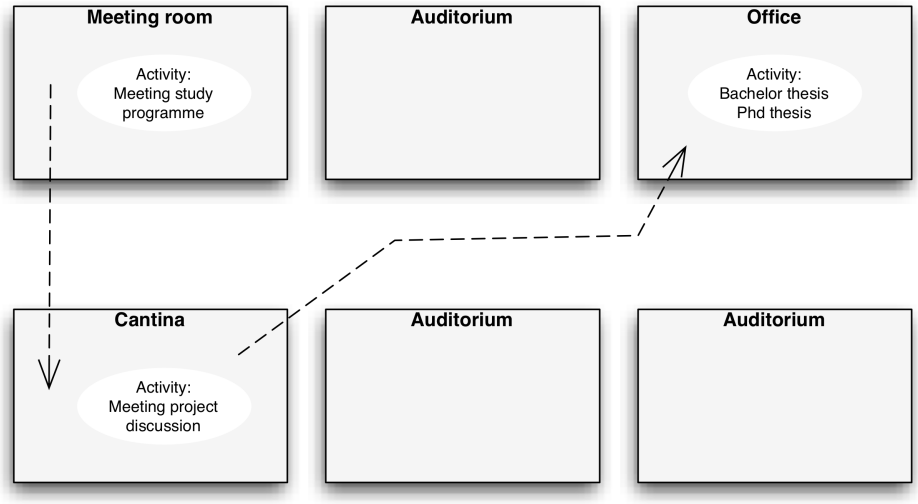

Figure 2.2: *Illustration on how the professor moves between locations in order to carry out his activities*

By using this information it is possible to keep an activity relation with specific locations in a building, and only show the activities that takes place in a given location. We can specify this kind of filtering as *location filtering*. Location filtering is thus a concrete way of handling activity awareness.

With regards to activity adaptation it is interesting to consider the result of the evaluation from Bardram [2009]:

> *This means that if an activity is roamed from a display with a large resolution (e.g., 1900ÃŮ1600) to one with a low resolution (e.g., 800ÃŮ600), a significant portion of the activityâĂŹs application windows will potentially be left outside the visible area of the display. A related issue arose when clinicians asked for activity roaming between very large devices (e.g., the wall display in the team room) to very small devices (e.g., a PDA or a SmartPhone). In principle, the ABC framework can run on a PDA or a Java-enabled SmartPhone using the J2ME edition. The real challenge, however, is to investigate further what it actually means to roam an activity between two such very different types of devicesâĂŤespecially if we take into consideration that the clinicians saw the small devices as tools for more specific actions*

within the overall activity. One possible approach may be to support roaming a subpart of an activity to a small device, instead of roaming the whole activity.

What is really interesting here is that it appears that the same kind of UI is implemented on very different devices, and that a possible solution could be to only handle a subpart of an activity. It should always be discouraged to handle very different devices similarly. It makes sense in standard desktop environments where most computers offer the same screen size and resolution, but when one moves from this environment, as is the case of activity based computing, one should also treat each family of devices differently. This means that PDA's and smartphones should have a distinct UI, tablets should have a distinct UI and so on. One could argue that this would mean a lot of overhead implement different UI's for different devices, but there exist design paradigms that takes this into account, and only require the UI part to changed, and not the rest of the implementation. It is also only natural that the UI is different as these devices would be used very differently as observed in Bardram [2009]:

[...] especially if we take into consideration that the clinicians saw the small devices as tools for more specific actions within the overall activity

So in order to fully make use of activity adaptation, it is important to recognize that each family of devices is different, and should be treated as such, and that their usage is also different. One would probably not replace a wall display with an iPad and hope to achieve the same thing. This is important to keep in mind when designing the UI of the client on the iPad, in order to support activity adaptation properly.

2.3 ABC Framework

In the following, a short historic overview of the ABC infrastructure will be provided, up to the current version, that is used in this project. The Activity Based Computing infrastructure is managemed by the Activity Research Team. Each subsection will give a brief overview, as well as the goal for each of the mentioned infrastructures.

2.3.1 ABC infrastructure version 4 - Windows XP

In 2006, Bardram et al. [2006], implemented an extension to Windows XP that would enable it to better support the ABC paradigm. The main focus of the paper released, remained on how to provide native Windows XP OS support, for an ABC infrastructure. The infrastructure was written in Java, and worked as a client-server solution to accomodate the Windows XP integration. The infrastructure was based upon easrlier infrastructures that supported activity discovery, activity roaming and activity sharing. The main difference from this solution and earlier versions was that the client did not have to be online at all times. All changes at the client side would be stored in a cache that would be sync'ed with the infrastructure when an online connection were available. The communication between the client and the infrastructure relied on the The Activity-Based Computing Protocol, that were modeled like the HTTP protocol by using stateless, string-based, unicast socket protocol with URI and URL string syntax.

2.3.2 ABC infrastructure version 5 - AEXO

In 2011, Bardram et al. [2012] implemented a new solution both on the client side and the server side. The first major change between this version and version 4, was the change from a client-server infrastructure, to a P2P infrastructure. The main focus of this infrastructure was to enable cross-device support. In version 4 the main focus was to support stationary desktop computers, using Windows XP, but now the version 5 supported several different devices and this was combined with using Adobe Flex that offered

a framework for developing applications for multiple devices and operating systems. The infrastructure itself was also changed to support the AEXO infrastructure which is a minimal, flexible, extensible datastructure that can be easily distributed across several devices. The AEXO framework were used to implement a distributed model-view-controller, and furthermore the AEXO infrastructure enabled an event-based mechanism for devices, such that it was possible to keep an synchronized copy of the distributed activity model across every device.

2.3.3 ABC infrastructure version 6 - RESTful

Both version 4 and 5 of the ABC infrastructure, were targeted a specific client solution that were to be used in correlation with the infrastructure in question. In version 6 this is changed. Using a RESTful interface, the infrastructure has been moved to the cloud to make the infrastructure completely device independant. The RESTful infrastructure still enables devices to store activities by creating, updating or deleting them. This version is currently work in progress by the Activity Research Team, and have not been completed yet, but for this project it was chosen due to the fact that communication is handled by wrapping JSON into HTTP GET, POST, PUT and DELTE requests, and this makes it completely device and OS independent. This were an important feature, since we were to develop a client for a completely new OS. Since version 6 the infrastructure that we will be using we will eloborate further on how the internal components works.

Figure 2.3 shows the structure of the infrastructure. At the bottom, three example clients is mentioned, in order to show the benefits of having a cloud infrastructure. Windows XP, ReticularSpaces as well as the client we are going to develop are be based on three completely different platforms, and with three completely different implementations, but by keeping the infrastructure in the cloud all three implementations would be able to easily communicate with it. The cloud infrastructure consist of three main components; Activity Access, Activity Manager and the File Manager.

Activity Access
The Activity Access handles the direct communication between the Activity Manager and the client the sends the requests. When a client makes e.g. a HTTP POST request, the Activity Access gets this request, and maps the request to a equivalent function on the Activity Manager component. Furthermore when the Activity Manager have completed the request, the Activity Access maps this result into a HTTP Response, that is sent back to the requesting client.

Activity Manager
When the Activity Manager gets a request and an input from the Activity Access, it executes the requested function on the input. The input itself is mapped to an Activity object defined in the Core library. The request is then send further to the File Manager. If the Activity Manager retrieved a GET request, meaning that the client wants to get a specific activity stored in the infrastructre, the Activity Manager asks the File Manager for the requested actvity, which is retrieved from the BLOB and handed over to the Activity Manager. If a POST or PUT is retrieved the Activity Manager asks the File Manager to update or create the Activity instead, and on DELETE requests, the File Manager is asked to remove the Activity from the infrastructure.

File Manager
The File Manager keeps track of all stored activities. It does so by using both the Data Table and Binary Large Object. The Data table is used for *Activity Wrappers*. An Activity Wrapper is essentially location information about a specific activity. It consist of the ID of an activity, as well as the specific PATH location of an activity within the service. Each time a new Activity is created the Activity Wrapper is created as well and stored in the DT. The activity itself is serialized and stored in a generated PATH. When a request for that activity is received, it File Manager looks up

the ID of the activity, and deserialize the activity from the PATH stored together with the ID, and hands the activity back.

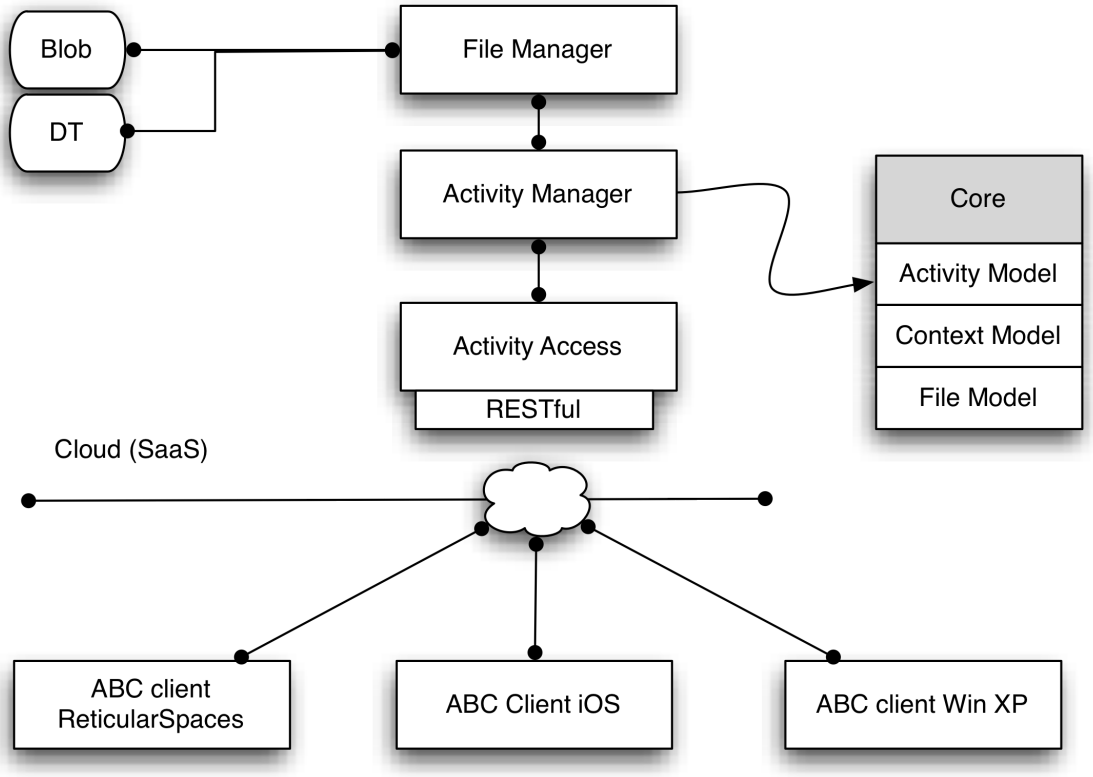

Figure 2.3: *Illustration of the ABC cloud infrastructure. At the buttom three examples on ABC clients connects to the cloud. Each client sends requests to the cloud service, which is then handled by the Activity Access and mapped to functions on the Activity Manager. Each request is then mapped to an activity object and an Activity Wrapper. The Activity Wrapper is stored on the Data Table, with info on where the serialized activity object is placed, such that it can be retrieved later.*

3 Mobile Activities

In this chapter three topics will be addressed: Transition from desktop to tablet computers, local and cloud computing as well as location tracking. Each section will provide an overview of the problem regarding these topics, and a discussion on how to solve it. Each section will end with a summary, on what solution or suggestions to use when implementing the proof of concept application.

3.1 Going from desktop to tablet computers

When going from a standard desktop computer with mouse, keyboard, screen and lots of computational power, to a tablet, which more a less is a screen that you can navigate on using just your fingers, the major physical differences between such two devices is evident. But when considering how to port the notion of activities from a desktop computer to a tablet computer the differences is just as many. Especially when visualizing information, the challenges become even greater as stated by Chittaro:

> *Unfortunately, limitations of mobile devices (e.g., limited screen size) make it impossible to follow a trivial porting approach from desktop computers to mobile devices. A considerable research effort is thus needed to understand how to design effective visualizations for mobile devices.*

It would therefore be interesting to consider how data is visualized in Bardram et al. [2006], which is a study on how ABC could work in a desktop environment and discuss how this could be represented in a mobile environment. One of the major challenges when going from the desktop environment to a mobile environment, no matter the reason, is the issue regarding parallel activities as described by Chittaro. So far we have considered activities to be a notion used in the ABC paradigme, but here one need to understand activities in a different context. When referring to parallel activities here, it refers to the fact that when a person is using a mobile device, they usually have to respond to other inputs as well. An example could be that one would be carrying around their mobile device, which means the holder of the device, not only have to interact with the device, but also needs to be able to walk at the same time. In another situation it could be a person that is performing physical work based on a drawing that is being shown on the device, or maybe a person is communicating with another person while interacting with the device. All of these situations leads on to the conclusion that when interacting with a mobile device, the user often does not have as much cognitive freedom to interact with the device, as they normally would in a standard office. Chittaro links this problem directly with the challenge of *presenting* and *selecting* data. Furthermore it is argued, that a user would still use powerful workstations for very complex tasks, given that they are probably way to hard to complete on a small mobile device, but often the users would like to take the result of these complex tasks with them to either show or use when carrying out another task. To further highlight this let us consider a scenario that takes place at an university:

Scenario 1

> *During the day the student participates in a programming lecture. The lecture lasts a couple of hours, and during the lecture the student is following the lecture throught the online slideshow on his mobile device. The classroom does not offer enough table space to allow for the student to bring a laptop, so he relies on his tablet computer. After the lecture the student need to complete some programming exercises, and finds a workstation with the necessary IDE and compiler installed, and begins solving the exercises. Once the student is complete he saves*

his work to his mobile device, leaves the workstation and head back to the class where the solutions will be discussed.

This example clearly show two things. First of all the mobile device mainly, only have to offer the ability to *present* data, not necessarly modify them. It also shows that complex tasks, such as a programming exercise, would ever be done on a mobile device due to its limitations. Since it would never be necessary to perform such complex tasks, files that are related to these tasks (such as IDE workspace, raw vector graphics for heavy foto editing and so on), are not necessary to present to the user. This means that only standard file types would be interesting, and the program in which they are opened, might not be as relevant as they are on the desktop. It should still be possible to open such file in ones favorite program on the mobile device, but a standard solution might just provide a standard way of showing such files, for quick access.

Another important aspect that Chittaro mentions is the importance of *human perception and cognitive capabilities*. He argues that for data to be properly visualized in a mobile environment, it is necessary for specific types of data to be easily recognizable, and more importantly, when one have implemented such a solutions it is very important that it be tested by proper users, in a real environment, in order to determine if the proper solution have been achieved. Such an evaluation will be a part of the full evaluation presented in chapter 5. The Windows XP integration with ABC allowed users to get a full overview over all opened files in a single activity, but also the ability to tab between activities. Due to the limited screen size, it is not feasible to show an opened version of all files, and neither is it feasible that one can tab between sets of opened files between activities. One should therefore consider a way to instead make it possible to easily identify a certain activity, and make the transition between activities fast, as well a giving a simple overview of the resources that an activity has. In other words there should be a way to make certain activities and resources distinctive. There should also be a simple overview of activities, such that the user would not be too confused.

Summary

In order to properly visualize activities several things need to be considered during implementation:

- Higher need of viewing items, than editing items.
- Only need to support simple tasks, not complex tasks.
- There should be an easy way to view most standard types of files.
- It should be possible to easily distinquish different activities.
- The overview of activities should be simple.
- It should be possible to quickly move between activities.

3.2 Local and cloud computing

Another important aspect in the transition from desktop computer to tablet computers is the visibility of the file system. In a traditional desktop operating system, the file system is generally very open and easily accessible by the generel user. One can simply navigate to a folder or a file in the systen, dobbelt click and then the file will be opened in the standard program for the specific file type. This is something that changes when moving to a tablet computer. In the case of the iPad, the file system is not visible to the general user, although it exist underneath every application to make it possible for opening and closing files. On the iOS operating system every application works like a little sandbox - they only have access to their own little box of files as can be seen in figure 3.1.

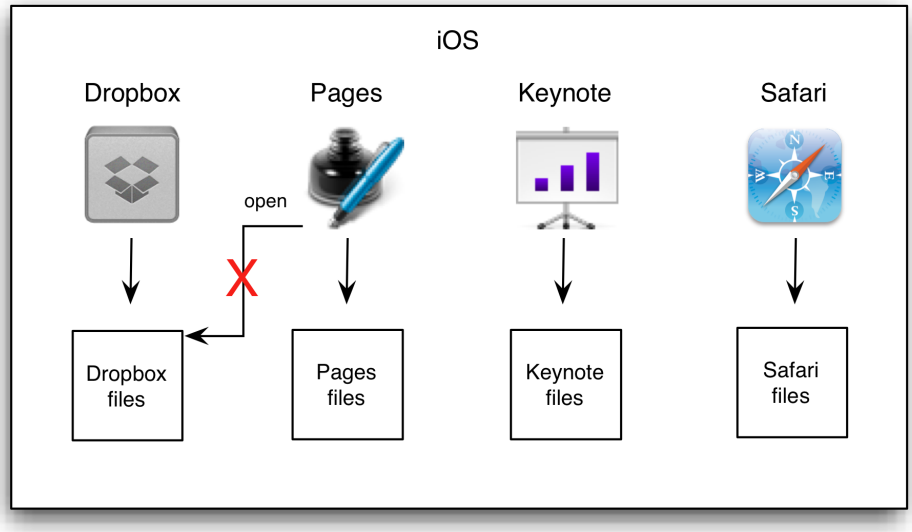

Figure 3.1: *Sandboxing in iOS. File are allowed to access their own protected pool of files, but cannot directly gain access to another programs files. In this example the Pages application for iPad would never be able to access the files from the Dropbox application.*

However, it is recognized that developers and users might want files in one application to be opened in another application. How this works is showed in figure 3.2. It is possible from a certain application to open one of its file in another application. In figure 3.1 the Pages application wanted to access another applications files and was denied. However it is possible for the Dropbox application to say that it want Pages to open its file.

The reason interaction between applications is like this is due to security reasons. If other applications cannot access your files, then they cannot change them, or apply destructive malware to your application. However this solution is not feasible. Dropbox is a widely used applications for synchronizing files between different personal devices. Dropbox thus serves as a mobile filesystem across several computers and devices, and even allows shared files and folders with other Dropbox users. Having to access files in the Dropbox application before being able to add them to an activity on the iPad is simply not a feasible solution, and the problem shown in figure 3.1 also shows that it is not possible to go the other way around. Only a few standard iOS applications allow other applications to access their files (like the foto application where images from the camera is stored), but this is very limited and cannot be extended to other applications. A solution to this problem is by using the cloudservice provided by Dropbox.

In cloudcomputing services are stored online, often in the form of data storage, or computing power, where it can be accessed from anywhere, usually through a webbrowser, independant of the OS accessing it. It only requires an internet connection as well as a compatible webbrowser as explained in the Cloud-Computing-Wiki [2012]. In this case Dropbox offers storage in the cloud, that can be accessed by local application using their public API. This way it is possible to access files in the cloud, instead of the local Dropbox application. It also makes it possible to download these files if necessary, and open them in external programs, directly from our own application. This solution is shown in figure 3.3.

This way it is possible to have a notion of a local file system, from which the user can add files as resources to activities, and also open these files in other applications within iOS.

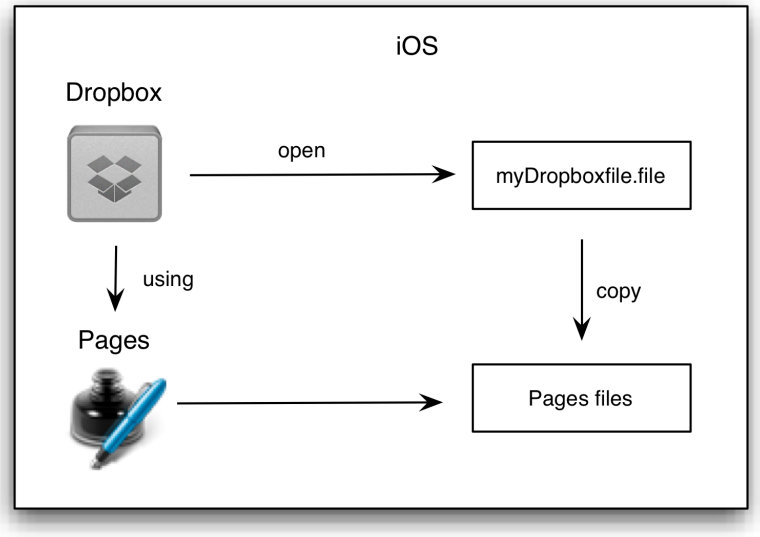

Figure 3.2: *How to open files in other applications in iOS. From the Dropbox applications point of view it is possible to open one of its private files in a another application, like Pages.*

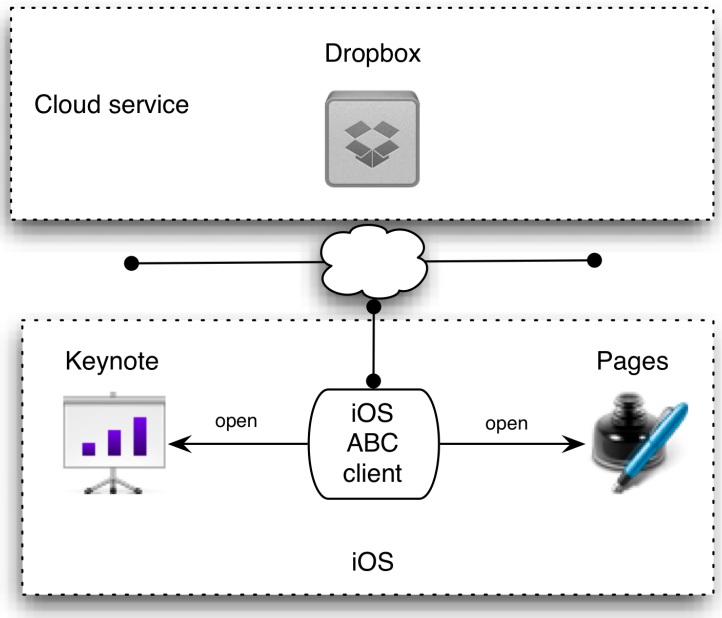

Figure 3.3: *Illustration on how the ABC client can communicate with the Dropbox cloud service, and present it as a local file system, from which the user can access and open files in other programs.*

Summary

To make it possible for users to access files, and use our application as the point from which all activities and resources is managed, it is necessary to connect to a cloud service, like Dropbox, which can act

as a fully accessible file system, from which users can add resources to their current activities. This is necessary because the full file system, is not directly accessible by the user in an iOS environment, and because, and it would not be possible otherwise, to prevent users from having to use other applications in order to make ours usable.

3.3 Location tracking

In section 2.2 it was argued how it was possible to implement a location tracking system within an ABC client. Location tracking will be further elaborated in this section.

The first important issue to solve when it comes to location tracking, is to define *how* to do location tracking. There are essentially two ways to do location tracking: coordinates or proximity. by using coordinates one would use a wireless technology to pin point your geographical coordinates. These coordinates can then be used to calculate your distance to other known places, using their geographical coordinates as well, and then determine whether you are within a suitable range, to be considered situated in that location. When considering such a solution two things are important to keep in mind; we are designing our solution for a university, in this case the IT university of Copenhagen, and we are going to implement a solution that is able to detect if we are in a specific room. Now using point processing one would be location independent, as one would always work on a uniform set of coordinates that can be used for distance calculations. However, finding out your exactly geographical coordinates can be problematic in a building, since one would either rely on coordinates retrieved from a satelite, which can vary a couple of meters from your actual position, or by using node triangulation which requires multiple wireless nodes to be able to locate the device. In this case we are not interested in knowing our current location *specifically*. What we are interested in is what we are close to as illustrated in figure 3.4.

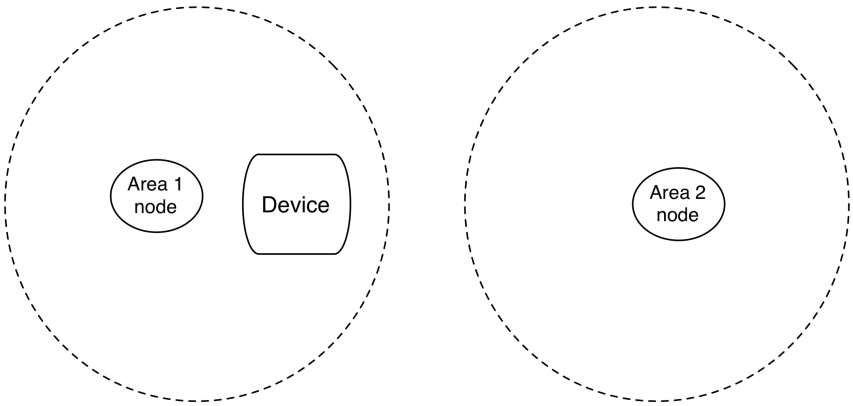

Figure 3.4: *In the case of proximity tracking, it is not important where your exact position is, just that you know you are in a certain area*

Using proximity it is possible to just query where the device is, and the node that detected you last, would respond that you are within its range. Bardram et al. [2012] made use of that in their RecticularSpaces study, by using the ITU Bluetooth location tracking system BLIP. The BLIP system offers a solution where it is possible for a device to query a webservice with its MAC address and is returned the current area in which it is situated, as well as the closest node that detected it. The BLIP system works on an event-based principle, such that each time a device is detected, the webservice is made aware of it, and the information is stored. When a device have not been detected for a while, its information is removed from the webservice. In order for a device to get detected, its bluetooth signature

needs to be visible to other devices. Since it is possible to get location information, the next step would be to associate activities with a certain location. When this table is made, each time the devices is detected by a BLIP node, the listed activities for a user can be updated.

Summary

To implement location tracking, the ITU bluetooth location system, BLIP, will be used. It offers the possibility to find out which area a certain device is currently in, and this information can be used to create a mapping between locations and activities to filter activities based on the information from the BLIP system.

4 Implementation

5 Evaluation

In this chapter we will test and evaluate our proof of concept implementation. It will be tested and evaluated with regards to the ABC principles that we support as defined in chapter 1. The tests are conducted in order to answer the question: *Is it possible to properly support the three ABC principles: Activity Centered, Activity Awareness and Activity Adaption on the iPad.*

We will evaluate the system based on constructed real-world scenarios. We want the system to be able to perform and be useful to people that are normally operating at a university, such as students. We will therefore carry out functionality tests, in order to determine the usefulness of the core features that have been implemented as discussed in chapter 3.

In the end we will discuss the results of the tests, and what improvement or changes should be considered for future work.

5.1 Evaluation of the proof of concept

Kuter and Yilmaz [2001] defines six steps as a guideline in order to properly carry out Human-Computer-Interaction tests:

- Set the goals - *What do you want to capture?*
- Decide on the target population and sample size - *Who will you ask?*
- Determine the questions - *What will you ask?*
- Pre-test the survey - *Test the questions*
- Conduct the survey - *Ask the questions*
- Analyze the data collected - *Produce the report*

We will use these guidelines as a basis for properly defining and carry out these tests. Kuter and Yilmaz [2001] further describes surveys as either quantitative or qualitative. Through quantitative surveys it is possible to get statistical data, but does is not very qualitative, that is it is impossible to know why a user likes or dislikes something in particular. Qualitative surveys are better for getting elaborated answers, but is very hard to use for statistical data.

Furthermore Kuter and Yilmaz [2001] defines face-to-face interviews as the best solution for gathering qualitative data. As a solution we will therefore conduct a quantitative survey for each scenario that the user is asked to participate in, and then follow up on each of these surveys with a face-to-face interview, in order to determine why they answered the what they did. This way it is possible to gather data for statistical analysis, and also to get specific feedback for future improvements.

5.1.1 Setting the goals

As the main objective is to find out how we can support the ABC principles Activity Centered, Activity Awareness and Activity Adaptation, it is important to find out how useful the functionalities that we implemented to support these principles are from a user point of view. If some of the features implemented turn out to not be very desireble, one could conclude that either it is not possible to support the affected principles, or that one need to rethink how to support it. It is therefore important that we define scenarios

that the user will go through, in order to simulate a real-world university situation, so that the test users will be able to better perceive the usefulness of the implemented functionalities. To summarize we define the goal of this test to dertermine the usefulness of the implemented features that support the Activity Centered, Activity Awareness and Activity Adaptation principles

5.1.2 Deciding on the target population and sample size

Since the proof of concept have been developed with a university environment in mind, it would be most suitable to bring in test users that are normally working at a university. We further narrow the test users down to be students. As testers we are very familiar with the student environment, and would be able to come up with a realistic real world scenario in which our application could be used. It would also be a lot easier for students to imagine the scenarios that we want them to complete, and also be easier for them to assert the value of the implemented functionalities. It would be interesting to bring in students with the same academic background as ourselves, but also students from other universities, in order to determine if our proof of concept would be feasible in more than one university. It is decided to use 7-8 test persons to carry out this test.

5.1.3 Determining the questions

It is possible to divide the questions into three categories: activities and resources, filtering and integration. Furthermore the questions will be formulated as statements, which the test user will scale, based on how much they agree with that statement, in order to get a quantitative measure. Each question will be measures on a scale from 1 to 5, where 1 means that the user strongly disagrees, and 5 means that the user strongly agrees. Each of the question categories will be further elaborated in the following.

Activities and Resources

The core concept of the proof of concept implementation is that we are able to support the notion of activities. An important functionality of the implementation is this that it should be possible to easility create these activities, but it is equally important how these are presented to the user in order for users to fully utilize activities. Another core concept of activities is the aggregation of resources. Resources, like activities, also have to be presented properly to the user, such that they are easilily accessible.

We therefore define these questions:

It is easy to create an activity. We want to find out if the proof of concept easily support the creation of activities.

The system gives a good overview of created activities. We want to find out if the proof of concept visualize activities in a logical and usable way.

I like the use of categories. We want to determine if the use of categories makes it easier to manage and handle activities.

I like the use of color coding. We want to determine if the use of color coding makes certain categories and certain activities easier to see.

The system gives a good overview of resources for a given activity. Like activities we want to know if resources are presented and visualized in a logical and usable way, but also to find out if these two concept should be handled differently, instead of similarly.

The ability to easily save a website that you are visiting is useful. We wish to know if users want to be able to quickly add a website they are visiting, instead of just writing the URL directly into a dialog box.

I had all features available in order to easily complete the tasks. This question might seem like a more usability minded question, but the intend is to force the user to think about if some core functionality is missing, in order for them to better handle the scenario that they are asked to complete.

Filtering

A very important part of the proof of concept is the use of location filtering, and it is therefore very important to get user feedback on how this works, and how useful they think it is. Furthermore we implemented the notion of categories, and also a filtering option based on this. An interesting result is also to see which of these filtering methods is perceived as the most useful.

We therefore define these questions:

I find the category filtering useful. We want to get feedback whether or not this kind of filtering is perceived as useful.

I find the location filtering useful. We want to get feedback whether or not this kind of filtering is perceived as useful. This is particularly important since the ABC paradigm defines Activity Awareness as a core concept, where activities are able to adapt based on its environment, and the result of this question could determine if this is a valid concept to use on the iPad.

Integration

Integration was done based on the paper that involved integrating ABC into a desktop environment. As discussed in chapter 3, it was important to integrate the proof of concept as much as possible into the existing operating system, but also by using cloud services, and provide an interface that worked as the basic interaction with the device as a whole. We therefore want to find out how desired and useful the implemented solutions are.

We therefore define these questions:

The ability to add resources from dropbox, image library and the camera is useful. Since we integrated three external systems that handles files, from which resources could be added, it is interesting to find out if this is a good solution for retrieving resources.

I find the all-time access to the browser useful. It was decided that the a browser should make up most of the UI space, and it would be interesting to find out if this a desired solution, or if it should be hidden until needed.

I find the integration with native apps useful. Last but not least, it is interesting to determine if integration with local apps is a desired and usable feature.

5.2 The Setup

As explained earlier, we wanted the users to participate by doing real world scenarios. The scenarios should be constructed such that they support the questions defined in section 5.1.

We came up with two scenarios: one that focuses on creating and managing activities and resources, as well as some of the integration solutions, and one that focuses on filtering and making use of local applications. Both scenarios will take place at the IT-University of Copenhagen.

The two scenarios are as follow:

Scenario 1

A student have a busy day tomorrow at the University. In the morning the student have to do a presentation in an Auditorium, which involves a PDF presentation, document notes and some sample websites. After the presentation the student have a meeting with his supervisor regarding a school project he is doing at the supervisors office. They will discuss several designs of a product the student is developing, and involves meeting notes, some online resources and several images of the design. Later that day the student need to attend a lecture on an interesting subject which requires him to have access to the lecture slides, the exercises presented, and a note document. This will take place in a small teaching room. At the end of the day the student will be attending workshop on innovation. This will include a website, and a sketch of a brilliant idea that the student would like som feedback on. The location for this activity have not been derermined yet.

Scenario 2

The day have come to carry out yesterdays preparations. The student will begin by going to 2C to do his presentation using the resources prepared yesterday. In the break during his presentation he wants to look through the rest of the presentation on in the local iBooks application. He finishes up his presentation and proceed to the meeting with his supervisor in 4C. He have a very fruitfull meeting, and are able to present all of his designs and ideas. They had a long discussion and a lot of drawings on the whiteboard, and the student desides to take a picture and add it to the activity. He also wants to add some comments to his notes, and opens up his note-PDF in the local GoodReader application. After a short break he moves on to attend his lecture in 4E. During the lecture he takes important notes, and feels refreshed by all the new things he have learned. He also finds a good tutorial online that he adds to his lecture activity. Last but not least he meets up with the other innovators, and they move around the university and find a suitable and available room. He finishes the workshop and head home.

Each survey will be organized as follow:

Survey - Scenario 1

- It is easy to create an activity.
- The system gives a good overview of created activities.
- The system gives a good overview of resources for a given activity.
- The ability to easily save a website that you are visiting is useful.
- The ability to add resources from dropbox, image library and the camera is useful.
- I find the all-time access to the browser useful.
- I had all features available in order to easily complete the tasks.

Survey - Scenario 2

- I like the use of categories.
- I like the use of color coding.
- I find the category filtering useful.
- I find the location filtering useful.

- I find the integration with native apps useful.

Initially it was necessary to create a known environment of activities in the application, from which the test user could familiarize themselves. It is also necessary to have more than the four activities that the test users are required to create during the first scenario. A student at a university would probably have at least two or three times that amount, to account for lectures, meetings, exercises and so on. It would also help to further emphasize if location and category filtering would be used, and the usefulness of color coding and category assignment for an activity. The example activities that are created prior to each test is shown in figure 5.1.

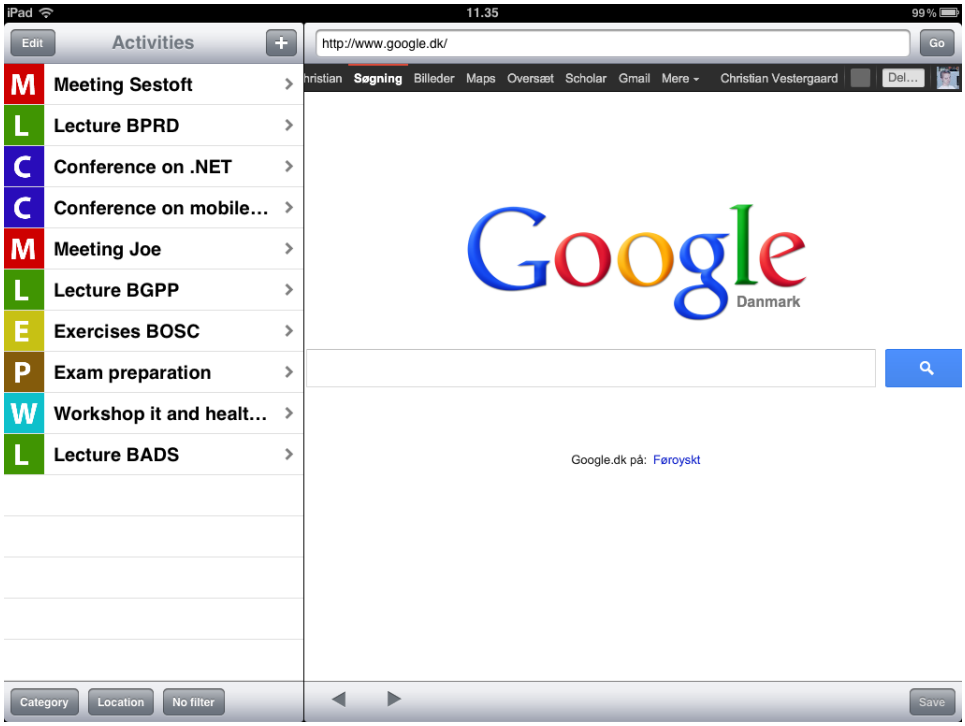

Figure 5.1: *The inital setup for a user. To the left are shown the example activities, with their category and color coding attached. To the right is the initial state of the browser.*

5.2.1 Walkthrough

One of the steps defined by Kuter and Yilmaz [2001] is that the testers should perform the test before performing the tests on the users themselves. Such a pre-test was conducted and in the following a sample walkthrough is provided.

Scenario 1

- Create an activity, by pressing the plus button in the top right of the activity overview, with the name of the activity (Lecture on thesis paper), the category P (brown) and the location 2C. Add the PDF-file presentation. pdf and document presentation_notes. doc from the dropbox folder by pressing the plus sign in the upper right corner, giving them a name, and a proper category (pdf, document), and the URLS: AQO.net and norseprojects.com, by accessing them in the webbrowser, and just press save in the lower right corner, and give them the category html page.

- Create an activity, by pressing the plus button in the top right of the activity overview, with the name of the activity (Project meeting), the category M (red) and the location 4C. Add the document meeting_notes.pdf from the dropbox and all the design images from the local image library folder by pressing the plus sign in the upper right corner, giving them a name, and a proper category (document, image), and the URLS: developer.apple.com and ui-patterns.com, by accessing them in the webbrowser, and just press save, and give them the category html page.
- Create an activity, by pressing the plus button in the top right of the activity overview, with the name of the activity (Lecture SIGB), the category L (green) and the location 4E. Add the document slides.pdf and exercises.pdf from the dropbox folder by pressing the plus sign in the upper right corner, giving them a name, and a proper category (pdf), and the URL to a google document by accessing docs.google.com in the webbrowser and adding it through the save button in the lower right corner.
- Create an activity, by pressing the plus button in the top right of the activity overview, with the name of the activity (Workshop Innovation), the category W (cyan). No location will be added. Add the image sketch.jpg from the dropbox folder by pressing the plus sign in the upper right corner, giving it a name, and a proper category (image), and the URL to itu-innovators.dk by accessing it in the webbrowser and adding it through the save button in the lower right corner.

Scenario 2

1. Have location filtering turned on and move to 2C. Of all the activities available only one should now be visible for easy access. He should download the presentation.pdf by holding down his finger on the resource, and then open it in iBooks from the popup window.
2. Have location filtering turned on and move to 4C. Of all the activities available only one should now be visible for easy access. Take a picture with the camera (by pressing the plus button, provide an image name and choose camera) of a whiteboard and add it to the activity.
3. Have location filtering turned on and move to 4E. Of all the activities available only one should now be visible for easy access. Go to the msdn.com and add it to the activity. He should download the meeting_notes.pdf by holding down his finger on the resource, and then open in GoodReader from the popup window.
4. Have location filtering turned off and move to 3D. Turn category filtering on for workshop. Only one should be visible for easy access.

5.3 Results

The tests were carried out using 8 users. All 8 users completed both scenarios, and all 8 users filled out both questionaires, and participated in a face-to-face interview after completing the surveys.

5.3.1 Quantitative Results - Survey

In generel most of the features are found useful by the users.

As can be seen from the results of the survey, the test users were very fond of especially three features; Integration with native applications, the ability to add resources from Dropbox, image library and camera, and the ability to easily save a website that you are using. The features that scored the least were the overview of created activities and if the user had all features available in order to easily complete the tasks. Based on these results it is especially interesting that the overview of activities scored lower than the overview of resources. One of the core functionalities, location tracking scored rather high, but not

Question	Avg score
It is easy to create an activity.	4.4
The system gives a good overview of created activities.	3.9
The system gives a good overview of resources for a given activity.	4.5
The ability to easily save a website that you are visiting is useful.	4.8
The ability to add resources from Dropbox, image library and the camera is useful.	4.9
I find the all-time access to the browser useful.	4.1
I had all features available in order to easily complete the tasks.	3.9
I like the use of categories.	4.0
I like the use of color coding.	4.1
I find the category filtering useful.	4.4
I find the location filtering useful.	4.4
I find the integration with native apps useful.	4.9

Table 5.1: The avg. result for each of the survey questions, based on the answer of 8 users. 1 is the lowest possible score and 5 is the highest possible score

as high as some other features. Following these results it is now interesting to consider the feedback from the users, on why they scored the different features the way they did.

5.3.2 Qualitative Results - Interview

The interviews were carried out as semi-structured interviews, as explained in Kuter and Yilmaz [2001]. The surveys were the basis for the interview, but the interview allowed the users to also talk freely about topics that were not necessearily mentioned in the survey. In the following some of the user feedback will be presented.

Activity Overview and Resource Overview

These two solutions turned out of be scored rather differently. It is therefore rather interestingly to find out why. If we start to look at what made the activity overview work well we got the following responses.

> *- I really like the use of color coding and categories in the overview - it makes the overview better in an unsorted list [...]*

> *- I like that all the activities are gathered in one place, and that you do not have to do too much to go through all of them.*

> *- Colors and categories are very nice, and gives and easy overview of what kind of activities you have to do.*

This shows that two things improves the overview of the list: Having all activities gathered in one place, and that colors and categories made it easier to get an overview. But when one looks a the critiques of the overview the color coding and categories are both mentioned:

- Activies are hard to get an overview of, because there are too many unfamiliar colors and categories.

- Overview is confusing. Would be nice to with some sort of sorting, or if it would be possible to define the colors and categories herself. Really likes the concept of color coding and categories though.

- Would be better if you could define your own colors and categories, that you find most familiar.

- The overview makes more sense for resources, since the list is probably shorter.

- It does not seem possible to get the location information for an activity. What if you forgot where to go?

The critique is mainly based on the fact that a: there is no possible sorting available. All categories are just shown randomly, and one can only improve this by using category filtering and location filtering, and b: it is hard to get an overview of colors and categories that you are not used to. This critique also reveals though that one could probably improve the overview a lot by a: providing options of sorting the list of activities without using filtering, b: make it possible for users to create their own activities and c: display the location for each activity.

For resources the following seemed to be the reason why it scored higher:

- I really like that the system suggest what I can do in the room that I am in right now. Then I don't need to think about it.

- Resources works better since there are fewer categories, and the images are well known.

- Resources are easier to identify, because the categories and its associated image is more well known.

This shows why resources worked better than activities. It is perceived that each activity, would probably not have as many resources in a list, as one would have activities. Furthermore the amount of categories, as well at the images used for these appeared easier to familiarize with.

Location Tracking

This concept was very important to get feedback on, and in the surveys the feature itself was scored 4.4 out of 5. The location filtering were mostly described as a nice feature, and it made it possible for the user to basically not think about filtering on their own - the application did it for them as described by three of the users:

- I really like that the system suggest what I can do in the room that I am in right now. Then I don't need to think about it.

- The location filtering really help to improve the lack of sorting. Suddenly you are only shown a couple of activities instead of a whole list.

- Location tracking in a university looks like it have great potential based on this solution.

These statements proves that; location filterings helps on the lack of sorting posibilities (which was addressed in 5.3.2), and that it enables to user to use less cognitive function in order to find a specific activity, and is perceived as a very feasible solution. Now when looking at the critique of location filtering, it is not so much that location does not seem feasible, but that it should be extended:

- It would be nice if it was possible to use location filtering like category filtering, such that you don't have to necessarily move to a location to utilize it.

Which means that not only should the solution be able to do this while moving around, but it should also be possible to utilize without actually being at the specified locations.

Integration

Integration with 3rd party services and applications was another major topic to be tested, and to find out if such a general feature were desirable. The feature regarding both integration with native apps, as well as the integration with Dropbox, image library and the camera scored the highest of all questions, and was very close to a perfect score for both. This also means that there were almost no critique but a lot of positive feedback:

- Integration wit dropbox is very nice. I rarely use any applications on the iPad that does not have Dropbox support, since i got all my work related files there.

- We often conduct biological experiments, which we document by taking pictures of the setup, so the integration with the camera is a very useful feature, since it make it possible to add it directly to your activity.

- In design projects one usually use a lot of images from your computer or camera, so being able to add images from the local image library is a very nice feature.

- I really like the integration with the local applications. I simply hate when I am not able to use my favorite programs for what I need to do.

These results emphazises how important it is to bring in known programs, that the users are used to. It also shows that the integrated were heavily used by students. Only one of the students did not use Dropbox on a daily basis, but used camera and images a lot instead. There were a single suggestion on how to improve the integration though:

- It would be cool if the integration could work both ways. Right now it is possible to open resources in 3rd party programs, but not add files from 3rd party programs to your activity.

Another integration was the access to the browser directly from our proof of concept. It did not score as good as the other integration solutions. It was generally stated that it was nice to have access to a browser directly, and that it did not require one to keep tabbing back and forth between out solution and the standard browser as well as most activities required one to access online resources, but the critiques were that it did not always have to be visible, but could be hidden until the use of the browser were needed, and then could be brought up. Others also argued that it did not really make any difference whether the browser were directly integration into the application. But at the same time those users also really liked the feature that one could save a visited site directly into an activity.

5.4 Suggestions for future improvements

The survey question *I had all features available in order to easily complete the tasks*, made it possible for the users during the interview to speak freely of what they thought would improve the use of the system. In the following we will therefore present some of the most notable, as well as most mentioned improvement for future work.

Calendar

Several of the users stated that they really missed an option to declare a date, and a time of the day of an activity. They mentioned that when one is at a university, it is usually only relevant to see what activities will happen today, and not tomorrow. Several of the users also expressed a concern with the current solution because they did not think that they would ever delete completed activities, and over time that would eventually involve having quite a lot of activities present in the activity overview. Having activities support time scheduling, would also make it possible to have the proof of concept work like an advanced calendar. One user even suggested that all appointments in the local calendar program on the iPad would automatically be retrieved an created as activities, that you could add resoruces to.

Metadata

In correlation with location filtering but also the above mentioned calendar feature, it should be possible to easily see this information for an activity. Many users complained that it was not possible to access this information, after an activity was completed. They were especially concerned with the fact that they might forget where an activity takes place, and then there would be no way through the proof of concept solution to find out.

Editing

It became clear very quickly that the lack of editing would have a negative impact on the user evaluation. We had not thought about this during evaluation, but we decided to complete the user test without chaning this. As was clear during the evaluation users were generally very quick to click their way throught he creation screens for both activity and resources, and often they wanted to change something they had already creation. Due to the lack of editing they were forced to delete what they had just created, and create the same thing anew witht the new changes. This caused a lot of frustration and should therefore be implemented. But another and even stronger argument was presented by one of the users:

> *- At my school there are a lot of problems with getting enough room for all the lectures, which result in lectures being moved around all the time, and its a big handicap if it is not possible to change the location on an activity that you have already created.*

The possibility to edit activities as well as resources should there be implemented in order to support such situations.

Sorting

The major point of critique of the activity overview, is that it lack sortings capabilities. Many users perceived location tracking a good solution to improve this, but that would not help if you were in a situation where you could not rely on this. Users specifically asked for the option to sort on name, time of day, date, collated categories (all meetings stand next to each other, all lectures stand next to each other and so on) or creation time (newest first). Users had the option to filter based on categories, but only found this useful if one had quite a lot of activities to filter on. In you just had a the current list full it would probably not be use as sorting would be more appropriate.

6 Conclusion

Bibliography

Jakob E. Bardram. Activity-based computing for medical work in hospitals. *ACM Transactions on Computer-Human Interaction*, Vol. 16(No. 2), 2009.

Jakob E. Bardram. The activity-based computing project. *AAAI Workshop (WS-11-04)*, Techniques and Languages, 2011.

Bardram et al. Support for activity-based computing in a personal computing operating system. *CHI 2006*, 2006.

Bardram et al. Reticularspaces: Activity-based computing support for physically distributed and collaborative smart spaces. *CHI 2012*, 2012.

Luca Chittaro. Visualizing information on mobile devices.

Cloud-Computing-Wiki. Cloud computing. `http://en.wikipedia.org/wiki/Cloud_computing`, 2012.

Ugur Kuter and Cemal Yilmaz. Choosing human-computer interaction (hci) appropriate research methods. `http://otal.umd.edu/hci-rm/survey.html`, 2001.

WirelessWikipedia. Wireless technology. `http://en.wikipedia.org/wiki/Wireless`, 2012.