NTNU – Trondheim
Norwegian University of
Science and Technology

Activity-Based Computing for the Cloud

Analysis of current technology and
suggestions for improvement

Oddvar Standal

Master of Science in Computer Science
Submission date: July 2013
Supervisor: Dag Svanæs, IDI

Norwegian University of Science and Technology
Department of Computer and Information Science

Preface

This report is the result of my master thesis, which has been carried out at the Department of Computer and Information Science (IDI) at the Norwegian University of Science and Technology (NTNU).

I would like to thank my supervisor, Dag Svanæs, for always having constructive ideas and for pointing me in the right direction throughout this thesis.

Oddvar Standal
Trondheim, July 2013

Abstract

Activity-Based Computing (ABC) is a way working with computers based on activites rather than just applications and documents.

This thesis covers the use of Activity-Based Computing concepts in current software products.

Several groups of software products have been analyzed in a product analysis, including cloud computing services and web browsers. This analysis reveals that most products have functionality similar to some of the ABC concepts. It also shows that some product groups includes more ABC concepts than others, and different products have similarities with different ABC concepts.

Suggestions for improvements to enable ABC in web browsers and cloud computing services are also included in this thesis.

A concept development approach have been used to propose possible solutions and improvements to web browsers. Web browsers are a good starting point for implementing ABC for use with cloud services and web applications.

The concept development is based on the analysis of the five currently largest web browsers, and provides suggestions for functionality required to fully implement Activity-Based Computing.

Samandrag

Activity-Based Computing (ABC) er ein måte å jobbe med datamaskiner basert på aktivitetar i staden for applikasjonar og dokument.

Denne masteroppgåva dekkjer bruken av Activity-Based Computing konsept i aktuelle programvareprodukt.

Fleire grupper av programvareprodukt har blitt analysert gjennom ei produktanalyse, inkludert cloud computing tenestar og nettlesarar. Denne analysa viser at dei fleste produkt har funksjonalitet som liknar på nokre av ABC konsepta. Den viser også at nokre produktgrupper inkluderar fleire ABC konsept enn andre, og forskjellege produkt har like trekk med forskjellege ABC konsept

Forslag på forbetringar for å gjere ABC til ein del av nettlesarar og cloud computing tenestar er også inkludert in denne masteroppgåva.

Ein konseptutviklingsmetode har blitt brukt til å foreslå moglege løysingar og forbetringar til nettlesarar. Nettlesarar er eit godt utganspunkt for å implementere ABC til bruk med cloud computing tenestar og nettapplikasjonar.

Denne konseptutviklinga er basert på analysa av dei fem største nettlesarane for augneblinket, og kjem med forslag for funksjonalitet som trengs for å implementere Activity-Based Computing fullstendig.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Activity-Based Computing

Activity-Based Computing (ABC) is a way of working with computers based on activities rather than applications and documents [20, 16]. The idea is to make human tasks as the entry point for users, i.e. instead of browsing files and applications, activities are the first class entities. An activity can include several applications and documents.

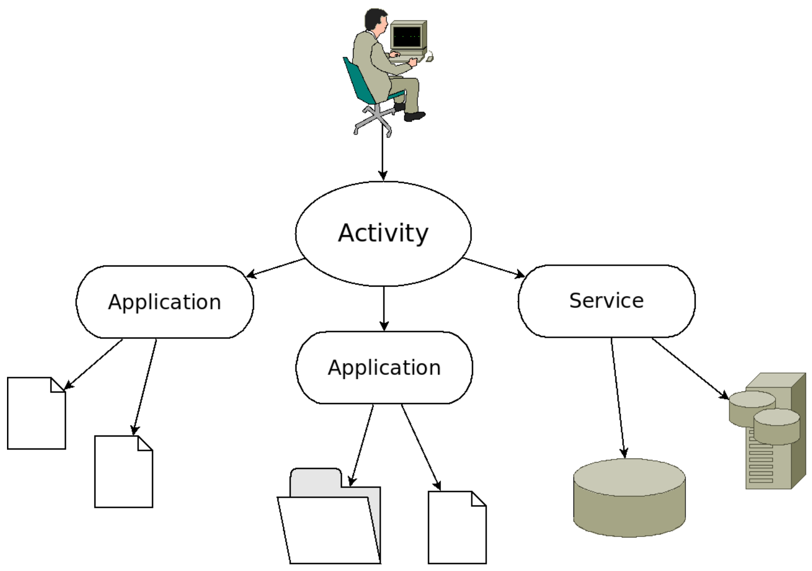


Figure 1.1: Activity-Based Computing

Applications and files are the most common way of working with computers, where applications are the tools for modifying data stored in files. By grouping both data and applications into activities (Figure 1.1), the user only has to select what task he will perform, and the computer automatically starts all needed applications and finds the relevant data associated with it.

An example of an activity could be to write a master thesis. The activity would consist of a text editor for writing, web browser for finding information, PDF-reader for reading documents and an email client for communication. All these applications are needed at the same time for this task to be effective. This is not necessarily hard to do on todays computing-model based on applications and documents. The problem occurs when the user gets interrupted and starts other applications to handle new tasks. The user is working on writing a master thersis and does not want to close the windows associated with it. After a while, the user has so many windows open and his desktop gets quite chaotic, so he decides to close some applications. When the user then wants to go back to writing the thesis, he will use time on reconfigure his desktop for this task. In the end, the user has spent a lot of time on just switching between tasks.

Activity-Based Computing has six fundamental concepts. These are explained thoroughly in chapter 2.

1.2 Cloud computing

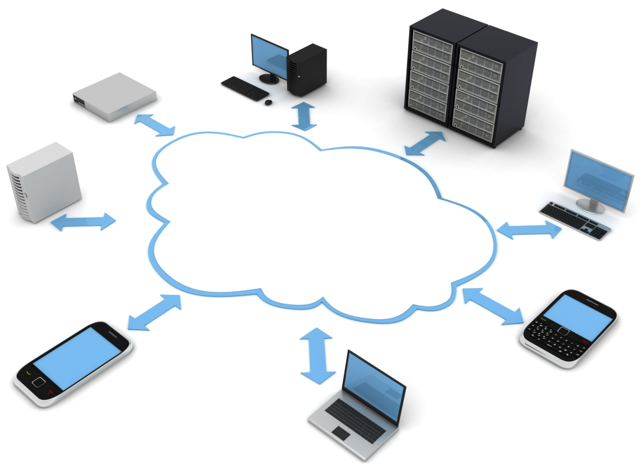


Figure 1.2: Cloud computing

Source: [44]

Cloud computing has become a very popular term in the last few years [9]. The term is also quite big and abstract [92]. If you ask different people what cloud computing is, you could get different answers. Some people thinks of the cloud as applications you can reach from anywhere, others may think that your files and documents are accessible everywhere, while corporations may utilize the cloud as computing resources on demand - and everyone has right. Cloud computing covers all these concepts, and more.

The technology behind cloud computing is not necessarily new [91, 92]. In the early computer history, centralized computing was the only way of working with computers. Back then the computers were large and the idea of distributed computing was very unlikely. But then the microprocessor came, and with it, the Personal Computer (PC). The PC has been under a rapid development, and has become much more powerful. It was able to do all the work and calculations itself. Along with cloud computing, centralized computing is on its way back - but this time in the form of large data centers with hundreds, or thousands servers. Even though PCs and other handheld devices (mobile phones, PDAs, tablets, etc.) are becoming more powerful, the need for computing power does still exist. Tasks that require heavy calculations or large data handling can be outsourced to the cloud.

Cloud computing is explained in chapter 4.

1.3 Web browsers

Figure 1.3: Web browsers

Source: [80]

Web browsers are, for many, the entry-point to the Internet and are becoming more and more important. With increasingly more applications in the cloud, people substitutes their local software with web-based software. People use web browsers for reading email, news, blogs, search for information, watch videos, listen to music, download files - and the list just goes on.

Web browsers have become much more than just a program that reads HTML-documents and lets users browse their way through web sites by clicking on hyperlinks. For a full web browsing experience, the browser needs to handle multimedia - from video/audio to games/applications. Most browsers supports this through plugins.

With plugin-support and the large number of web applications available in the cloud, web browsers are soon the only piece of software you need to have installed locally. Google has the same idea, and have come up with Chrome OS [36]. Chrome OS is basically a web browser running on a small Linux-based operating system. Applications in Chrome OS are based on web applications, which are opened in a new browser window. This window is without the typical web browser menu and addressbar, which gives the web applications a more standard application-like look and feel.

More on web browsers is found in chapter 6.

1.4 Research questions

The combination of cloud computing services and web technology leads to people spending more and more time in web browsers. People can, to a greater extent, do all their tasks in the web browser instead of being dependant on other local applications.

Nowadays, people usually own and use more than one computer or mobile device. With web browsers available on almost any computer or device, a user can access his cloud computing services regardless of what computer or device he has in hand. This also means that a user is not limited to his own computer.

People tends to do several tasks simultaneously. Most web browsers can support this, either through multiple windows or multiple tabs. Switching between tasks and their respective windows and tabs can be a tedious task, especially when you have many, many open windows and tabs.

The Activity-Based Computing concepts relate to these challenges. My goal is to find out which ABC concepts exists in software products and web browsers today, and how these products can be improved to fully support the ABC paradigm. This leads to the following two research questions:

- **Research question 1:** To what extent do we find functionality similar to the concepts of Activity-Based Computing in current cloud computing services and web browsers?
- **Research question 2:** How can current cloud computing services and web browsers be improved to support Activity-Based Computing?

1.5 Research methods

To answer the research questions in this thesis, two research methods have been used, product analysis and concept development [77, 78].

According to Shaw in [77], there are five different approaches to software researching, ordered by how thoroughly the results are (5 = most thorough):

1. **Qualitative or descriptive model**

   > "Organize & report interesting observations about the world. Create & defend generalizations from real examples. Structure a problem area; formulate the right questions. Do a careful analysis of a system or its development."

2. **Technique**

   > "Invent new ways to do some tasks, including procedures and implementation techniques. Develop a technique to choose among alternatives."

3. **System**

   > "Embody result in a system, using the system development as both source of insight and carrier of results."

4. **Empirical predictive model**

   > "Develop predictive models from observed data."

5. **Analytic model**

   > "Develop structural (quantitative or symbolic) models that permit formal analysis."

1.5.1 Product analysis

The first part of this thesis is to find existing Activity-Based Computing concepts in software products and the latest, state of the art web browsers. To do this, I have performed a product analysis, in which I have selected a number of software products and analyzed each one of them. The products are split into smaller parts and each relevant functionality is compared according to the six ABC concepts (Explained in chapter 2). A product analysis fits the Shaw's qualitative or descriptive model (1).

The outcome of each ABC concept is graded in three steps:

- **Implemented** If the product has functionality that is the same as an ABC concept, it gets marked by a ✓.
- **Partially implemented** If the product has some functionality that resembles an ABC concept, it gets marked by a (✓).
- **Not implemented** If the product does not include any functionality that resembles an ABC concept, it does not get any mark.

1.5.2 Concept development

The second part of this thesis is to come up with possible improvements to both web browsers and cloud computing services. To answer this, concept development has been used. Concept development is based on ideas and proposals, including architectural designs. It does, however, not include any implementation or testing. Concept development comes in under Shaw's technique approach (2).

1.6 Thesis outline

This thesis consists of nine chapters and is composed as follows:

Chapter 1 - Introduction

This chapter introduces the context of this thesis and the research questions.

Chapter 2 - ABC concepts

This chapter explains the six basic concepts of Activity-Based Computing.

Chapter 3 - ABC framework

This chapter gives an overview of the ABC framework and applications related to ABC.

Chapter 4 - Cloud computing

This chapter explains cloud computing, including the different services, cloud types and challenges.

Chapter 5 - ABC concepts in software products

This chapter covers the analysis of Activity-Based Computing concepts in different software products.

Chapter 6 - ABC concepts in web browsers

This chapter will show the various ABC concepts found in the largest web browsers - Firefox, Opera, Chrome, Safari and Internet Explorer.

Chapter 7 - ABC and the cloud

This chapter outlines possible improvements for integrating Activity-Based Computing concepts in web browsers.

Chapter 8 - Discussion

This chapter will discuss the possible solutions and answer the previously stated research questions.

Chapter 9 - Conclusion

This chapter contains the conclusion, based on how the research questions have been answered. Suggestions for further work is also introduced.

Chapter 2

ABC concepts

This chapter presents the Activity-Based Computing concepts and their background.

Jakob E. Bardram is a central person when talking about Activity-Based Computing. He is a professor working at the Pervasive Interaction Technology Laboratory (PIT—LAB) at the IT University of Copenhagen [49], and has done a lot of work on Activity-Based Computing [20, 15, 19, 16, 22, 18, 23, 17, 21]. Alongside of his research in pervasive- and Activity-Based Computing he has co-founded Cetrea A/S [25] as a spin-off company. Cetrea is specializing in pervasive computing technology for hospitals.

Activity-Based Computing is based on six different concepts. These concepts were established after studies of computer usage in healthcare environments [15, 22]. Healhcare workers use computers in different ways than traditional computer workers. They are mobile and often has to switch between tasks. This leads to a challenge for the standard computational model. With just applications and files, it makes reconfiguring for a new task unnecessarily complicated and time consuming. The six Activity-Based Computing concepts addresses these challenges.

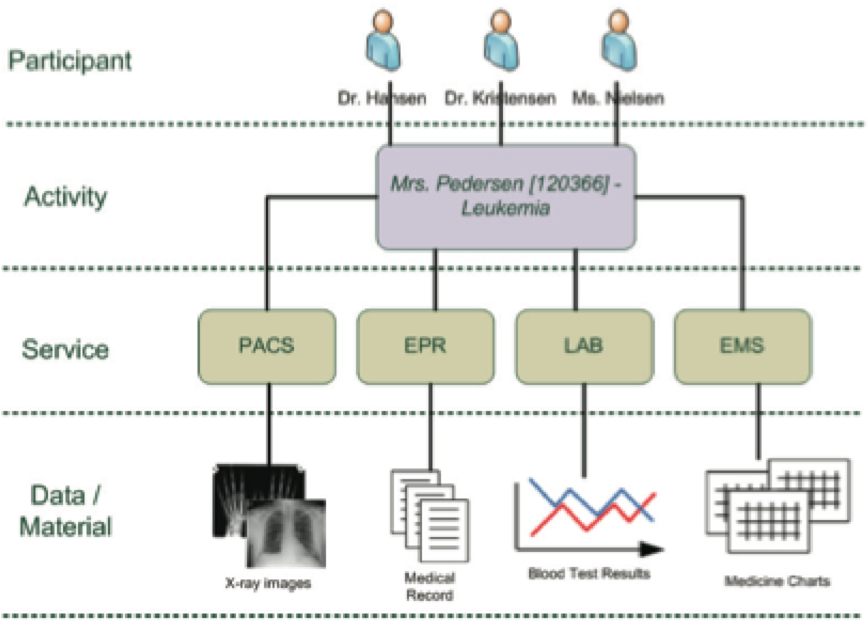


Figure 2.1: ABC conceptual model

Source: [84]

2.1 Activity-Centered

ABC is activity-centered. It introduces a new layer of abstraction on top of applications and files (See figure 2.2). Instead of relating to all these applications and files as individual tasks, the participants only needs to relate to the activity. The computer manages the rest.

According to Bardram [15], there are three levels of abstraction:

1. **Activity**

   This is the top level layer which represents a human process. An activity is a collection of both applications/services and files/documents accociated with the specific task. In figure 2.1, we see that the activity is Mrs. Pedersen - Leukemia.

2. **Applications and services**

   The middle layer consists of applications and services. Applications are used to display, edit and manipulate data - like word processing or image manipulation. Services can be web-based, for instance using Google

Search to look up information. In figure 2.1, the applications/services included in the given activity is PACS, EPR, LAB and EMS.

3. **Files and documents**

   This is the bottom level and consists of files and documents. Files are just data stored on a data storage device - hard drive, network storage, etc. A file is stored using a format that makes it readable/editable in an application. In figure 2.1, each service has their corresponding files relevant for the given activity, like X-ray images, Medical Records, etc.

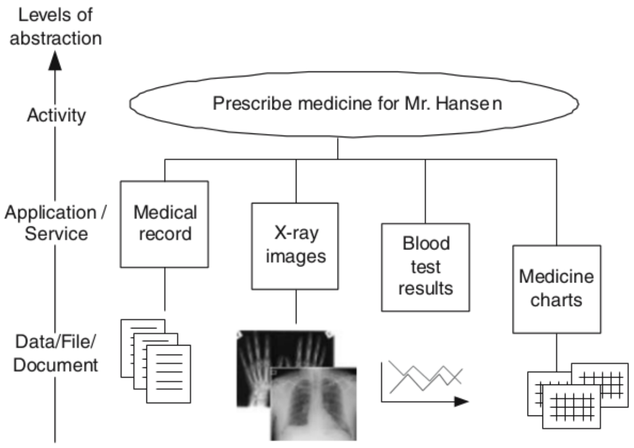


Figure 2.2: ABC abstraction layers

Source: [15]

2.2 Activity Suspend and Resume

The principle of suspending and resuming is dependant on storing the activity state information. This includes programs in use, where they are positioned on the desktop, what documents and files are opened in their respective programs, the position in the document (e.g. page number), etc. The idea is to give the same look and feel as last time the activity was used, so the user can continue working exactly were he left off.

The state of the activity can be stored in either a local file or on a server, which gives it the ability to be suspended and resumed. This is useful when a person has to work with many different tasks in parallel. A normal usage would either have many windows/programs open or require to use valuable time and effort on closing and restoring everything related to a specific task.

A desktop full of open windows can easily get messy and hard to keep track of what belongs to where. This could reduce the overall efficiency of completing tasks. With the Activity-Centered approach and the ability to suspend an activity, and then resume it later, this becomes a much more trivial solution.

A good example of why an activity needs to be able to suspend and resume can be found in hospitals [18]. Clinicians are often interrupted during a day, which can be from the fact that medical work is highly collaborative. This causes a stop in their current activity, and they have to switch context fast. An easy way to do this is to suspend the current activity and open the new one. When they are done with the new activity, they can suspend it and resume the original activity. This gets the user back to where he was before the interruption, with the same programs, services, documents he was using - exactly how he left it.

2.3 Activity Roaming

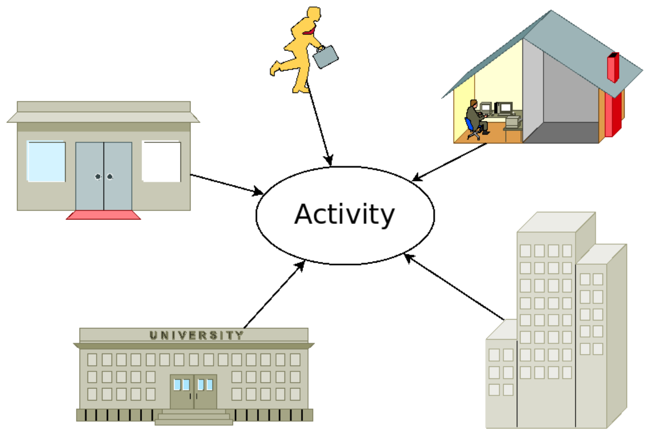


Figure 2.3: Activity Roaming

In many cases, a person has to be mobile. Instead of having to carry a computer around, the user should be able to use an arbitrary computer where he is. Activity Roaming makes it possible to work with the same activity regardless of location. This concept relies on the previous concept (See 2.2), the ability to suspend and resume an activity. The idea is to be able to resume an activity on any compatible device, anywhere. This can be done by storing the activity on a server which is accessible over a network. A person can start an activity on one computer, suspend it, and then resume it on another computer and keep working.

Roaming is useful for bringing your activities to where you are, whether you are at home, at the office, at the university or even when you are on the run. A practical example is from hospitals, where healthcare workers are changing location many times during a day. Research [18] revealed that it was too cumbersome to carry around a tablet PC. It was too big and heavy. Clinicians preferred to use stationary displays standing around in the hospital. Activity Roaming was proven useful to healthcare workers, because of the quick and easy access to the task-related information.

2.4 Activity Adaptation

In combination with Activity Roaming (See 2.3), Activity Adaptation is an important concept. It states that an activity has to adapt to different devices, including desktop computers, laptop computers, mobile phones, PDAs, tablets, wall displays, etc. These devices has different resources avaiable.

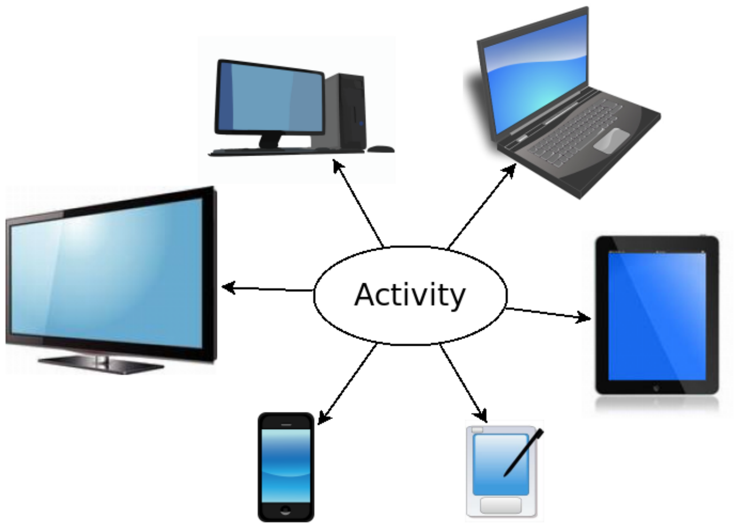


Figure 2.4: Activity Adaptation

The first, and perhaps most obvious challenge, is the screen size. If an activity is suppose to work both on a small screen (PDAs and mobile phones) and larger screens (wall displays), it has to support a mechanism for resizing its content. This can be done by zooming the entire activity, so it fits the screen resolution and size [16].

Secondly, the input devices plays a large role. If the device feature a touch-screen and has no keyboard, the activity must adapt to allow operation without a mouse pointer. This could be bigger buttons, because fingers are not as accurate as a mouse. It must also allow an on-screen keyboard to be present, without loosing functionality because of the smaller screen. Perhaps the device is a large wall display with multi-touch for several people working at it simultaneously. Or it could be just a normal desktop computer with a keyboard and mouse.

Other limitations could be processor, memory and network. Small mobile devices operates on a battery, and with limited battery capacity available, low-end processors are used to save energy. This could have a negative effect on activities which involves heavy software.

If an activity includes many applications, a problem could occur if the device or computer resuming the activity has low memory. When a computer is running low on memory, the harddrive steps in as a reserve. Harddrives are tremendously slower than memory, and thus makes the computer slow and unresponsive.

When roaming, the network connection has great importance for the experience. Networks can be slow in two ways, either low on bandwidth, or high on response time. When transferring large amounts of data, the bandwidth becomes the bottleneck. When using a response time critical service, like remote desktop (See 5.2) where you need feedback on your actions, it is important to have a low respones time. Otherwise, the service feels very slow and unresponsive.

Activity Adaptation is a principle that addresses these challenges and enabling an activity to be used across multiple heterogeneous devices, regardless of their available resources.

2.5 Activity Sharing

Collaboration and teamwork are important aspects of many jobs. Activities can be shared among a set of participants. There are two ways for an activity to be shared, synchronously and asynchronously [18].

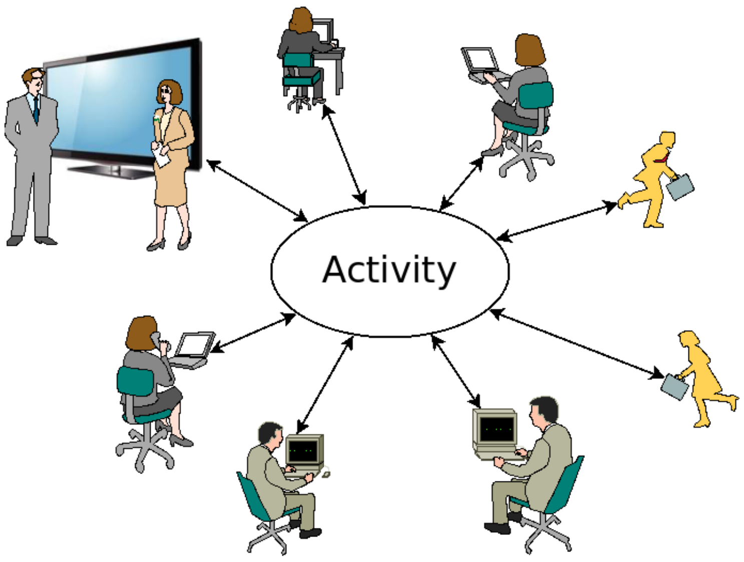


Figure 2.5: Activity Sharing

Asynchronous Activity Sharing

Asynchronously sharing is when two or more people are working with the same activity, but not on the same time. It builds on the same principle as the Activity Suspend and Resume concept (See 2.2). Multiple participants have access to the same activity, including its state. After a user is done with it, he suspends the activity. When another participant needs to continue working on the same activity, he resumes it.

In combination with Activity Roaming (See 2.3), the participants does not need to be on the same device. The state of the activity is stored on a network accessible storage device, for instance a server. An example is when people are working shifts, and they use their own computers. When a person is done for the day, he suspends the activity, packs his computer and goes home. The next person then resumes the activity on his own computer, exactly where the first person left off.

Synchronous Activity Sharing

Synchronous sharing is when participants are using the same activity at the same time. Every participant have the same view on everything within the activity. This is synchronized across the network, so every device will have a live display of what everyone is doing.

2.6 Activity Awareness

An activity is aware of its context. Based on location, surroundings and physical work context an activity knows what the user is doing and can adjust itself accordingly [16, 28, 18]. Activity Awareness states that the computer system should keep track of a user's real-world activities, and thus manage to suggest appropriate activities for the user.

Research shows that the Activity Awareness concept is useful [18]. Instead of searching through large amounts of data to find the relevant records, a physical RFID-tag was presented to the computer, and it instantly found the correct data.

Activity Discovery

Activity Discovery was a principle for helping users to identify, create, manage activities based on context, recurring activities, work habits and templates [22]. This has later been combined with Activity Awareness [21].

Chapter 3

ABC framework and applications

This chapter presents briefly the ABC framework and related applications with focus on how the ABC concepts affects the user interface. The ABC framework exists in several versions. This chapter will go through version 3 (3.1), version 4 (3.2). Another implementation of the ABC concepts, the co-Activity-Manager (3.3) will also be introduced.

The ABC framework is an implementation of the Activity-Based Computing concepts, and provides a runtime environment and programming platform for Activity-Based Computing applications. It has been used as a platform for researching the use of Activity-Based Computing for health workers [18, 50, 71].

A version 5 of the ABC framework also exists [21].

3.1 Version 3

Version 3 is an early implementation of the ABC framework [20, 15, 19]. This version is based on Java and requires ABC-aware applications. An advantage when using Java, is its platform independence - it works on Windows, Mac OS X and Linux.

Figure 3.1 shows the user interface of the ABC framework. It resembles the look and feel of the user interface in Windows and consists of the following parts (numbered accordingly):

1. **The Activity Bar**

   The Activity Bar provides the user with a set of tools for handling the different parts of the ABC framework. This is the most central part of the user interface and is similar to the taskbar in Windows.

2. **The Collaboration Frame**

   The Collaboration Frame displays information on who is logged in at the different computers, their location, status, which activity is currently active, etc.

3. **ABC applications**

   This is ABC-aware applications running in the current activity. They are displayed as regular programs. When changing activity, current windows are hidden, and the new activity's windows are shown.

4. **Tele-pointers**

   This is the mouse cursors of all the participants when collaborating on an activity. Each tele-pointer displays user information related to it. This way, each participant can see which cursor belongs to who.

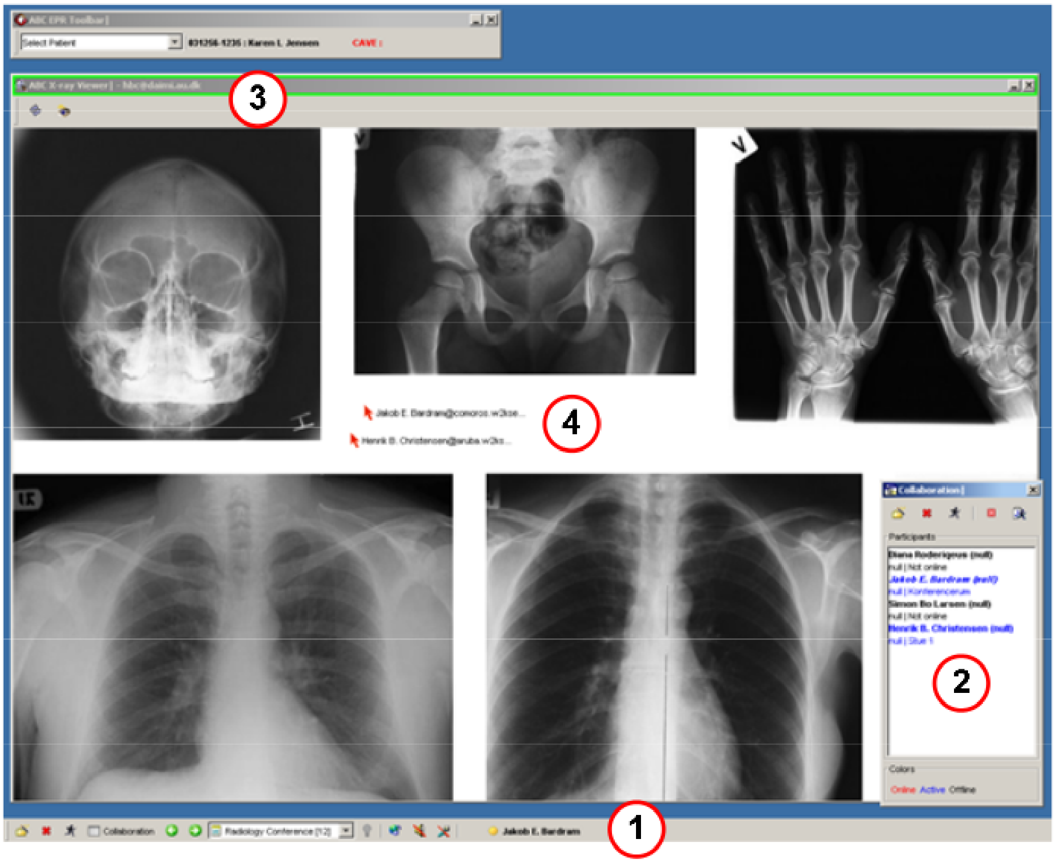


Figure 3.1: ABC framework 3

Source: [19]

3.2 Version 4

Version 3 of the ABC framework was pointed towards hospital-related work, but with version 4, the use of Activity-Based Computing was intended to normal work situations in Windows. Version 4 is implemented in .NET [16, 24, 22]. The point was to make normal Windows application easy to include in the ABC framework. This way, it could be used with more than just the hospital-related software available for the version 3.

Figure 3.2 shows how the Activity Bar in version 4. It has the same look and feel as the Windows XP taskbar, and it shares the same functionality, except that instead of listing programs, it shows activities consisting of several programs. The original Windows taskbar was replaced by the Activity Bar. When pressing the *Activities*-button, a dropdown menu shows up (See 3.3). This contains all activities, both suspended and current. Each activity has a thumbnail of the desktop view associated with it. To choose if a program should be a part of the current activity, a button on the application frame was implemented (See 3.4). With a simple click, the user could toggle whether or not the activity should be a part of the activity.

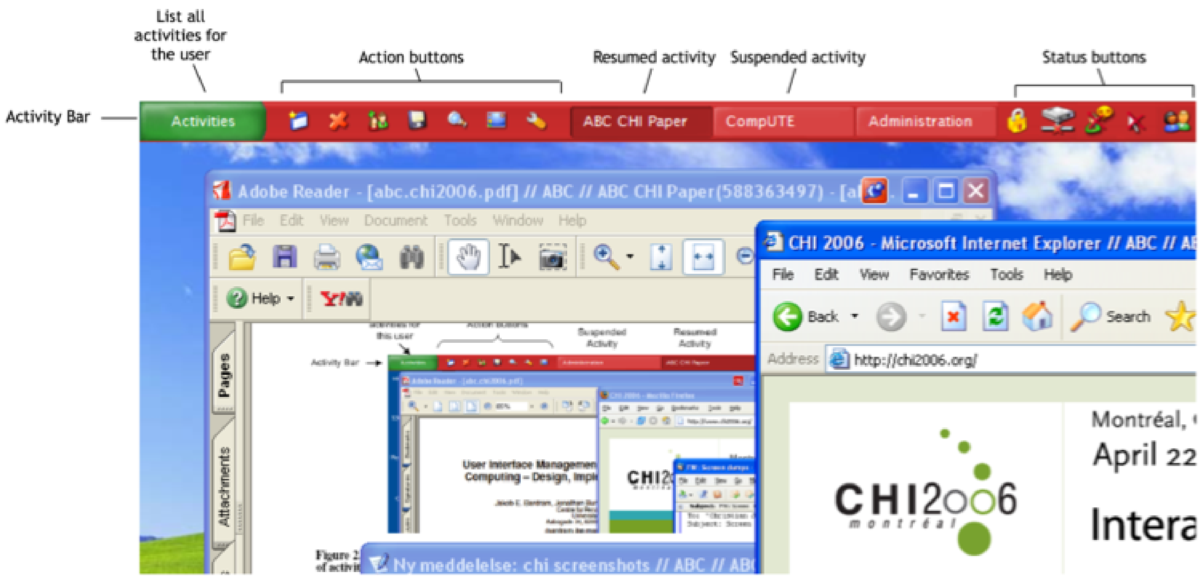

List all activities for the user
Action buttons
Resumed activity
Suspended activity
Status buttons
Activity Bar
Activities
ABC CHI Paper
CompUTE
Administration
Adobe Reader - [abc.chi2006.pdf] // ABC // ABC CHI Paper(588363497) - [a
File Edit View Document Tools Window Help
Help
CHI 2006 - Microsoft Internet Explorer // ABC // AB
File Edit View Favorites Tools Help
Back
Search
Address http://chi2006.org/
Pages
Attachments
CHI2006 montréal
Montréal,
April 22
Intera
Ny meddelelse: chi screenshots // ABC // AB

Figure 3.2: ABC framework 4

Source: [83]

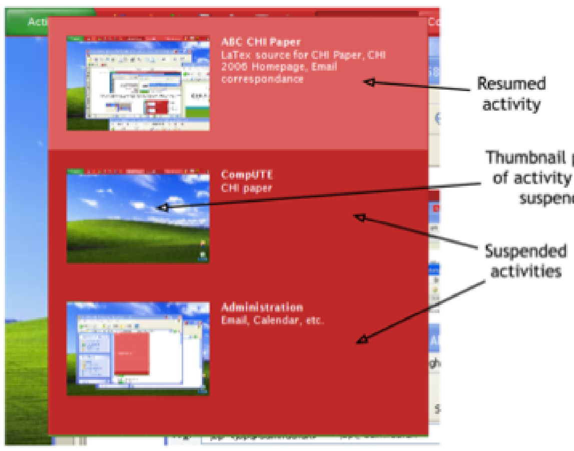

ABC CHI Paper
LaTex source for CHI Paper, CHI 2006 Homepage, Email correspondance
CompUTE
CHI paper
Administration
Email, Calendar, etc.
Resumed activity
Thumbnail pictures of activity as last suspended
Suspended activities

Figure 3.3: Activity dropdown menu

Source: [83]

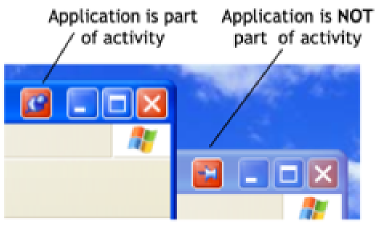

Application is part of activity
Application is NOT part of activity

Figure 3.4: Activity include button

Source: [83]

3.2.1 Architecture

Figure 3.5 shows the overall architecture in ABC framework version 4. A more detailed description of the architecture can be found in [24]. The client architecture and how it integrates with Windows XP is shown in figure 3.6. More details can be found in [16].

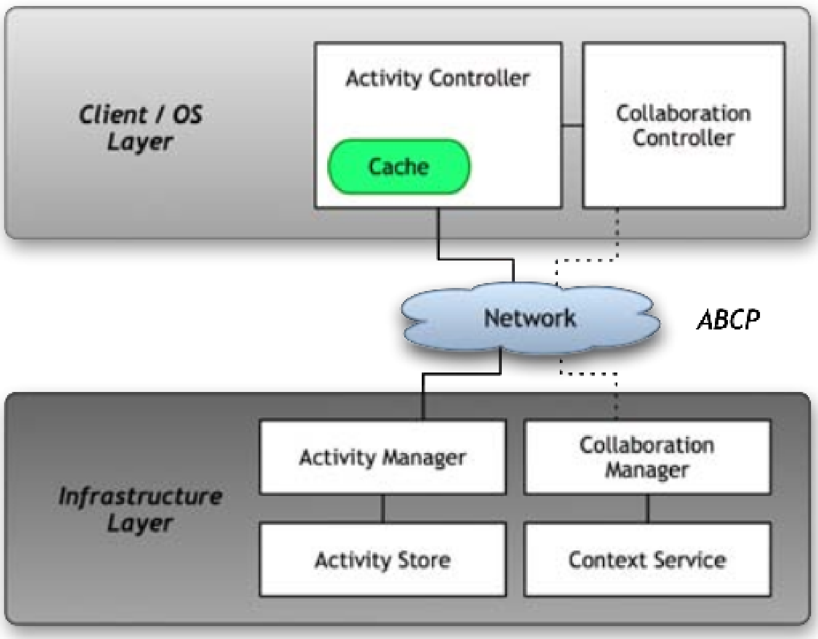


Figure 3.5: Overall architecture

Source: [24]

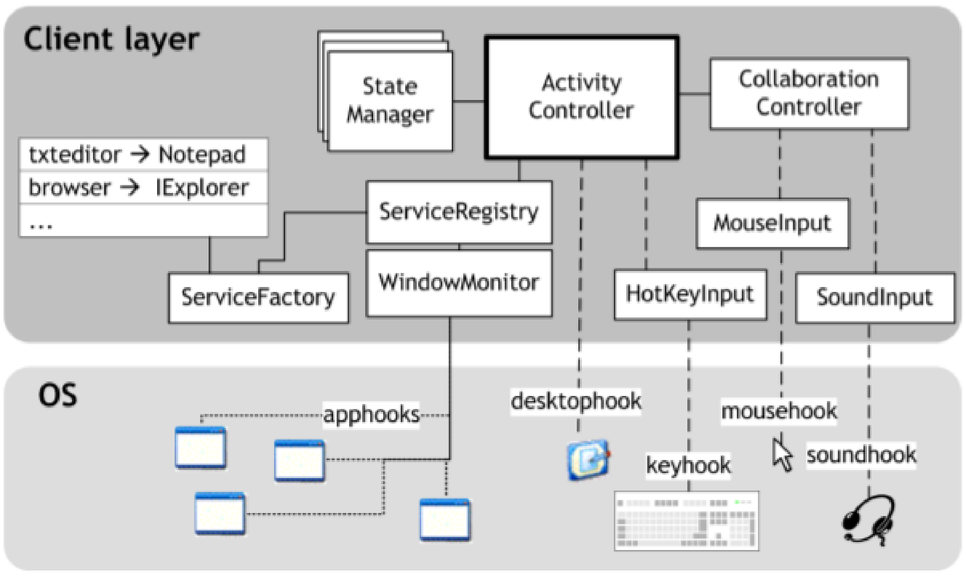


Figure 3.6: Client architecture

Source: [16]

3.3 co-Activity Manager

co-Activity-Manager (cAM) is another implementation of the ABC concepts. cAM is built for Windows 7, and integrates into the window- and task manager. Each activity will have its own virtual desktop [47].

co-Activity-Manager uses a cloud storage service (See 5.5) for keeping files, documents and activities synchronized across computers and users [45].

Figure 3.7 shows the user interface of co-Activity Manager. The different parts are:

A Virtual desktop workspace.

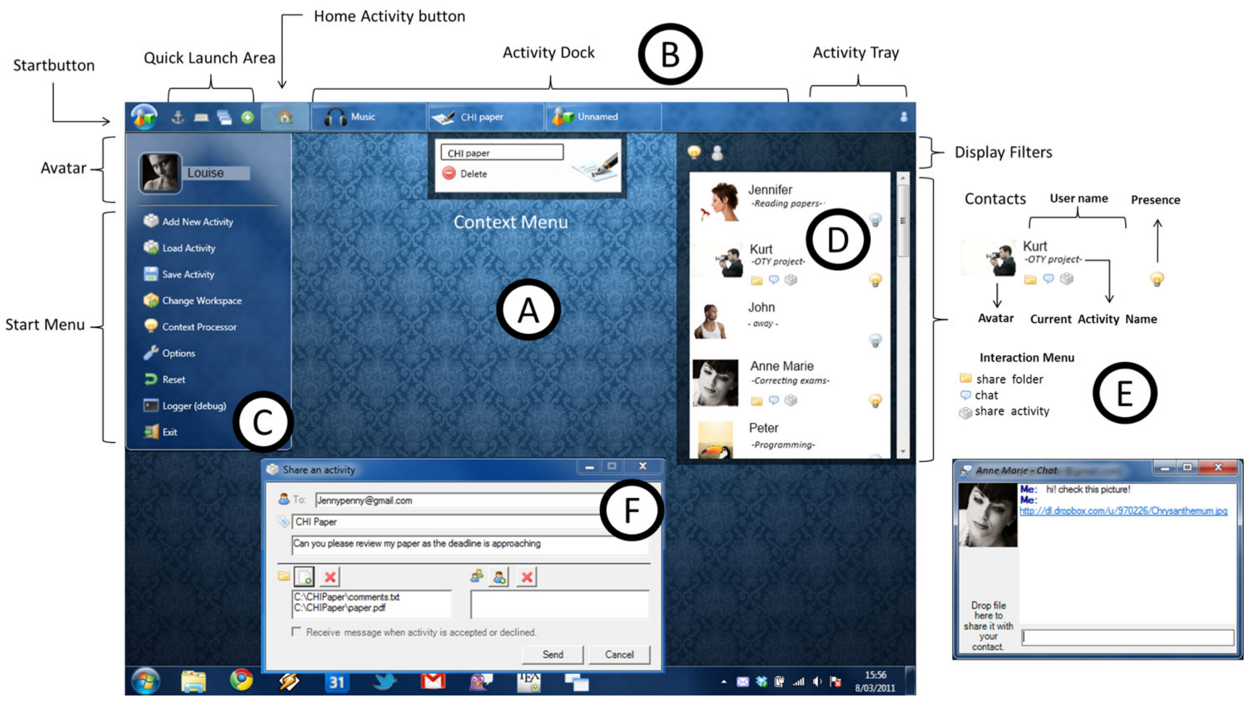
B Activity task bar, which lists all activities.

C Start menu for managing activities and applications.

D Collaboration manager.

E Interaction menu, for sharing folders and activities or chat.

F Share activity window.



To: Jennypenny@gmail.com
CHI Paper
Can you please review my paper as the deadline is approaching
C:\CHIPaper\comments.txt
C:\CHIPaper\paper.pdf
Receive message when activity is accepted or declined.

Me: hi! check this picture!
Me: http://dl.dropbox.com/u/970226/Chrysanthemum.jpg

Figure 3.7: co-Activity Manager

Source: [46]

Chapter 4

Cloud computing

This chapter presents cloud computing and the different service types used to provide a cloud-based service.

4.1 Services

Cloud computing can be split up in several different service models. The main three are Software-as-a-Service, Platform-as-a-Service and Infrastructure-as-a-Service [89, 96]. These service models are often called cloud layers [93] (See figure 4.1).

Other services keeps popping up, including Data-as-a-Service [91] and Network-as-a-Service [27].

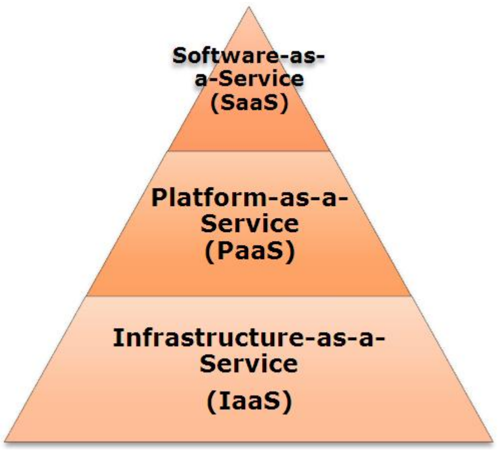

Software-as-a-Service (SaaS)
Platform-as-a-Service (PaaS)
Infrastructure-as-a-Service (IaaS)

Figure 4.1: Cloud layers

Source: [93]

Software-as-a-Service (SaaS) is the top layer, which is what the clients ”see”. Platform-as-a-Service (PaaS) is the middle layer, which is a software development environment, including storage and hosting. The base layer is Infrastructure-as-a-Service (IaaS), which usually refers to virtualization of servers.

These layers defines how much of the software/hardware stack the client is managing when using a service. The old model, with local packaged applications, every user or corporation must manage every step, from hardware to software. This requires both much expertise and investment in hardware. Basically, these layers are representing how much of the service is outsourced to a cloud provider. Figure 4.2 shows the differences between the three layers.

One advantage the three service models have, is that the cloud subscriber saves the cost of bying physical server and network infrastructure.

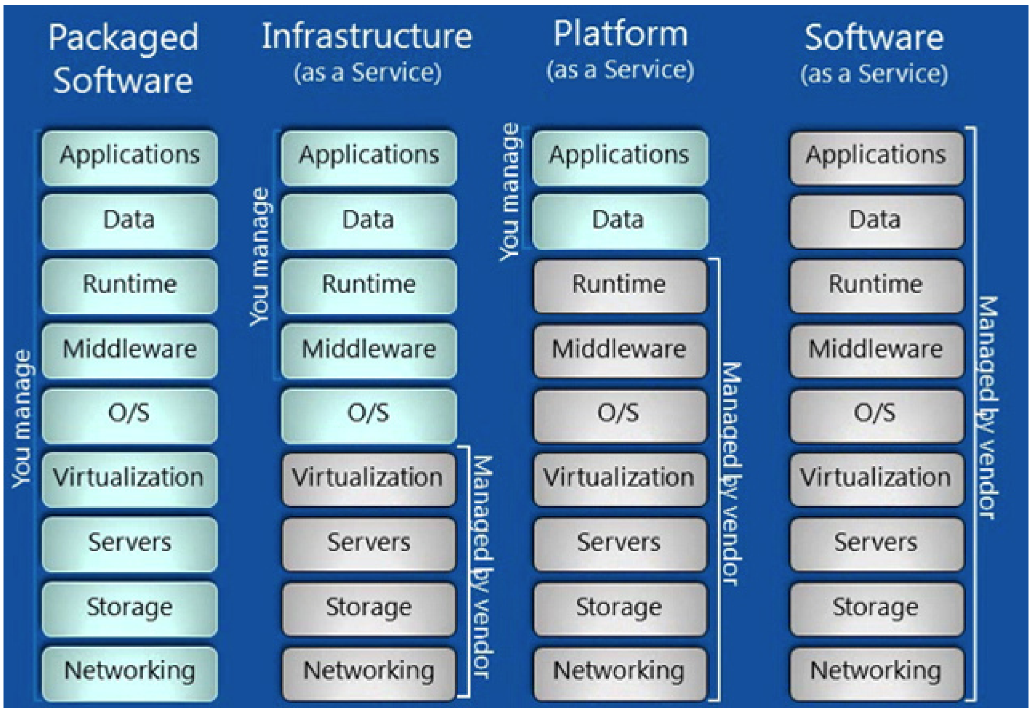


Figure 4.2: Layer management

Source: [53]

4.1.1 Software-as-a-Service

Software-as-a-Service refers to applications in the cloud. This is the service most users relate to when talking about cloud computing. SaaS is cloud computing pointed towards the end user.

For years, software companies have made applications that runs locally on a clients computer. This required both the infrastructure locally and people with knowledge on how to set it up and maintain it. Now, when the Internet has become so large and with higher capacity, SaaS introduces a new way of providing applications - over the Internet. Software providers deploy their applications on servers in the cloud. All code exists and is executed on these servers. SaaS applications are usually web-based, which means that a user typically only needs a web browser to run them.

SaaS advantages:

- **No extra costs in advance**

  No need to invest in additional hardware, servers, infrastructure or software [96]. A customer only pays for what he uses. All technical work is done by the software provider, so there is no need to hire more IT staff.

- **On-demand**

  Applications are provided on-demand. You only pay for what you use. A single user license can be used on multiple computers by the same user, no need to buy additional licenses on every computer running the application.

- **Available instantly**

  Before SaaS, when a company decides to start using an application, they have to first buy it and then install it to every computer. This could be a time consuming process. SaaS applications are available instanly - no need for deployment. And they are available to almost every computer at the same time.

- **Lower maintenance**

  All source code is located on the provider's end. This makes is very easy to implement new upgrades, which is immediately available to all users. The local IT staff does not have to worry about upgrading every computer to the latest software. This will reduce the overall maintenance cost for a company.

- **Available everywhere**

  As long as the user has a connection to the Internet and a web browser, the applications are available. The user does not need to use his own computer to have access to his applications.

- **Platform independent**

  The software provider does not need to take different architectures and operating system into account when developing an application. They just develop something that works on a web browser.

- **Scalable and dynamic**

  Because SaaS applications ”live” in the cloud, they are also scalable and dynamic as the cloud itself. Resources are easily accessible and if an application needs it, more resources are allocated instantly [96]. Another bonus is that a company does not have to plan for these future needs.

Control

When using SaaS applications, the cloud provider has the top level of control on every layer in the service, from hardware to software [14] (See figure 4.3). The client or cloud subscriber only has user level administration rights on the top level, which could include creating new user accounts or reviewing user activity - but only inside his scope of users. Cloud providers has the global administration rights, which include deployment, configuration and management - in other words, keep the service running. Responsibility for the different levels follows whoever has the control of it.

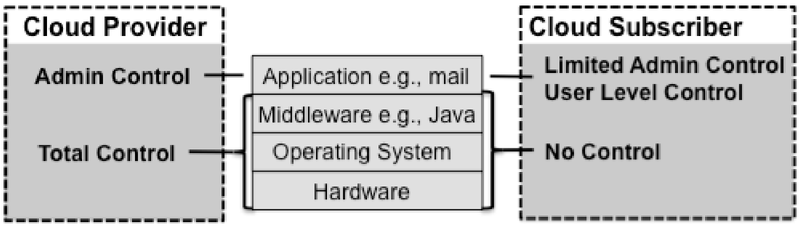


Figure 4.3: SaaS control and responsibility

Source: [14]

4.1.2 Platform-as-a-Service

Platform-as-a-Service is the middle of the three cloud layers (See figure 4.1). PaaS is directed towards application administrators and provides tools for developing, deploying and administration. Developers can usually utilize the included toolkit and environment for application development. This can be a runtime environment, programming languages, framework, etc. Applications deployed on a PaaS provider are built to support processing of large amounts of data, handle many, many consumers and be reachable from anywhere.

From a developers point of view, the PaaS work just like an arbitrary platform for deploying applications, e.g. a local server running Linux, with Apache web server and PHP support. This makes deploying applications on a PaaS cloud as easy as deploying them locally, if not even easier because of the included tools. Another bonus is that even though the application is available everywhere, there is only one entry point to upload code. The developer does not need to go into every server and update the application. It is easier to manage an application when all code is in one place.

As with SaaS, the customer does not need to think about servers and hardware resources or other infrastructure needs. PaaS providers handles all the underlying infrastructure and resources (processing, storage, network, etc.).

When an application has been deployed in a PaaS cloud, it works in the same way as a SaaS application (See 4.1.1).

Control

When developing application for PaaS clouds, the developer has full control over his own software, which is the top layer of the software/hardware stack [14] (See figure 4.4). The provider, on the other hand, has full control over both hardware and operating system. This is totally shielded from cloud subscriber, which only has access to the programming interface of the middle layer - the application platform.

With control comes resposibility. If something is broken in the application layer, then it is the cloud subscriber's responsibility to fix it. On the other side, if something in the middleware- or hardware layer is broken, it is the PaaS provider's responsibility.

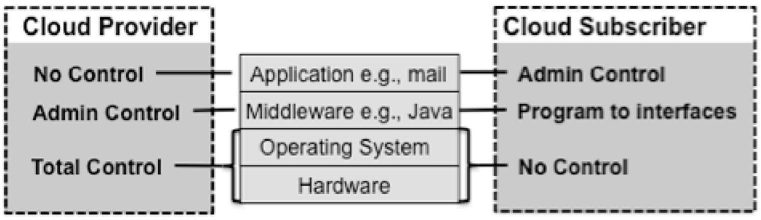


Figure 4.4: PaaS control and responsibility

Source: [14]

4.1.3 Infrastructure-as-a-Service

Infrastructure-as-a-Service (IaaS) is the bottom layer of the cloud layers (See figure 4.1). It provides computer resources on-demand. The most common is virtual servers. A large number of servers are running a hypervisor, which is a barebone operating system for hosting virtual servers.

These servers are organized in resource pools, and a single virtual server is not directly attached to a single physical server. On advanced resource pools, a virtual server can be moved between physical servers during runtime without causing downtime. This brings more flexibility and reliability to the virtual server. The customer or connected clients will not notice any difference because the network understands that the server is moving, and re-routes all traffic to the new physical server.

Each virtual server can be assigned more resources if needed, on-demand. This makes it easy to scale up and down according to the momentary needs. An example could be when a company has just released a new product, and many people will test the new product right after release. This results in a high-demand peak, which is far greater than the normal usage. With virtual servers, and huge resource pools, this extra computing power can be added instantly and released when the demand goes down. The benefit with this is that the company releasing the product does not have to invest in expensive hardware and infrastructure to handle peak loads which happens rarely.

Another great advantage with IaaS is that the customers has a full range of operating systems, application platforms and frameworks available. They can customize all software levels to fit their needs.

Control

With IaaS, the customer has complete control over the virtual server, from operating system to the end-user application. The IaaS vendor still has full control over everything hardware-related, and has administration rights to the hypervisor (See figure 4.5). The customer can make requests to the hypervisor regarding their own virtual servers, e.g. boot, reboot, shutdown, etc.

IaaS does, however, depend on every cloud subscriber to have local expertise on running servers. If something is broken with the operating system, lets say a broken kernel upgrade, then it is the cloud subscriber's responsibility to fix it. The cloud vendor does only have responsibility to keep the physical servers running and the hypervisors working.

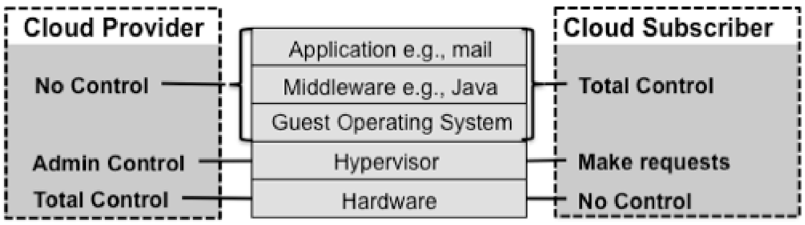


Figure 4.5: IaaS control and responsibility

Source: [14]

4.2 Cloud types

There are several different cloud computing types [14]. The type of choice is dependant on the usage and who the customer is. Each type has its advantages and drawbacks [29].

4.2.1 Public

Public clouds are entirely outsources to an external vendor and are publicly available to everyone across the Internet. These services are suitable for small projects, single persons, small business, as well as services/SaaS applications that are used by a great number of people, etc.

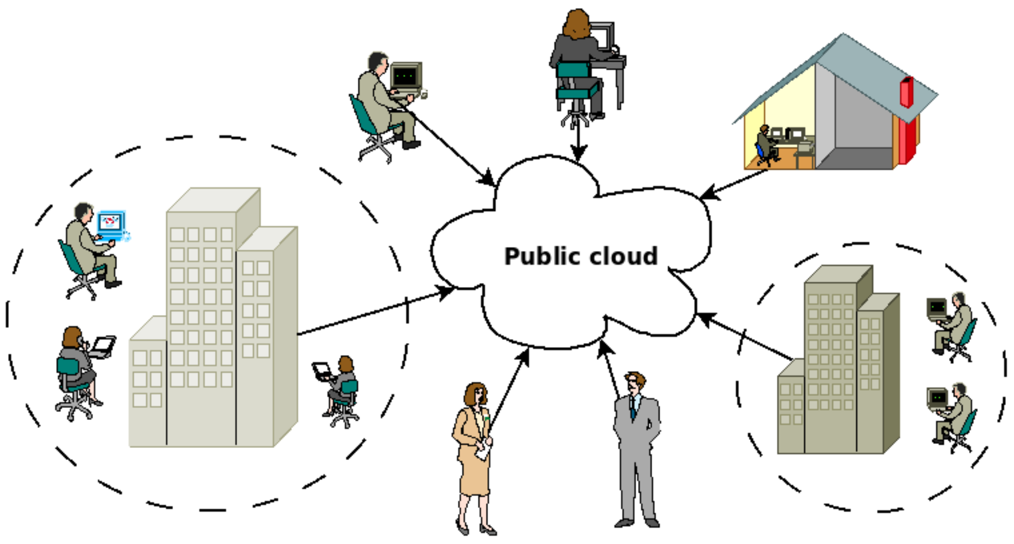


Figure 4.6: Public cloud

Advantages

- **No startup costs**

  Public clouds already has the infrastructure needed, which makes the initial cost of starting a service very low.

- **Scalability**

  If the service grows fast, it is easy to add more resources with a public cloud.

- **Pay-per-use**

  With public clouds, you only pay for what you actually use. This makes running a small service cheaper than larger services.

- **Less maintenance**

  All maintenance work is done by the provider. This saves the customer both time, staff and money.

- **Infrastructure sharing and utilization**

  Public clouds are already using data centers with the associated infrastructure. The capacity of these are rarely maxed out, and by sharing capacity with others, the existing infrastructure can be utilized better.

Disadvantages

- **No data control**

  When using public clouds, the cloud subscriber does not have full control over its data. Although public clouds usually have good security, the data is still stored on the providers servers and gives the provider the ability to access it. All data is also transferred across a public network (Internet). This makes public clouds not best suited for information sensitive services.

- **Limited customizability**

  Public clouds does not always support your needs for a specific operating system, programming plaform or applications.

4.2.2 Private

Private clouds are built on the same architectural principles as public clouds. The main difference is that a company or organization runs the cloud service itself. Private clouds usually only pays off if the company is large enough to fully utilize it as opposed to a public cloud.

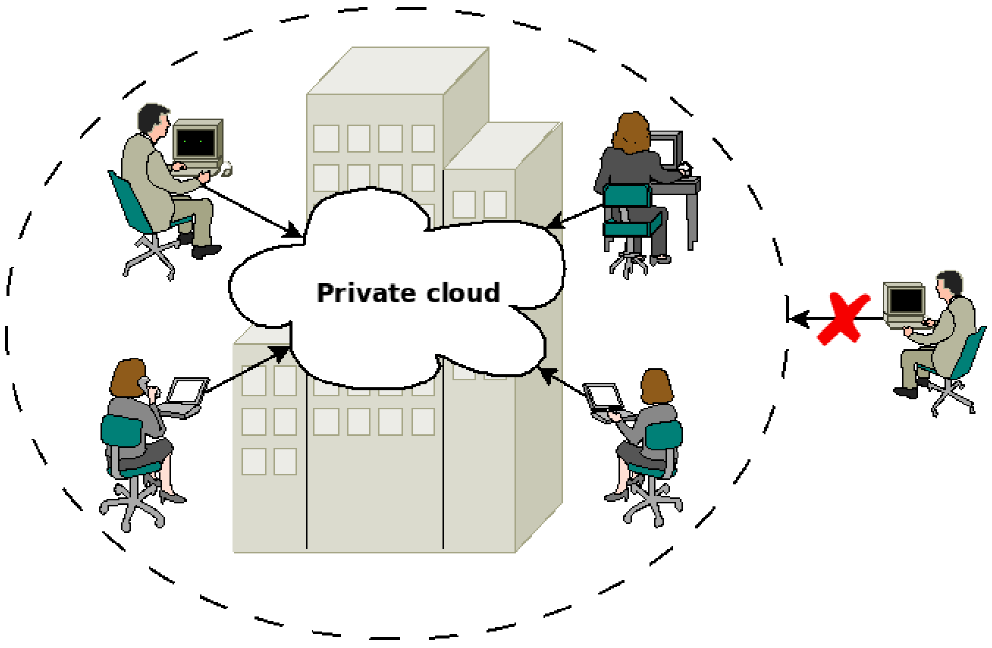


Figure 4.7: Private cloud

Advantages

- **Data control**

  When using private clouds, the company itself operates all the servers, data storage and infrastructure and thus have complete control of every aspect of the cloud. As opposed to public clouds, no external vendor can access the data.

- **Highly customizable**

  Private clouds can be built and customized as needed. This means that the choice of operating system, platform, etc. is entirely up to the company itself.

- **Restricted access**

  Only people with the correct permissions can access the cloud. Unauthorized attempts are restricted.

- **Performance**

  If the private cloud is hosted within the same building or in the local area close by, the performance of the cloud is not bottlenecked by the network speed. The infrastructure is not shared with others, and all capacity is dedicated to the company.

Disadvantages

- **High initial cost**

  Servers, data centers/server rooms and infrastructure must be invested in before the cloud service is up and running. This is expensive and brings the initial cost up.

- **Maintenance**

  All maintenance must be performed by the company. Hardware failures needs replacement, software upgrades and patching must be done, etc. Everything has to be handled locally, and thus depends on maintenance staff.

4.2.3 Community

Community clouds combines elements from both public- and private clouds and are becoming more and more popular. They are not public, but not private either, more like a compromise between the two. Several companies or organizations collaborate to create a common shared cloud.

Community clouds suits smaller businesses and companies better, but is also appealing for larger companies. This is because of the lower initial cost of investing in hardware. Risks are spread, expenses, maintenance and infrastructure is shared between the participating companies. It is not as cheap as a public cloud, but cheaper than a private cloud.

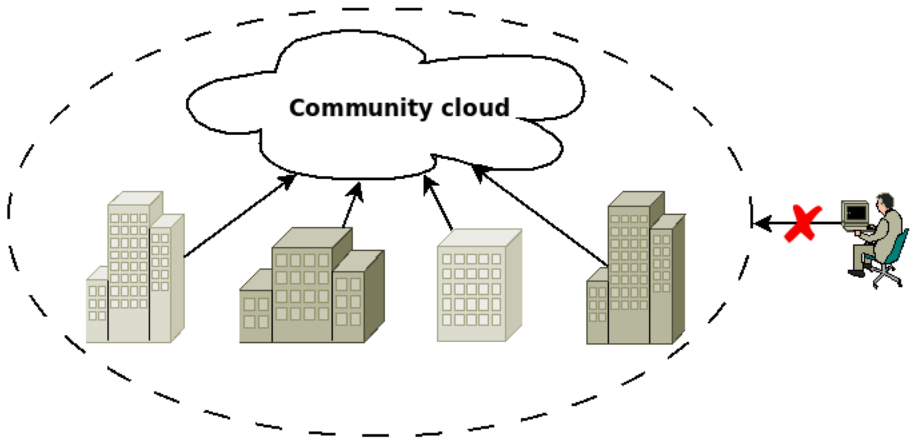


Figure 4.8: Community cloud

4.2.4 Hybrid

Hybrid clouds are a combination of multiple cloud types. An example could be if a company handles normal usage through its own private cloud, and when needed, utilize public- or community clouds for additional computing power. This is useful when delivering a service which gives high peak load, but fairly low average load. Instead of investing in large quantities of servers and infrastructure to handle peak loads, the company can invest according to normal usage, and then outsource and pay-per-use additional computing resources.

Another bonus with hybrid clouds are the redundancy and higher availability it gives. If the private cloud service is down, the public cloud service can ”take over”.

As with public clouds, a drawback is that data existing on the public cloud is not in the owners complete control. This means that hybrid clouds are not suited for every type of tasks.

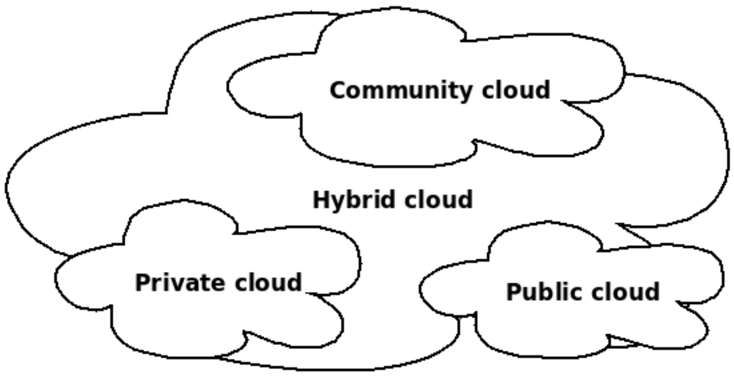


Figure 4.9: Hybrid cloud

4.3 Cloud services providers

4.3.1 SaaS providers

Google

Google provides SaaS applications through Google Apps [39]. This is a collection of web-based applications, such as Drive, Calendar, Gmail, etc.

Dropbox

Dropbox is a very popular file syncronization service (More in 5.5.1). Every file you add to Dropbox will be syncronized instantly to all other devices you have registered with Dropbox. You can also share your files with others with just specifying who you want to share a folder or file with.

Salesforce

Salesforce has several different SaaS applications [76] and are primarily known for their customer relationship management software - Sales Cloud. Other applications include Service Cloud and Marketing Cloud.

Zoho

Zoho is providing many SaaS applications online [26]. They group them into three categories - Business, Collaboration and Productivity. One of the collaboration apps is Zoho Docs, which is an online document management software, including Word Processor, Spreadsheet and Presentation.

4.3.2 PaaS providers

Google

Google App Engine [38] is Google's PaaS product and provides an environment for developing web applications. These applications runs on Google's servers. Google is known for its reliable and high performing infrastructure, and gives the same benefits to applications made with Google App Engine.

App Engine supports several programming languages, including Java [70], Python [32], PHP [42] and Go [85]. This makes it quite versatile regarding developers' choice of programming language.

Google App Engine makes integrating Google accounts into web applications easy. This can reduce the development time of an application because the user authentication service is already implemented by Google. Another benefit is that users does not need to create a new user to use the application - if they have a Google account. Google is one of the companies with an already huge user database.

Microsoft

Windows Azure [62] is Microsoft's cloud service. It provides both IaaS and PaaS. Windows Azure is spread across several data centers, and includes automatic load balancing. Scaling of applications happens without changing the application code, and works in a pay-per-use way. You only pay for the resources you use.

Azure makes deploying web applications easy. It is tightly integrated in Microsoft's Visual Studio ecosystem. You can manage your Azure web applications directly within Visual Studio.

Windows Azure supports several programming languages, including ASP.NET [55], PHP [42], Node.js [48] and Python [32].

4.3.3 IaaS providers

Amazon

Amazon is an IaaS provider with the Amazon Elastic Compute Cloud (EC2) [2]. This is a web service for providing virtual servers in Amazon's cloud infrastructure. When you need more computing resources, you can have an Amazon EC2 virtual server up instantly. It is also equally easy to stop a virtual server, making Amazon EC2 really an elastic on-demand cloud service.

Rackspace

Rackspace [74] provides virtual servers running either Linux or Windows, and a web interface for creating and managing these. The cloud architecture of Rackspace is based on OpenStack [69]. This means you are not locked to the same vendor, and that servers you create on Rackspace can be moved to another IaaS provider.

Chapter 5

ABC concepts in software products

This chapter covers the analysis of Activity-Based Computing concepts in different software products. All of these products incorporates features similar to one or several of the ABC concepts. The products are arranged in products groups. These are virtual desktops (5.1), remote desktop (5.2), remote applications (5.3), office suites (5.4), file synchronization services (5.5) and other products (5.6).

5.1 Virtual desktops

Virtual desktops is an old concept. Ever since windows-based operating systems was developed, people like to have several windows open at the same time. This became a problem, due to the small screen size and low resolutions. The desktop was often too small for many applications and became crowded. A resolution to this was to have several desktops. Virtual desktop was first introduced in 1986 with Rooms [43]. Today, both Ubuntu [51] and OS X [7] incorporates this feature natively within the desktop environment. Figure 5.1 shows a typical view of four virtual desktops in Ubuntu.

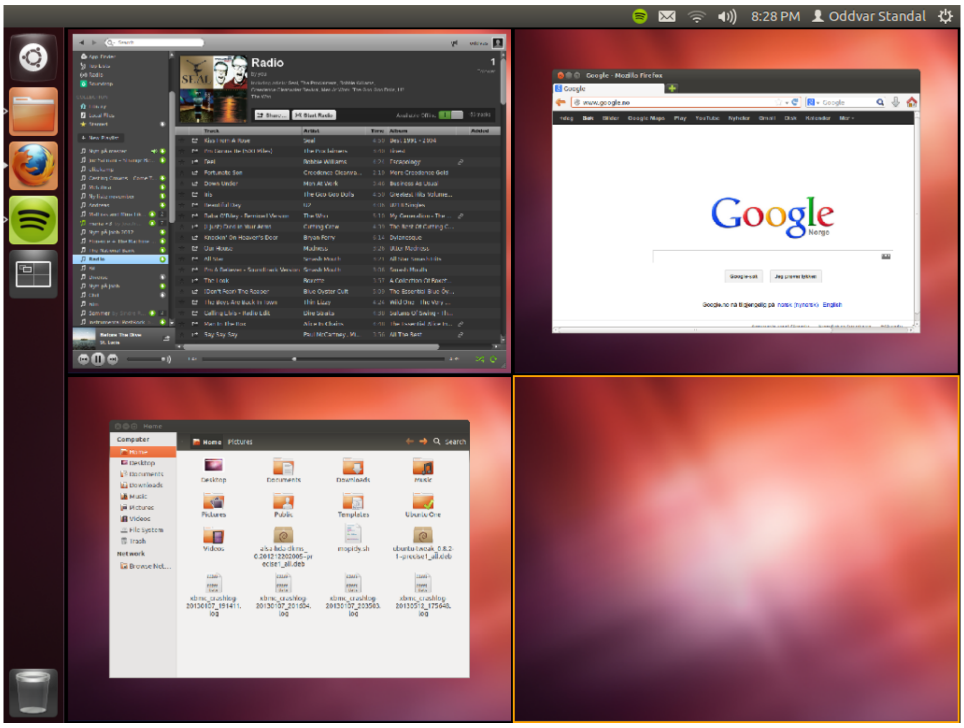


Figure 5.1: Virtual desktops in Ubuntu

With increase in both screen size and resolution, we are now able to put more applications on a single desktop. The need for virtual desktops in terms of desktop space is not necessarily the main reason to use virtual desktops anymore. As people like to multitask, virtual desktops are more and more used for separating tasks - like ABC's Activity-Centered concept (See 2.1).

5.1.1 Workspaces

Virtual desktops in Ubuntu are just called Workspaces. There is no limit to the number of workspaces/desktops that can coexist, and each workspace can be renamed to whatever the user wants. Applications are normally opened in the current workspace, but you can also make programs open in a specific workspace every time. It is also possible to send individual applications to another workspace during runtime. If an application is very important, or is in use all the time, it can be displayed on all workspaces. This can be useful for an instant messaging application when the user wants to be available across activities.

With many workspaces, each workspace can be dedicated to a certain activity. This resembles the most fundamental concept of Activity-Based

Computing, which is Activity-Centered (See 2.1).

Although we can say that Workspaces supports suspend and resume, it is limited to the extent of the current session. When a user log in on a computer, he can start several applications and split them into different activities in different workspaces. As the user switches to another workspaces, he ”suspends” the current activity and resumes the other. This works fine as long as the user is logged in. Whenever he logs out or reboots the computer, all the workspaces are reset and emptied. The next time the user log in, he must configure every activity again. Even if certain programs might have been configured to start in a specific workspace, the ”activity” is not resumed. The same program can be a part of another activity as well, an thus it needs to be opened and moved every time. This is why Workspaces have some similarities to, but does not include every aspect of, the Activity Suspend and Resume concept (See 2.2).

Ubuntu’s default desktop environment, Unity [52], have an app called Workspace Switcher. This litte application has mainly one function, and that is to switch between workspaces. When you click on the button on the taskbar, it opens all virtual desktops and displays them in a grid (See figure 5.1). This gives a nice overview of every workspace, and makes it easy to change activity.

Desktop Pager is a similar app and is used in more lightweight desktop environments available - like LXDE [54]. Figure 5.2 is showing how Desktop Pager integrates with the taskbar in LXDE when having three workspaces. Each button represents the workspace with a miniature image of the application layout. With Desktop Pager in the taskbar, it is as easy to switch between activities as to switching between individual applications. This is quite similar to the Activity Bar in the ABC-framework (See 3.2).

Another way to quickly switch between workspaces is to use keyboard shortcuts, e.g. *Ctrl + Alt + Left/Right* for previous/next.

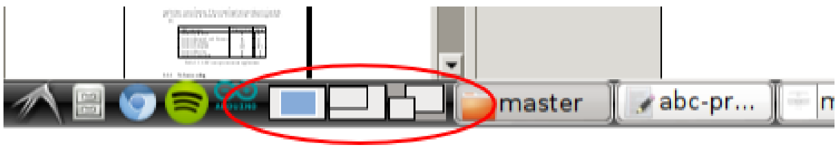


Figure 5.2: Desktop Pager

5.1.2 Mission Control

Virtual desktops in OS X are called desktop spaces. With OS X Mountain Lion, Apple introduced Mission Control [6]. Mission Control is a combination of Exposé, Spaces, Dashboard and full-screen apps (See figure 5.3). Virtual spaces are handled by Spaces. Mission Control provides the user with an overview of every running app on the computer. The different virtual spaces are accessible through the top menu, while every application are displayed in the lower part. Each window are grouped according to what program it belongs to. This makes it fast and easy to switch between tasks.

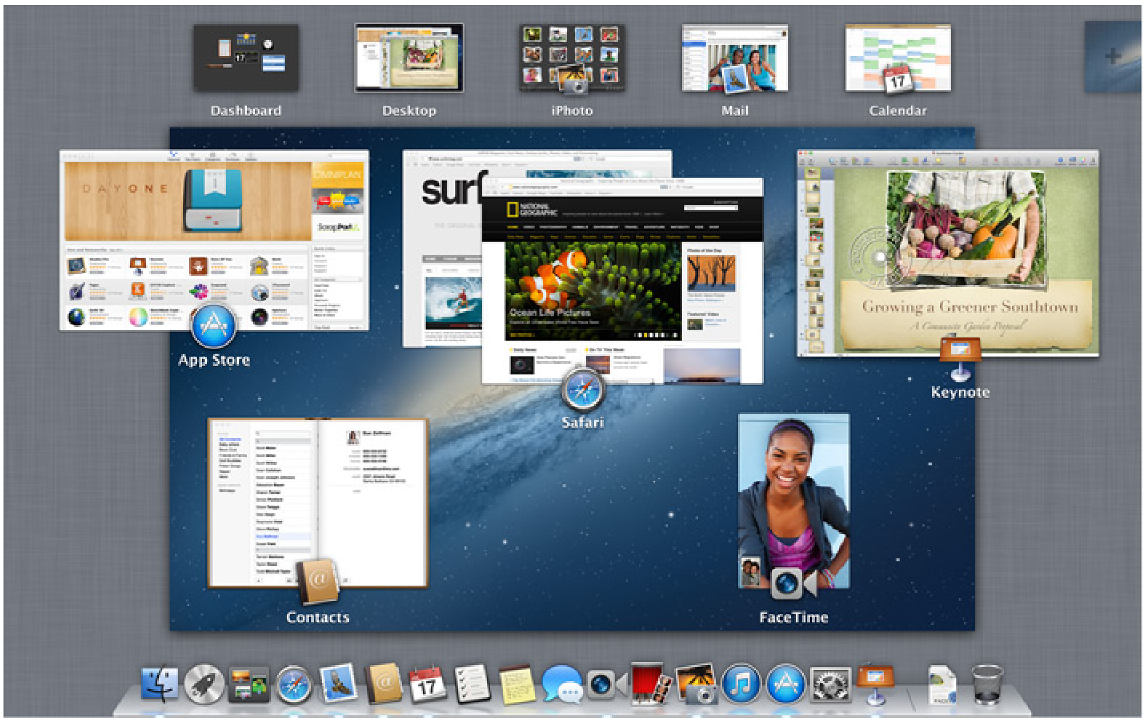


Figure 5.3: Mission Control

Source: [6]

The main principle of virtual desktops is still valid with Mission Control. A user can devide different tasks across different desktop spaces. Mission Control makes switching between them fast and easy with a simple swipe or click. This compares to the Activity-Centered concept of Activity-Based Computing (See 2.1).

As with Workspaces, Mission Control does not support suspending and resuming across sessions. Within a single session, desktop spaces can be used to suspend an activity by switching to another virtual desktop. It can then be resumed later on. However, when the user logs off, he has to start all over again the next time. This is why the Suspend and Resume concept (See 2.2) is only partially comparable.

5.1.3 Comparison of virtual desktops

Table 5.1 gives a summary of the ABC concepts in the two virtual desktop systems.

ABC concepts	Workspaces	Mission Control
Activity-Centered	✓	✓
Activity Suspend and Resume	(✓)	(✓)
Activity Roaming	.	.
Activity Adaptation	.	.
Activity Sharing	.	.
Activity Awareness	.	.

Table 5.1: ABC concepts in virtual desktops

5.2 Remote desktops

Remote desktop is a way of using a computer remotely. A user connects to e.g. a server, logs in and is presented with a complete desktop experience. This looks just like if the user was working on his own computer. Remote desktop can also be used on a personal computer, for instance at home. Lets say that a person has a computer at home which is always on, and he has his email client, with all his email accounts configured and every email downloaded on it. He also has his favorite web browser with all bookmarks, passwords stored and tabs open. All his documents are stored on the computer and it has an office suite installed. Remote Desktop makes this personalized computer experience available from another computer or mobile device.

When remote desktop is used with a server, then multiple clients can connect at the same time and have their own sessions. This introduces thin-clients. A thin-client is a computer with minimal computing resources. It has mouse, keyboard, sound, network, USB-ports for storage and a display. The only task for a thin-client is to connect to a remote desktop server and present a desktop-like environment for the users. Because a server can handle multiple clients, this is a very effective way of utilizing computer resources and at the same time save energy. Another benefit of thin-clients is the cost. They are cheaper than regular computers. This model is normally used in corporations where they need workstations for many people without having to invest in expensive hardware for every user.

5.2.1 Microsoft Remote Desktop Services

Microsoft Remote Desktop Services (formerly Terminal Services) [61] consists of both server and client software for remote desktop. It is using the proprietary Remote Desktop Protocol (RDP), which is developed by Microsoft and is a popular remote desktop protocol. When a user connects to a server with Remote Desktop Services, he starts a session based on his username. A single user can only have one session on the same server, however he can connect to as many servers he want. If he then splits up each session to a single activity, you have something that could look like an Activity-Centered concept (See 2.1), but it is just a way of using it. Switching activities in this way is about as fast as switching between applications, but then the user has to have more than one server accessible. Remote Desktop Services does not support any Activity-Centered aspects by itself.

Remote Desktop Services supports basic screen resolution changes, which means that it is easily adaptable to devices with smaller screens. Screen size is mainly the difference between clients. Because all program code, both from the remote desktop environment and the applications within it, runs on the server, local resources are not heavily in use. This means that Remote Desktop Services supports adaptation to most devices, and thus resembles the Activity Adaptation concept (See 2.4). The biggest limit is the usefulness of a full sized desktop environment on a small screen, which leads to a lot of zooming and panning across the screen.

Remote Desktop Web Access

Remote Desktop Web Access is a part of Remote Desktop Services and is a web-based implementation of the client. Although RDP-clients exists on most operating systems and devices (Figure 5.4 is showing Remmina [75], a RDP client for Linux), Remote Desktop Web Access expands the horizon for computers and devices without one. This is practical if a user is using a computer with limited permissions, and does not have the ability to install an RDP-client. He can open a browser, point it towards the Web Access server and connect to his personalized desktop from anywhere. Altogether, this makes Remote Desktop Services accessible from anywhere, which compares to the Activity Roaming concept (See 2.3).

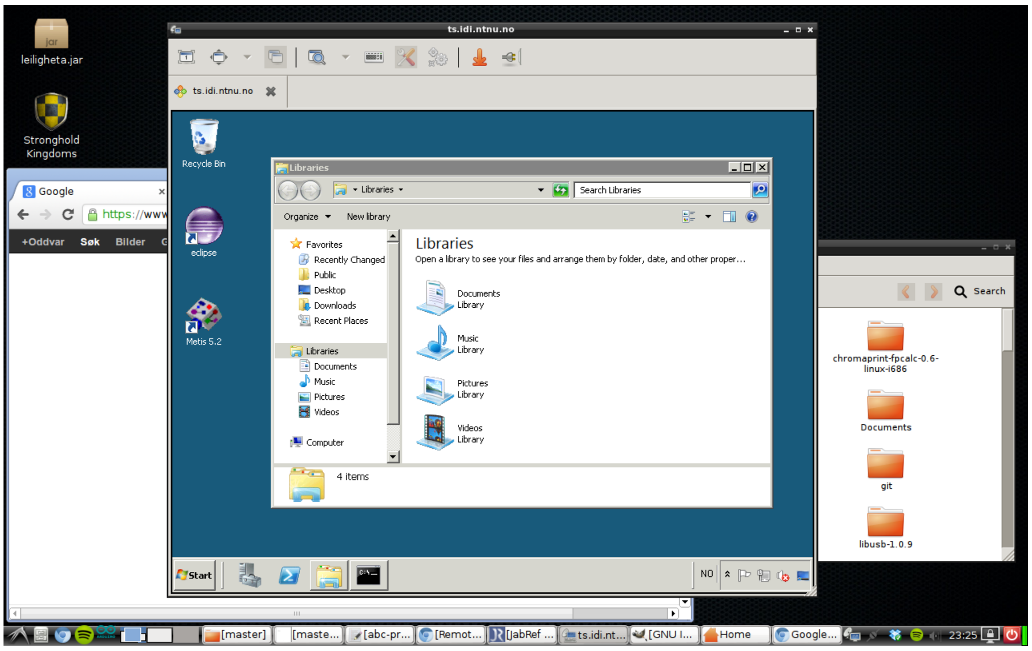


Figure 5.4: Connecting to Remote Desktop Services from Linux

5.2.2 Windows Desktop Sharing

Windows Desktop Sharing [63] is a technology for sharing a desktop session between multiple users. It is still remote desktop and use the same protocol (RDP) as Remote Desktop Services. Users can connect to a desktop in a view-only mode, or with full access. Full access means that they can control input devices like mouse and keyboard. This can be useful when a person needs remote assistance or a group of people are collaborating and conferencing.

Remote assistance Windows Desktop Sharing provides a synchronous view of a user's desktop, with the person needing assistance on one computer, and the person assisting on another computer. This has become a popular way of helping users with computer software-related problems. Instead of using phones and try to talk through the problems, it is easier to just share the whole desktop. This way the assisting person can guide the other on how to solve several problems.

Collaboration and conferencing Another use of Windows Desktop Sharing is collaboration and conferencing. A group of people share the same desktop view, which is synchronized live between all participants. Every user can interact with the desktop. This also includes group chat and voice/video communication.

With Windows Desktop Sharing, the computer which is running the shared desktop has to use Windows. The clients can, on the other hand, connect from any compatible device with an RDP-client. This resembles the Activity Roaming concept of ABC (See 2.3). Similar products include Team Viewer [90] which is a multi-platform desktop sharing and collaboration software. Team Viewer works on Windows, Mac OS X, Linux, in addition to clients on iPhone and Android, and thus is more versatile than Windows Desktop Sharing.

Bottom line is, Windows Desktop Sharing and other desktop sharing solutions builds on the same principle as synchronous Activity Sharing (See 2.5).

5.2.3 Comparison of remote desktops

Table 5.2 shows which ABC concepts exists in remote desktops.

ABC concepts	Remote Desktop Services	Windows Desktop Sharing
Activity-Centered	(✓)	.
Activity Suspend and Resume	✓	.
Activity Roaming	✓	✓
Activity Adaptation	✓	.
Activity Sharing	.	✓
Activity Awareness	.	.

Table 5.2: ABC concepts in remote desktops

5.3 Remote applications

Remote applications is a way of running an application on a server, while displaying it locally. They work in a similar way as remote desktops. The difference is that instead of connecting to an entire desktop environment, you only see the applications you are running. These remote applications integrate seamlessly into your own desktop environment, and looks like just another application.

A benefit of remote applications is that IT administrators has one place to manage the software, instead of doing the same maintenance work on every employee's own computer. Another benefit is that corporate software with expensive licenses are not spread across many computers, with the potential of releasing the licence to other users. This means one less concern for IT administrators.

5.3.1 Microsoft RemoteApp

Microsoft RemoteApp [57] is a part of Microsoft Remote Desktop Services (See 5.2.1). The technology behind is the same. It runs a single application remotely instead of connecting to a desktop session. Each application starts its own connection. This means that, unlike Remote Desktop, it does not support suspending and resuming. Remote applications in RemoteApp starts fresh every time.

RemoteApp is also available everywhere, as long as the client has an internet connection. This is similar to the Activity Roaming concept (See 2.3).

Because RemoteApp builds on the Remote Desktop Protocol, there exists many clients on most operating systems and mobile devices. This way, RemoteApp can bring desktop applications to mobile devices. The problem with this is the smaller displays. One solution is to just make the application span across a larger virtual screen, and letting the user pan around inside. This might resemble the Activity Adaptation concept (See 2.4).

An example from NTNU is the Software Farm [67]. This builds on Microsoft Remote Desktop and RemoteApp. Students and employees at NTNU have access to software from anywhere without needing to install them locally. All they need is a computer with a Remote Desktop client installed. Running software remotely on a server also gives the possibility to boost performance, especially on heavy programs since most servers are more powerful than the computers the clients use.

5.3.2 X-forwarding

X Window System (X) [73] is a system for managing Graphical User Interfaces (GUIs). It is cross-platform and uses the client-server model. Most of the Linux-based operating systems are using the X Window System for displaying a GUI. X basically consists of a software called an X-server. It acts as a layer between the application and the screen. The X-server takes inputs from a mouse, keyboard, touch screen, etc. and displays programs on a monitor connected to the computer. Every application with a GUI acts as a client and will try to connect to the local X-server and sends all visual parts to it. The X-server handles multiple incoming connections and displays them as different windows.

The most common implementation of X Window System is the open source version made by the X.Org Foundation [94] in cooperation with freedesktop.org [33].

X-forwarding is a method for displaying applications running on a remote computer, for instance a server, on a local device. All parts of the software runs on the server, including the GUI, but instead of displaying the GUI on the server, it is transferred across the network to the clients local X-server. The remote computer does not even need to have a monitor connected, or even a local X-server running.

When talking about X in terms of clients and servers, the different roles must not be confused. All applications are clients and connects only to a single X-server (See figure 5.5). However, with X-forwarding, the roles becomes inverted. The client is connecting to the server, and forwards his own X-server, which make it possible for applications to connect to the clients X-server (See figure 5.6).

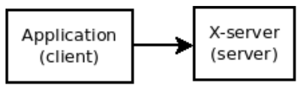


Figure 5.5: X application perspective

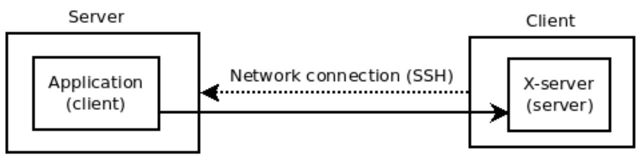


Figure 5.6: X-forwarding perspective

X-forwarding is often used in combination with a Secure Shell (SSH) [68]. If you are working on a computer with an X-server running, you can just type in this simple command in a terminal:

```
ssh -X username@remoteserver
```

The key here is the -X, which enables X-forwarding. With this you can connect to a server, log in, start a program remotely, and then it shows up on your local display.

Since the X Window System is cross-platform, and only used for displaying a graphical user interface, it can be used on almost every device. An open source Android version is available [86], and thus enabling displaying of X applications on mobile devices. This resembles ABC's Activity Adaptation concept (See 2.4), however, this is quite limited. Even though it is possible to resize programs to fit the small screen on mobile phones, its user interface is not built for these small screens. On the other hand, it does not require as much from the hardware on the device, since most processing is done on the server.

Because X is designed on a client-server model, it can be used over a network connection, and thus from almost everywhere. This feature is similar to ABC's Activity Roaming concept (See 2.3).

5.3.3 X Persistent Remote Applications

X Persistent Remote Applications (Xpra) [95] is based on the same principle as regular X-forwarding (See 5.3.2). The difference is that instead of starting the programs every time you connect to a server, the programs are running in the background when you are not connected. This means that you can resume applications on different devices, from different locations. Under normal circumstances, this is impossible due to the fact that applications can not switch between several X-servers while running. When an application starts, it connects to an X-server and is dependent on that connection the entire runtime.

Xpra is built on a client-server model. What Xpra has done, is making a proxy-like X-server the applications can connect to. This X-server is implemented in the Xpra server. The Xpra client connects to the server (either local or over a network connection). The X-server with all the remote applications in Xpra server is connected to the X-server running the Xpra client. This will display these remote applications on the clients (See figure 5.7).

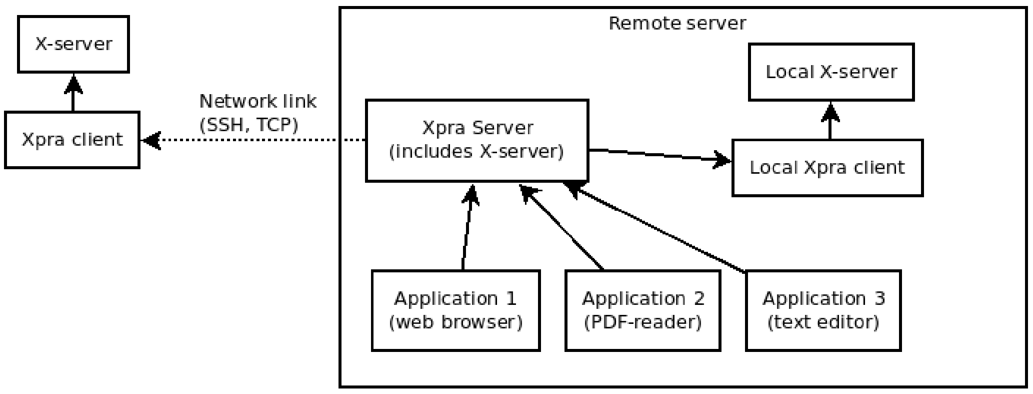


Figure 5.7: Xpra overall design

You start the Xpra server on the computer that will run the programs.

To view the programs, you must start the Xpra client and connect it to the server. Xpra can also handle several sessions, which is called displays. When you start an Xpra server, you define what number the display should have. On the remote server, you can start Xpra with this command:

```
xpra start :5
```

This starts an Xpra server with displaynumber 5. To connect to this session, you can either type:

```
xpra attach :5
```

for local access, or:

```
xpra attach ssh:username@remoteserver:5
```

for remote access through SSH [68]. A third way is to combine X-forwarding with Xpra. This is practical when the device you are using does not have the Xpra client. With X-forwarding, it is enough to have a local X-server running, and Xpra installed on the remote server. This gives a wider range of devices you can access your programs with. ABC's Activity Roaming (See 2.3) concept is comparable to this feature.

When attaching a remote Xpra session, all programs in that session will pop up on your desktop as if they were local programs. You can also attach to many sessions at a time, and when you disconnect a session, the programs will still be running on the server, but closed on your desktop. With a simple attach, it is easy to resume your work at anytime. This is similar to ABC's Activity Suspend and Resume concept (See 2.2).

You can start an Xpra session on display 5 for surfing the web, and another session on display 6 for programming in Eclipse. If you need to switch between the two, you just disconnect the current session, and attach to the other. Or you can leave both connected. This resembles ABC's Activity-Centered concept (See 2.1).

Xpra also has some limited similarities to ABC's Activity Adaptation concept (See 2.4). It can be used on many devices, including mobile phones. But, as with X-forwarding, the user interface can become a limiting factor when desktop applications gets resized to fit many different screen sizes, especially small screens.

5.3.4 Comparison of remote application technologies

Table 5.3 compares different remote application technologies with the ABC concepts.

ABC concepts	RemoteApp	X-forwarding	Xpra
Activity-Centered	.	.	(✓)
Activity Suspend and Resume	.	.	✓
Activity Roaming	✓	✓	✓
Activity Adaptation	(✓)	(✓)	(✓)
Activity Sharing	.	.	.
Activity Awareness	.	.	.

Table 5.3: ABC concepts in remote applications

5.4 Office suites

Office suites have been, and still is, very popular and necessary for many people, including home users, students and corporations. The largest office suit vendor has been Microsoft with it's Microsoft Office package. As Cloud computing has become more and more popular, online office suites has made their arrival. A well known provider is Google with its Google Drive (formely Google Docs), which is part of Google Apps [39]. Microsoft on the other hand has also jumped on the Cloud computing hype with a cloud-based version of Microsoft Office - Office 365 [59].

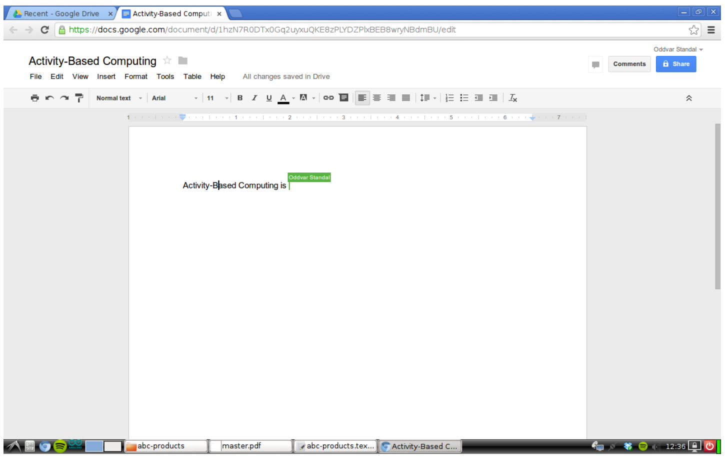
5.4.1 Google Drive

Google Drive [41] is primarily a cloud storage service (See 5.5). However, it incorporates a full featured online office suite - also known as Google Docs. This section will focus on the office suite in Google Drive. Google Docs is the former name for this service, but now, Google Docs has become the name of the word processor in Google Drive. Other applications includes Google Sheets - the spreadsheet editor, Google Slides - the presentation editor, and Google Drawings.

Both Docs, Sheets and Slides saves automatically. Every little change is registered and stored. This helps to provide the in-built version control system, which means that you can go back to a previous version if you want. With automatic saves, you do not have to worry about remembering to save every time you close your browser. You can simply just close it and later on, when you want to continue writing, you can open the same document, which is exactly as you left it. The only small drawback is that you always go back to the beginning of the document. If it is a large document, with hundreds of pages, it may take some time to find the position you left it at. This does, however, resemble the Activity Suspend and Resume concept (See 2.2).

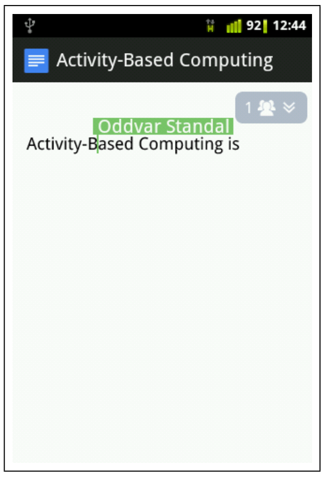
Because Google Drive is web-based, it is reachable from anywhere, and any browser. In combination with automatic saves, you can close the browser on one computer, and resume your writing in another browser on a different computer. As long as the computer has Internet connectivity, you can use it for Google Drive. This is similar to the Activity Roaming concept (See 2.3).

Google Drive supports several mobile devices, including devices with Google's own Android operating system. Figure 5.9 shows how documents are viewed and edited on an Android smartphone. The same document is shown on a desktop web browser in figure 5.8. This shows how well Google Drive adapts to different devices, especially to the small screens on smartphones. Google Drive supports the same principles as the Activity Adaptation concept (See 2.4).


Activity-Based Computing
File Edit View Insert Format Tools Table Help
All changes saved in Drive
Comments
Share
Oddvar Standal

Activity-Based Computing is

Figure 5.8: Google Drive



Activity-Based Computing is

Figure 5.9: Google Drive on Android

The need for collaboration on documents has become increasingly important. Google Drive supports both asynchronous and synchronous collaboration. With a click on the share button, a document can be shared to one or several other people. These people will instantly get the document in their Google Drive folder and can open it in a web browser. The owner can decide if new users gets write permissions, or just read permissions. If they have write permissions, they can edit the document. This can be done one at a time, and everyone will have the latest version, which is asynchronous collaboration.

If the users have the document open simultaneously, they can edit it, and in real-time, see each other's modifications. When editing a text document with multiple people, each will have their own color-coded cursor marking where they are, similar to tele-pointers (See 3.1). Instead of sharing the view when collaborating, Google Drive only sends small data packages containing the state of every participant, which enables multiple users to collaborate on the same document, but on different positions in it. This is synchronous collaboration.

Google Drive even supports synchronous collaboration across different computers and devices, as shown in figure 5.4.1 and 5.9. This makes Google Drive support the Activity Sharing concept (See 2.5).

Google Drive is dependant on Internet connectivity to work correctly. If the connection is broken, you can not edit the documents until the network comes up again. Google has come up with a partial solution to this - making Docs, Sheets, Slides and Drawings available offline. This feature requires Google Chrome (See 6.3). It works by downloading both your files and the application to your local harddrive. When working offline, your documents are synchronized to the local harddrive instead of Google's servers, and when you have an Internet connection again, your modified files will be automatically synchronized with Google Drive.

5.4.2 Microsoft Office 365

Office 365 [59] is a service which combines Microsoft Office applications and Office Web Apps [60]. Office Web Apps is Microsoft's answer to Google Drive. It is an office suite with online versions of many applications from Microsoft Office, like Word, Excel, PowerPoint and OneNote.

Office 365 supports suspending and resuming (See 2.2). You can write a document in Office Web Apps, close the browser, and then resume your work by opening the document again.

Integration with SkyDrive (See 5.5.2) makes your documents and files available everywhere. If you are not using your own computer, and still want to edit your documents in Microsoft Office, then Office Web Apps is available through SkyDrive. This resembles the Activity Roaming concept (See 2.3).

As an additional feature, Office 365 is available on computers running Windows through application streaming. This feature is called Office on Demand, but is only available to customers with a premium subscription. Office on Demand is not installed on the computer, but it works just like the normal Office package. This is practical for users that need to work with Office, but does not use his own computer.

Documents in Office 365 can be shared to other users. Each person will always have the latest version of the document available through his SkyDrive folder.

With Office Web Apps, users can collaborate on the same document simultaneously. They can write on the same sentence if they want. Every symbol a user writes, is sent to all other participants. This works like synchronous sharing in the Activity Sharing concept (See 2.5).

Office Web Apps are built with focus on a touch-friendly user interface, which makes it ideal for touch screen computers or tablets. Office 365 is also available on most large mobile platforms, including Windows Phone, Android and iPhone. This is available through Mobile Viewers, which are touch-friendly web applications viewed in the device's web browser. Activity Adaptation (See 2.4) can be compared to this.

5.4.3 Comparison of office suites

Table 5.4 shows which concepts exists in both Google Drive and Office 365.

ABC concepts	Google Drive	Office 365
Activity-Centered	.	.
Activity Suspend and Resume	✓	✓
Activity Roaming	✓	✓
Activity Adaptation	✓	✓
Activity Sharing	✓	✓
Activity Awareness	.	.

Table 5.4: ABC concepts in office suites

5.5 File synchronization services

To physically carry documents and files around is becoming closer and closer to a part of history. This can be thanks to file synchronization services, which have gained popularity during the last years. The main purpose of a file synchronization service is to keep files across several computers in sync. To do this, most services provides an online storage, which also makes them cloud storage services. The bonus of this is that the files are available everywhere, and also have the possibility to be shared with others. There are many file synchronization service providers, but Dropbox is possibly the most known.

A problem with file syncronization services is when the computer is offline. If the user modifies the content in a file or folder, then it can not be syncronized immediately. Possible solutions to this is either that the local client keeps track of changes since last sync, or when online again, do a comparison between the online content and offline content. Then it sends just the modified data.

5.5.1 Dropbox

Dropbox [30] is a very popular file syncronization service and comes both as a native client for many devices and operating systems, and as a web service you can use through your web browser. Because of the great platform independence, Dropbox is a good choice for keeping files in sync. This makes it easy to reach your files from everywhere and from any capable device, which is similar to ABC's Activity Roaming concept (See 2.3) and Activity Adaptation (See 2.4).

Dropbox also has a built-in support for file sharing. A file or folder can either be shared between a group of people, or made publicly available for anyone. Dropbox is often used in collaboration projects, where several people work on the same files. ABC's Activity Sharing concept (See 2.5) can be compared to this.

5.5.2 Microsoft SkyDrive

Microsoft SkyDrive [58] is a cloud storage service. It has a desktop app for Windows and Mac OS X, and a mobile client for Windows Phone, iOS and Android. Microsoft also provides a web portal for SkyDrive. This makes SkyDrive, like Dropbox, a quite platform-independent service, and thus resembles the Activity Adaptation concept (See 2.4).

Because SkyDrive is a cloud service, it is basically built on the principle of being available everywhere. As long as you have an internet connection, you can connect to SkyDrive either through the web portal or any native application for mobile devices. This makes it similar to Activity Roaming (See 2.3).

With SkyDrive, you can share your documents and files easily through email, posting to social network services (like Facebook, LinkedIn, etc.) or through a link. SkyDrive supports all files, when sharing documents (.doc, .xls, .ppt) with others, they can use the built-in Office Web Apps (See 5.4.2) to view and edit the documents directly in the browser. Multiple people can edit the same file simultaneously, and every change will be shared synchronously. This is the same principle as Activity Sharing (See 2.5).

SkyDrive also has, in a way, some similarities with the Activity Awareness concept (See 2.6). The main usage of documents are to view, edit or print them. Based on the document type, SkyDrive lets the user do this directly in the browser with the help from Office Web Apps. Other files are just uploaded and downloaded.

5.5.3 Comparison of file synchronization services

Table 5.5 compares Dropbox and Microsoft SkyDrive with the ABC concepts.

ABC concepts	Dropbox	SkyDrive
Activity-Centered	.	.
Activity Suspend and Resume	.	.
Activity Roaming	✓	✓
Activity Adaptation	✓	✓
Activity Sharing	✓	✓
Activity Awareness	.	(✓)

Table 5.5: ABC concepts in file synchronization services

5.6 Other products

5.6.1 Document Viewer - Evince

Evince [31] is the name of a popular document viewer for Linux. It is the pre-installed document viewer in Unity [52], which currently is the default desktop environment in Ubuntu [51]. It supports most document formats, including PDF, PostScript, etc. One of the features of this program is that is stores what documents has been previously opened. When a user opens the same file again, it automatically restores the position in the document from last time.

This can be compared to the Activity Suspend and Resume concept (See 2.2).

5.6.2 Online whiteboard - Twiddla

Twiddla [88] is an online whiteboard web application with focus on team collaboration. As with normal whiteboards, which are actually whiteboards standing in a room, multiple people are able to draw at it at the same time. Twiddla has simulated this by sharing the view between every participant. This is done live and syncronously and resembles the Activity Sharing concept (See 2.5). There is also support for voice communication between the participants. Other features includes writing text and open documents (like Google Drive, see 5.4.1), as well as browse a web page or open an image - all inside Twiddla. The view is still shared and synchronized between the participants and they have the ability to draw and sketch over it.

Because Twiddla is a web application, it can be used from everywhere. This is similar to Activity Roaming (See 2.3).

5.6.3 Comparison of other products

Table 5.6 shows the different ABC concepts in other products.

ABC concepts	Evince	Twiddla
Activity-Centered	.	.
Activity Suspend and Resume	✓	.
Activity Roaming	.	✓
Activity Adaptation	.	.
Activity Sharing	.	✓
Activity Awareness	.	.

Table 5.6: ABC concepts in other products

Chapter 6

ABC concepts in web browsers

This chapter will compare the largest web browsers with the Activity-Based Computing concepts. Firefox (6.1), Opera (6.2), Chrome (6.3), Safari (6.4) and Internet Explorer (6.5).

Today, many people use several computers and devices for surfing the web. This introduces a challenge in having to remember every bookmark, passwords, open tabs, etc. across multiple devices. To solve this problem, most large web browsers have implemented a system for synchronizing these elements between devices [72].

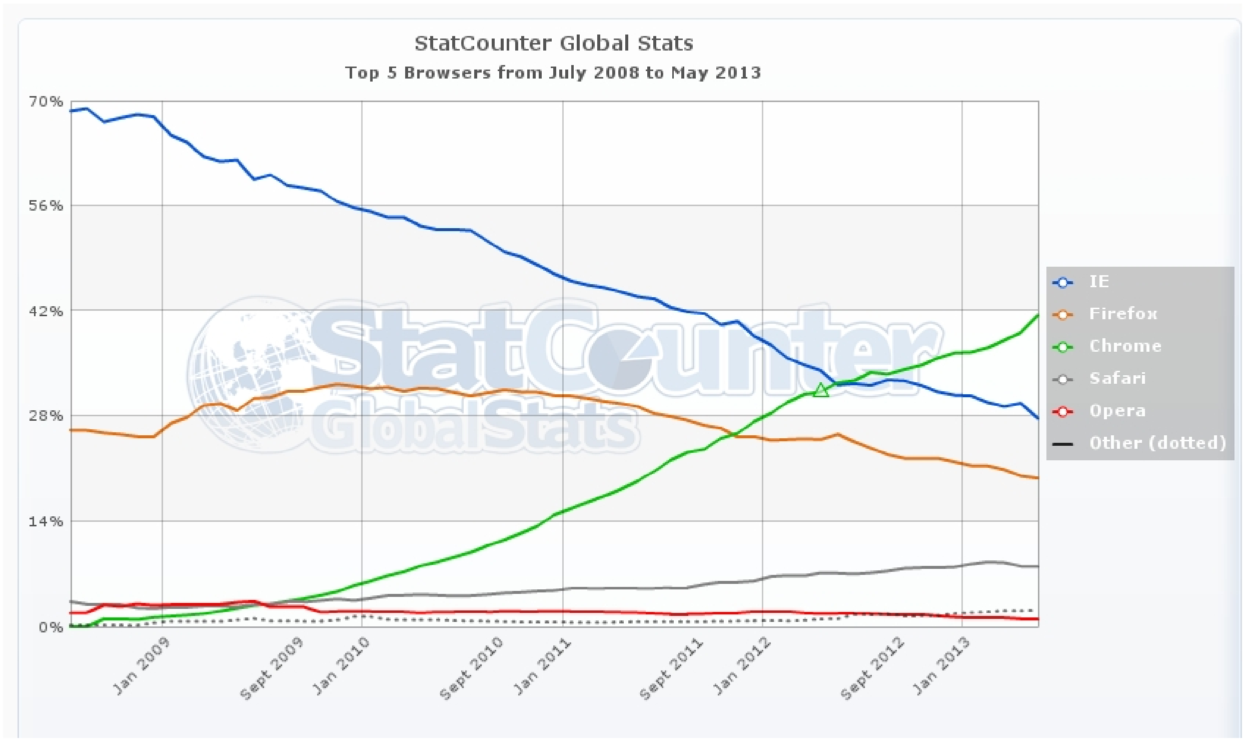


Figure 6.1: StatCounter showing web browser usage 2008-2013

Source: [82]

Figure 6.1 shows the usage statistic for the five web browsers. StatCounter base this statistics on data collected from over 3 million web sites with a total of around 15 billion pageviews per month [81]. This gives an overview of which browser is the most popular and used the most, as well as the different trends in web browsers usage.

6.1 Firefox

Mozilla Firefox [66] is a popular web browser available on most large operating systems. It is open source and supports plugins. The great number of plugins makes Firefox very versatile and gives the user the possibility to add and customize most functions.

As most modern web browsers, Firefox supports browsing multiple sites at the same time. Each web site opens up in a new tab inside the current window. If one of these sites are something you use all the time, they can be pinned. This means that they will not close, and are always shown in the tab menu. Tabs and windows can be used to have different tasks open at the same time. A user can separate all web sites relevant to a specific task as tabs in one window, and other web sites relevant to other tasks in another window.

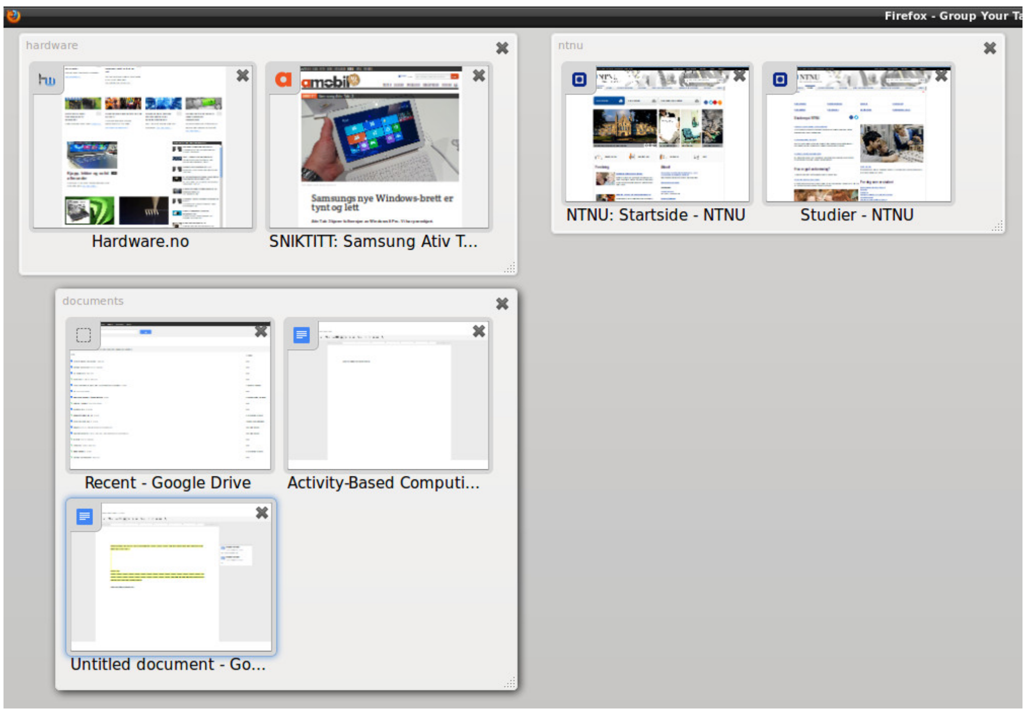


Figure 6.2: Firefox tab groups

Firefox supports grouping of tabs, which means that a user can group related tabs according to what task they belongs to. Each group can be assigned with a name, and thus makes it easy to know what the context of a group is. With the tab groups view in Firefox (See figure 6.2), you get an overview of all current groups and a thumbnail of each tab inside the group. Tabs can be moved simply by drag and drop between the groups. When selecting a group, all other tabs are hidden. But with a simple click on the tab groups button (See figure 6.3) or by pressing *Ctrl + Shift + E*, all groups are shown. This makes switching between tasks simple and quick, and is similar to the Activity-Centered concept (See 2.1).

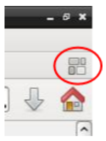


Figure 6.3: Firefox tab groups button

Firefox also features something they call the Awesome Bar. This replaces the standard location bar and can be used for both navigating by typing in an URL and searching by typing in search keywords. The Awesome Bar use an auto-complete algorithm which is based on a user's browsing history, most visited web pages, recently visited web pages, search suggestions from a search engine (e.g. Google, which is default) and open tabs. This gives some quite accurate suggestions for the user in a scroll-down list. Another feature is that Firefox, with permission from the user, can send location data to web sites. This will help to find more relevant results from searches. All this is similar to the Activity Awareness concept (See 2.6).

The default behavior when starting Firefox is to begin a new session with a single tab showing the start page. This can be changed to resume last session. However, if Firefox crashed last time it was run, it suggest resuming the last session. Firefox does not include a way to save multiple sessions across runtimes, but with help from plugins, this functionality can be added. When Firefox is configured to restore last session, the different tab groups are also restored. The Activity Suspend and Resume concept (See 2.2) can be compared to this.

6.1.1 Firefox on mobile devices

Firefox can be used on mobile devices, but only if they are running Android. It is named Firefox for Android [64]. As with the desktop version, Firefox for Android supports addons to enrich the functionality or browsing experience. Although the mobile version of Firefox is limited to Android, it covers many different devices nonetheless. In addition, the desktop version is platform independent. This makes Firefox still comparable to the Activity Adaptation concept (See 2.4).

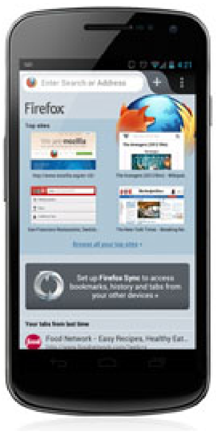

Enter Search or Address
Firefox
Top sites
Set up Firefox Sync to access bookmarks, history and tabs from your other devices »
Your tabs from last time
Food Network - Easy Recipes, Healthy Eat...

Figure 6.4: Firefox for Android

Source: [1]

6.1.2 Firefox Sync

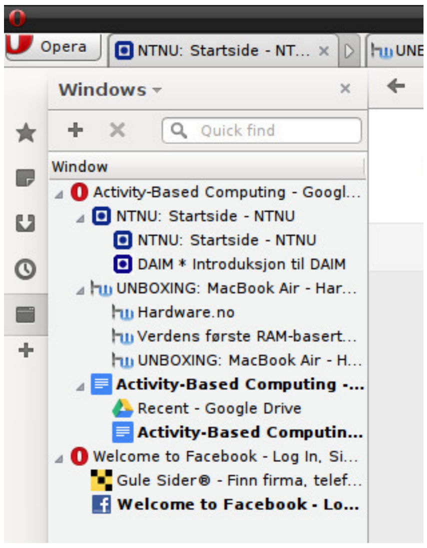
Firefox Sync [65] is used to synchronize browser information across several computers and devices. A user's bookmarks, passwords, history, addons, preferences and open tabs can be reachable from any Firefox browser - either a desktop browser or mobile browser. Even tab groups can be synchronized, meaning that whole ”activities” can be reached from anywhere. The data is stored on Mozilla's servers, and is automatically synchronized to a web browser when a user logs in with Firefox Sync. This makes Firefox similar to the Activity Roaming concept (See 2.3).

6.2 Opera

Opera [10] is a web browser from the Scandinavian company with the same name. It is available on most operating systems, including Windows, Mac OS X, Linux, which makes it quite versatile. However, Opera is more than just a web browser. It incorporates an email-client for reading and writing emails, IRC-client for chatting, BitTorrent-client for downloading files, RSS-client for reading web feeds, etc.

Opera supports tabbed browsing. This means that you can have several web pages open simultaneously in the same browser window. In addition to web pages, the different ”applications” in Opera also opens as new tabs.

This way you can have all web pages, emails and chat associated to a task in a single window of Opera. You can open a new window for different tasks, and have separate chat groups, emails, etc. Then each window represents an activity, with all information related to it inside in different tabs. Opera supports a feature called following tab. This is similar to Firefox' tab groups and enables the user to group related tabs together. Figure 6.5 shows how Opera organize tabs, following tabs and windows in a tree-view. This way Opera can be used in an Activity-Centered (See 2.1) way.



Activity-Based Computing - Googl...
NTNU: Startside - NTNU
NTNU: Startside - NTNU
DAIM * Introduksjon til DAIM
UNBOXING: MacBook Air - Har...
Hardware.no
Verdens første RAM-basert...
UNBOXING: MacBook Air - H...
Activity-Based Computing -...
Recent - Google Drive
Activity-Based Computin...
Welcome to Facebook - Log In, Si...
Gule Sider® - Finn firma, telef...
Welcome to Facebook - Lo...

Figure 6.5: Opera tabs and windows overview

Another feature in Opera is the ability to save and restore sessions. A session contains information on open tabs and can be assigned a name. The user could use sessions to divide web tasks into several activities and name them accordingly. By default, Opera also saves the last used session. This means that whenever a user starts Opera, he will be presented with the same tabs open as the last time. This compares to the Activity Suspend and Resume concept (See 2.2).

6.2.1 Opera on mobile devices

Opera is also available on mobile devices, both as Opera Mobile [13] and Opera Mini [12].

Opera Mini is a lightweight browser for most mobile phones. Opera uses their servers to both compress and adapt web sites before sending them to the user. Figure 6.7 shows how the *ntnu.no* web site is displayed on Opera

Mini after the layout has been adapted for the small screen. The original web site can be seen in the desktop version of Opera in figure 6.6. By compressing web pages on Opera's servers first, it will reduce the data needed to display a web page. This will both speed up web browsing when using slower network connections, as well as reducing costs if you pay per megabyte downloaded. Because Opera Mini uses servers for a lot of the work, it can be run on most devices, even if they are restricted on avaiable resources - like slower and older mobile phones.

Opera Mobile is a bit closer to a regular browser. It does not use external servers, so most processing is executed on the device itself. On the other hand, Opera Mobile is not dependant on Operas servers to function correctly, but it still has the possibility to use them for compressing pages.

With support for both many operating systems on desktop computers, as well as mobile devices with Mobile and Mini, Opera is very adaptable to almost any device you use. This resembles the Activity Adaptation concept (See 2.4).

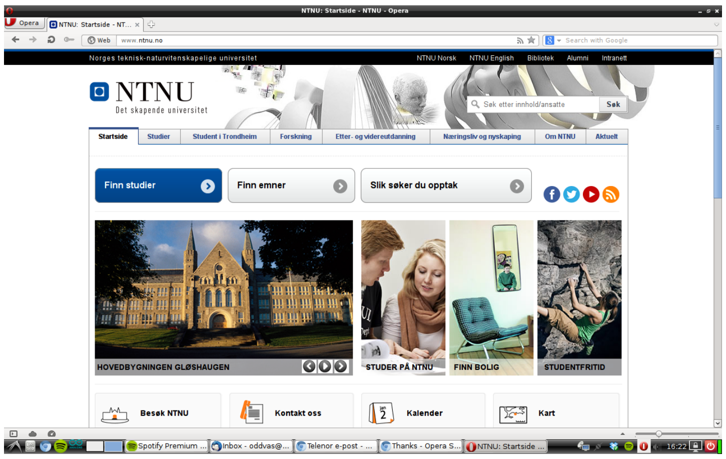


Figure 6.6: Opera desktop browser

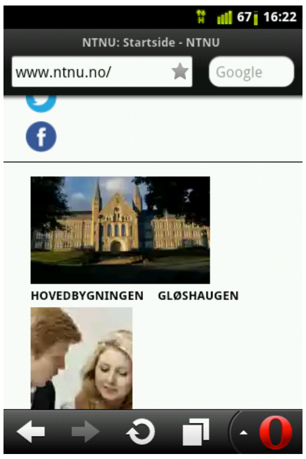


Figure 6.7: Opera Mini

6.2.2 Opera Link

With an Opera browser on so many devices, Opera has come up with a way of synchronizing bookmarks, passwords, history, etc. It is called Opera Link [11]. Opera Link helps users to synchronize browser information between all computers and devices they use. This infomation is stored securely on Opera's servers, and is available when you sign in to Opera Link. When you are browsing, information is always in sync. You can either choose everything, or synchronize only certain parts - like bookmarks.

Opera Link connects all Opera software on a user's computers and devices. This makes Opera support a feature similar to the Activity Roaming concept (See 2.3), although not completely. A user can not resume his stored sessions across multiple devices, unless using alternative options with third party software (e.g. Dropbox for synchronizing all configuration files to several computers).

6.3 Chrome

Google Chrome [40] is a web browser based on the open source browser Chromium [37], made by Google. It has focus on speed, security and simplicity, but also functionality and customization through apps and extensions.

As both Firefox and Opera, Chrome is also supporting tabs and multiple windows. A difference in Chrome is the use of multiple processes. This means that if a tab freezes, the whole browser does not need to be restarted. Only the affected tab. With tabs and windows it is possible to surf the web in an Activity-Centered way (See 2.1).

Chrome also supports some of the Suspend and Resume concept (See 2.2). Through a setting in Chrome, a user can configure whether to resume the last session or start a fresh one each time Chrome is started. It does not support saving multiple sessions, which would be practical if we are looking from an Activity-Centered perspective. However, this can be added through an extension. If Chrome is not closed properly, it will assume that it was unexpected and automatically resumes the last session.

Google Chrome has only one text input-bar on top. This is a combined location and search bar. Chrome automatically identifies a user's keywords and will try to autocomplete with web pages in the browsing history, web sites with similar address and search suggestions from Google's search engine. Your location, based on your IP-address, can also be sent with your search, giving you more local search results. These are probably more relevant. Along with Single Sign-On (more in Chrome sign in), this makes Chrome support functionality that resembles the Activity Awareness concept (See 2.6).

6.3.1 Chrome on mobile devices

Chrome is available on many devices, both on Android and iPhone, and goes under the name of Chrome Mobile [35]. The mobile version does also support unlimited tabs, synchronizing of devices (more in Chrome sign in), automatic sign in to Google services and autocomplete suggestions. Chrome has also implemented a function call Send to mobile. A user can send pages to a mobile phone with a single click. These pages are automatically sent to the device for immediate or later use. They can even be read offline when a person is on the go. All this compares to the Activity-Roaming concept (See 2.3).

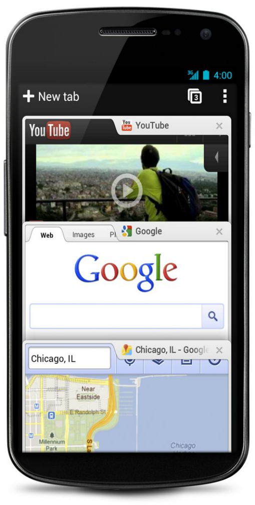


Figure 6.8: Chrome on Android

Source: [79]

Figure 6.8 shows Google Chrome on an Android smartphone. Chrome is able to display several tabs on top of each other.

6.3.2 Chrome sign in

Google's answer to Firefox Sync and Opera Link is to simply signing in to Chrome. Signing in will bring all of a user's bookmarks, passwords, apps, browsing history, configurations and even open tabs to the current Chrome instance. A user can install an app in Chrome on one computer, and it will instantly be available on other computers. All this data is stored securely on Google's servers. This makes Google Chrome comparable with the Activity Roaming concept (See 2.3).

Another bonus with signing in to Chrome, is that it automatically send login information to all of Google's online services. Signing in to Chrome works like a Single Sign-On (SSO) for Google services. It is enough to enter your username and password in one place, and Chrome anticipates that the user stays the same throughout the session. This results in a smoother browsing experience when using several Google services.

Chrome also supports multiple user accounts on the same computer. This way several users can use the same browser, with all their personal configurations at hand with a single sign in. When there are more than one person sharing a computer, this function is quite practical.

6.4 Safari

Apple Safari [8] is the default web browser on Mac OS X [7]. It provides a slim user interface with almost any web browsing functionality you need. If there is something missing, it is easy to add by installing an Extension.

Safari supports tabbed browsing and multiple windows. Tabs can be dragged around to be rearranged, or dragged out to open a new window. You can use mouse gestures to zoom out of a web page, and thus entering the tab panorama view. The panorama view gives an overview of currently open tabs, with their associated thumbnail images. This makes switching between tabs easy and straightforward. Tabs and windows makes Safari similar to the Activity-Centered concept (See 2.1).

If Safari crashes or the computer is shut down without closing Safari, the previous browsing session can be restored. It is also possible to configure Safari to always open the last session when starting, but only the last, there is no possibility to have several sessions stored. This makes Safari partially support the Activity Suspend and Resume concept (See 2.2).

Safari is also partially context aware, which is the sixt ABC concept (See 2.6). If you are reading an article, Safari knows it, and suggest that you use the Safari Reader. Safari Reader provides a clean view of articles. If the article you are reading is spanned over several pages, Safari Reader retrieves these pages and the entire article is displayed in a continous view.

Safari has an integrated feature which enables the user to share sites through various channels, like email, Twitter, Facebook, etc. This does, however only share links to others, and is not suited for shared view or collaboration, unless it is a link to a web service which supports live collaboration. It is not valid as featuring the Sharing and collaboration concept (See 2.5).

6.4.1 Safari on mobile devices

Apple has made a mobile version of Safari. This is the default web browser on iOS [4] which is used on the iPhone (See figure **??**), iPad and iPod Touch. Safari on iOS shares many features with Safari on OS X. Web pages are displayed as they should, and with the small screens of the iPhone and the iPod Touch, the ability to zoom is a key feature.

Safari adapts nicely to mobile devices, both regarding screen size, user interface, slower processors, etc. However, it does not support other devices than Apple's own running on iOS. This makes Safari partially include the Activity Adaptation concept (See 2.4).

Figure 6.9: Safari on iOS

Source: [5]

6.4.2 iCloud

Safari is tightly integrated with Apple's cloud service, iCloud [3]. iCloud synchronizes settings, bookmarks and the Reading List between Safari browsers across iCloud-enabled devices. A user can save whole web pages in the Reading List for reading later when he is offline.

With iCloud comes also a feature called iCloud Tabs (See figure 6.10). When a user opens a tab on his Mac laptop, iPhone, iPad or any iCloud-enabled device with Safari, it gets stored in iCloud. This means that the user can pick up his web browsing from any Apple device.

Another feature is the iCloud Keychain. This synchronizes all usernames and password across your devices and comes in handy when you are using your mobile browser and have long passwords to both type in and remember.

This resembles the Activity Roaming concept (See 2.3).

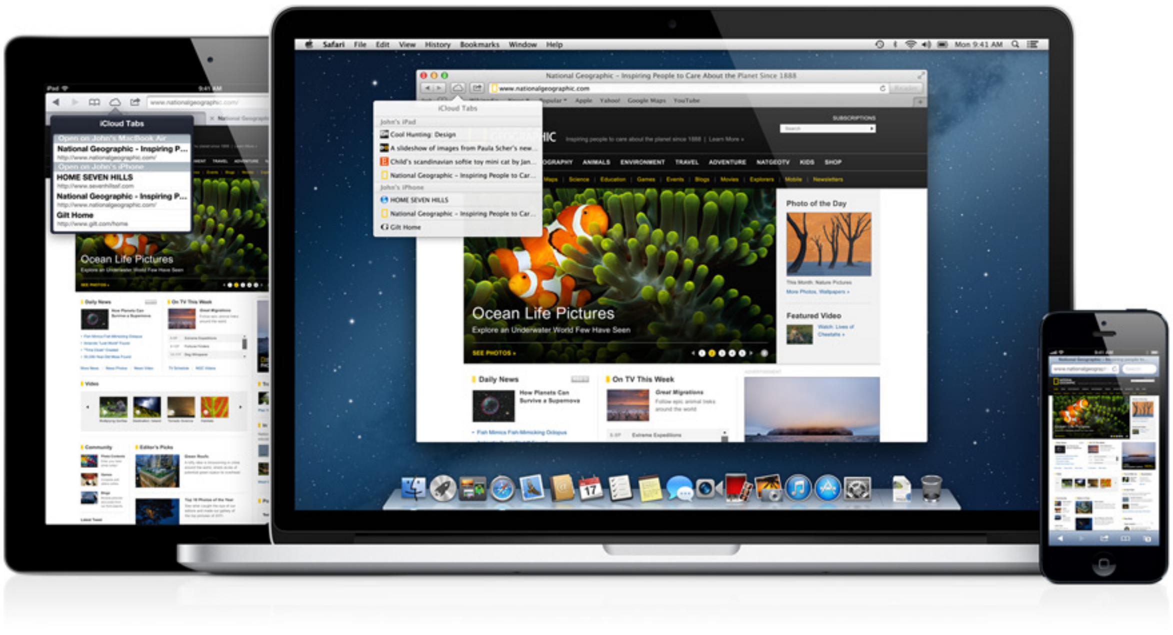


Figure 6.10: iCloud Tabs

Source: [8]

6.5 Internet Explorer

Microsoft Internet Explorer (IE) [56] is a web browser made for the Windows operating system. IE has been the most used web browser for years, until last year, when Google Chrome bypassed it (See figure 6.1). The current version is 10, and requires Windows 7 or later to run. IE 10 comes in two flavors, a desktop version for Windows 7/8 and as a Windows Store app. When using Windows 8 and its touch user interface, Internet Explorer is focusing on being more touch-friendly, i.e. fullscreen view, larger buttons, swipe gestures, etc. However, both the touch version and desktop version is the same browser, and shares configurations and settings.

The touch version supports only one window at a time, but it supports multiple tabs (See figure 6.11). These tabs are hidden until the user swipes down from the top of the screen. Tabs can be pinned, which means that they are always open. The desktop version, however, supports multiple windows and tabs, and thus can be used in an Activity-Centered way (See 2.1).

Internet Explorer supports only resuming of the last session, which can be opened by a simple click in the new tab view. This is only supported in the desktop version of IE, not the touch-friendly version. If the computer crashes, or you shut it down without closing Internet Explorer, you can reopen all windows and tabs if you want. This makes IE a little similar to the Activity Suspend and Resume concept (See 2.2).

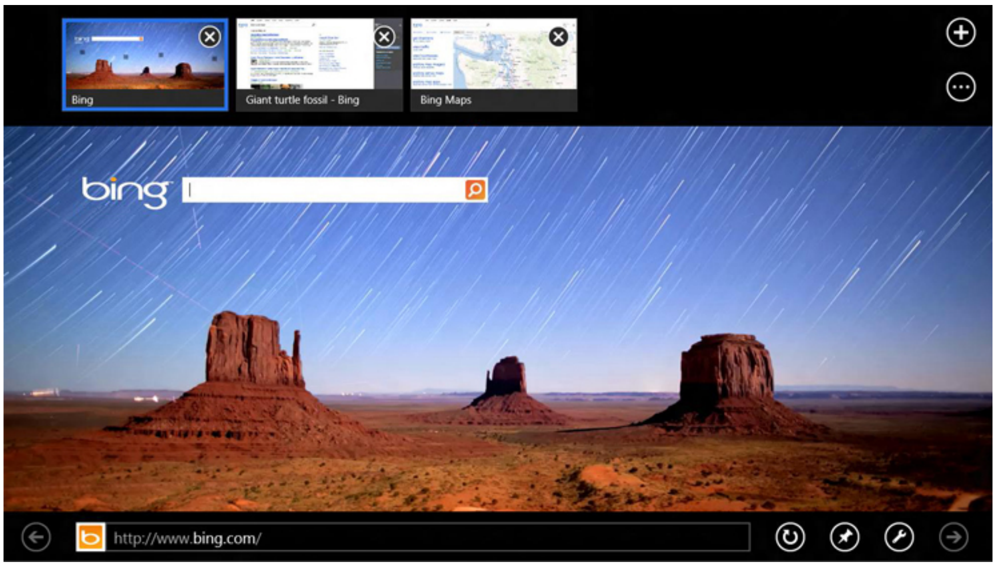


Figure 6.11: Tabs in the touch-friendly Internet Explorer

Source: [56]

6.5.1 Internet Explorer on other devices

Internet Explorer is available on tablets using Windows 8. Tablets use the same touch-friendly version as normal computers, and thus share both look and feel and web browsing engine. This gives the user a smooth transition when going from the computer to a tablet. However, both tablets and computers running the touch-friendly interface does not support multiple windows. This reduces the possibilities for an activity-centered experience.

Internet Explorer Mobile is the smaller version made for Windows Phone. It is also based on the same web browsing engine as the regular Internet Explorer 10, but its user interface is more minimalistic (See figure 6.12). Favorites can be synchronized with the mobile version through SkyDrive.

Figure 6.12: Internet Explorer Mobile

Source: [87]

Microsoft has also made Internet Explorer available on their gaming console, the Xbox. This is only available if the user subscribes to the Xbox Live Gold services. With the help of Xbox SmartGlass, it can also be remote controlled from various devices, including smartphones running Windows Phone or Android and tablets running Windows 8.

Internet Explorer adapts good to the supported devices, but this is limited to Windows 8 (desktop and tablets), Windows Phone and Xbox. This makes it partially similar to the Activity Adaptation concept (See 2.4).

6.5.2 Synchronizing devices

Unless you are using Windows 8 or other third party software, then Internet Explorer does not have a way of synchronizing settings, bookmarks, passwords, etc. between computers or devices. However, with Windows 8, this feature is integrated with SkyDrive (See 5.5.2). This enables users to have their configurations and favorites synchronized across different computers as long as the user signs in with a Microsoft account (formerly Windows Live ID). Compared to the other browsers, this functionality is somewhat limited. The concept of Activity Roaming (See 2.3) is partially existing in Internet Explorer.

6.6 Comparison of the web browsers

Web browsers incorporates many of the Activity-Based Computing concepts, but to a limited extent - browsing the web. Table 6.1 compares the five web browsers to the ABC concepts.

ABC concepts	Firefox	Opera	Chrome	Safari	Internet Explorer
Activity-Centered	✓	✓	(✓)	(✓)	(✓)
Activity Suspend and Resume	✓	✓	(✓)	(✓)	(✓)
Activity Roaming	✓	(✓)	✓	✓	(✓)
Activity Adaptation	(✓)	✓	✓	(✓)	(✓)
Activity Sharing	.	.	.	.	.
Activity Awareness	(✓)	.	(✓)	(✓)	.

Table 6.1: ABC concepts in web browsers

Chapter 7

ABC and the cloud

This chapter outlines possible solutions for integrating Activity-Based Computing concepts in web browsers (7.1).

7.1 Web browsers

7.1.1 Activity-Centered

The first, and most fundamental concept of Activity-Based Computing is Activity-Centered (See 2.1). To completely integrate an activity-centered way of working with web browsers, there are several points that needs to be implemented.

Windows and tabs

Most desktop web browsers supports having multiple windows open, each with multiple tabs inside. By building further on this principle, we can make web browsers activity-centered. Windows can represent activities. Every related web page or -application within a task, are gathered inside one window as tabs (See figure 7.1). This relies on the operating system's window manager to switch between activities.

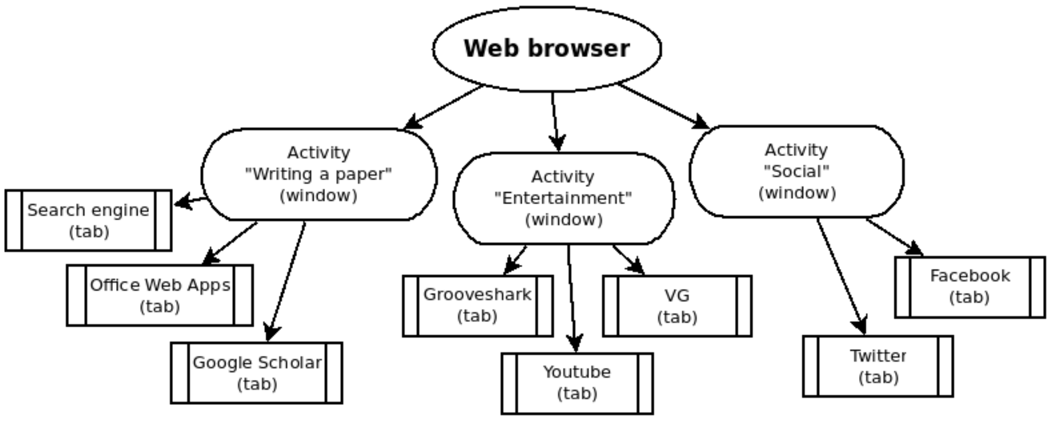


Figure 7.1: Overall design for activity-centered web browsers

Grouping tabs into activities

Instead of using web sites as base units, web browsers can implement the activity as the base unit, which is basically a group of tabs. When a user launches the web browser, or opens a new window, he should be presented with a choice of which activity the new window would represent. This view could be similar to the Speed Dial in Opera (See figure 7.2), but instead of web sites, it is showing activities.

Creation of activities

If the user does not find an existing activity that fits the task, he can just create a new one. This can be done similarly to adding a new web site to Speed Dial in Opera, or just open a new browser window and tabs as you normally would do.

If an activity is already open, the user can choose between opening the already open activity, or create a duplicate of it. This way, it is both fast and easy to create activities that are similar, but not equal. E.g. one activity can be *"Writing a master thesis"* while another is *"Writing a scientific paper"*.

Activity names

Each activity has its own name, so that similar activities can be separated. When the user opens a new window, he can choose between just browsing, or give the window a name. If he gives the window a name, it will automatically be stored as an activity. This implies that the browsers must have the ability to display activity names - or simply give browser windows individual names. Names helps users to identify the correct activity.

If the user has a large amount of activities, it should be possible to search though activities based on names, but also on the associated tabs, web applications and URL-addresses. This makes it easier to find the correct activity, even if the user does not remember the name, but do remember some of the web sites he used in it.

Multiple tabs visible simultaneously

Many tasks reqires multiple tabs open simultaneously. If you have a large screen and high resolution, web pages and -applications rarely use the entire display area, especially the horizontal width. Instead of having to separate a tab into a new window, or constantly switching between tabs, it should be possible to display them side-by-side, or even as windows inside the web browser window. This way, the user does not need to create multiple activities for a single task.

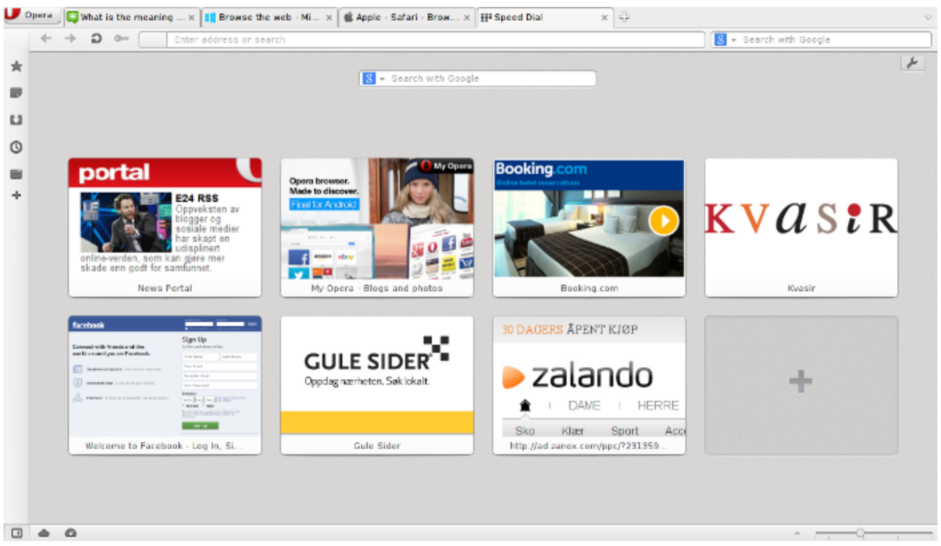


Figure 7.2: Opera Speed Dial

7.1.2 Suspend and resume

The next concept is Activity Suspend and Resume (See 2.2). Many browsers already incorporates session resume, but this is limited to the last, and usually only, session used. There are several steps to enable suspend and resume in web browsers.

Persistent activities

An activity has to be persistent, i.e. the web browser has to store the content of an activity, including its name and the different tabs associated with it. This way, the activities can be obtained after the browser has been closed or the computer has been rebooted.

Activity state

The state of an activity does also have to be stored, i.e. which tab had focus, the order of tabs, the position in the web pages, browsing history, etc. In case of multiple tabs visible, either side-by-side or as windows, the position of these tabs relative to each others must also be stored.

Automatic log in to services

Most web applications and -sites requires a user to log in to operate functionally. These sites usually have an inactivity timer which automatically logs out the user after a certain amount of time. When the user tries to use the service later on, he is required to log in again. Most web browsers incorporates a username/password storage and provides the user with these credentials pre-filled in. When resuming an activity, the web browser must automatically sign in the user to the associated web applications and -sites. For a true activity resume, the browser does this in the background and makes the resume process seamless and smooth for the user. If the user was browsing a specific site inside a web application, he should be returned to it as if he never left.

7.1.3 Roaming

Another important concept of Activity-Based Computing is Activity Roaming (See 2.3). This means that the user should be able to access his activities from anywhere.

Store activities in the cloud

For Activity-Based Computing to be implemented in web browsers, we need a place to store activity information. This information has to be accessible everywhere. A good choice would be to utilize cloud computing technology. Activity Roaming is very dependant of a stable service and cloud computing is quite reliable due to its highly virtualized base infrastructure (See 4.1.3).

Register every state change

The web browser must register and send all information about an activity's state to the cloud service. If a person rearranges the order of the tabs, this information must be sent immediately, making these modifications available on other devices almost instantly and the roaming experience more smoothly.

7.1.4 Adaptation

Activity Adaptation (See 2.4) is directed at the ability to use activities on many different devices, with different resources available. Most web browsers have a mobile version, with its user interface adapted to fit smaller screens, use touch input, etc. To support this concept, the mobile web browsers must be able to do most of the tasks that the desktop web browsers can do. However, for most web applications, this is something that has to be done by the application provider. Web browsers can only display the web page.

Different screen resolutions

When switching between computers with different screen resolutions, it is up to the web applications to resize themselves accordingly. However, with multiple tabs visible simultaneously, the web browser can store the layout in a relative scale with percentage instead of absolute coordinations for windows. This way, the activity will ”zoom” itself and fit the screen nicely. If the screen is very small, e.g. a netbook or tablet, the web browser can choose to show them as regular single tabs at a time.

Tab groups

Most mobile web browsers supports multiple tabs. However, they do not support multiple windows. As with desktop web browsers, the mobile web browsers must be able to group tabs together to form an activity. While not being able to have multiple windows open, mobile web browsers can implement an interface for switching between activities. This can be a simple menu with a list of activities. When selecting an activity, all open tabs are closed, and the new tabs are opened.

Apps

Many web applications and cloud computing services have made their own app for use on mobile devices. These apps are more tailored to the limited resources available and are better suited on smaller devices. If an activity contains a web application, which has its own app on the mobile platform the web browser is running on, the web browser could substitute the associated tab with the actual app. This depends on an ability to start an app inside the web browser, but would give a better experience for the user.

7.1.5 Sharing and collaboration

None of the web browsers supports the Sharing and collaboration concept (See 2.5). Collaboration has been up to the web applications to support, like Google Drive (See 5.4.1) and Twiddla (See 5.6.2). However, there are certain steps the web browser can do to enable sharing and collaboration.

Sharing the activity

Firstly, the activity must be shared. The person owning the activity gives other users access to it through a sharing menu in the web browser. Other users are identified by either email or username. When shared, the activity pops up in the activity list to the new user.

Permissions

Activities can have one owner, but an unlimited number of participants. These participants can either have full access to the activity, meaning that they can both view and edit it. The other option is that they can only have view permissions, which means that they can see the activity and all its associated tabs, but not able to browse or edit it. Useful for demonstrations and guides with many viewers and few actors.

Shared state

Each connected participant's web browser will both send and receive the state of an activity when it is shared. When sharing a state, only the changes are sent to other participants. If a user opens a new tab in a shared activity, the same tab opens up in every participant's web browser. Another user can click on a text input box in a web page, and the other participants will see that the text box is marked by the given user. Any text he inputs will also be sent to the others, and thus will fill the text box on their computers as well. This will enable participants to collaborate on web applications without being bound to view precicely the same. A person can work at the top of the page, while another user can see the bottom.

Shared view

Shared view is another method for collaborating. Instead of sending state changes, the browser send the entire view to other participants, similar to the remote desktop and -application technologies (See 5.2 and 5.3). This makes all participants have the exact same view throughout the activity.

Show participants

The web browser must display a list of all participants with their own color, so that each user knows who else is connected. Each user has his own color-coded mouse cursor, so that everyone sees where everyone is working. This is similar to the tele-pointers of the ABC framework (See 3.1). The same color is marked at the different tabs, showing which tab is in focus for each user. If the users are browsing in the same tab, a square indicating the view-area of each participants is drawn in the web page, which is also in the same color.

Chat rooms

Chat can be used for communication between the participants in an activity. Each activity works as a chat room, with only the participants in the active activity is member of, and thus hides all unrelevant chat regarding other activities.

Voice communication

The same goes for voice communication. Each activity is a separate channel, and will only send and receive voice chat from the participating users. The web browser needs to be able to handle microphone input and sound output. It must also be able to know which participants are using the current activity.

7.1.6 Context awareness

The last concept of Activity-Based Computing is Activity Awareness (See 2.6).

Location

The web browser is often able to track its position with the help of IP-addresses and an IP-address location service [34]. This enables the web browser to send this information to for instance search engines, to give more local search results.

Mobile devices does often have a GPS device integrated, and mobile phones can use the network cell information to provide a location. This gives a more precice location information which can be used in mobile web browsers.

Attached hardware

Web browsers can be enabled to read information from attached hardware, like sensors, RFID readers, etc. This can be used to suggest new activities or used as input to web applications. These sensors makes the web browser aware of the physical environment of the user.

7.1.7 Architecture

For an easy way to enable Activity-Based Computing in web browsers, a user's settings and activity states must be stored in the cloud. This approach relies on the client-server model. The cloud acts as a server, and the web browser is the client.

In addition to connecting to regular cloud-based web applications, the web browser also connects to an ABC-service in the cloud. This is necessary to keep activities synchronized across devices and users. All this happens automatically in the background, and is not something the user has to think about when working.

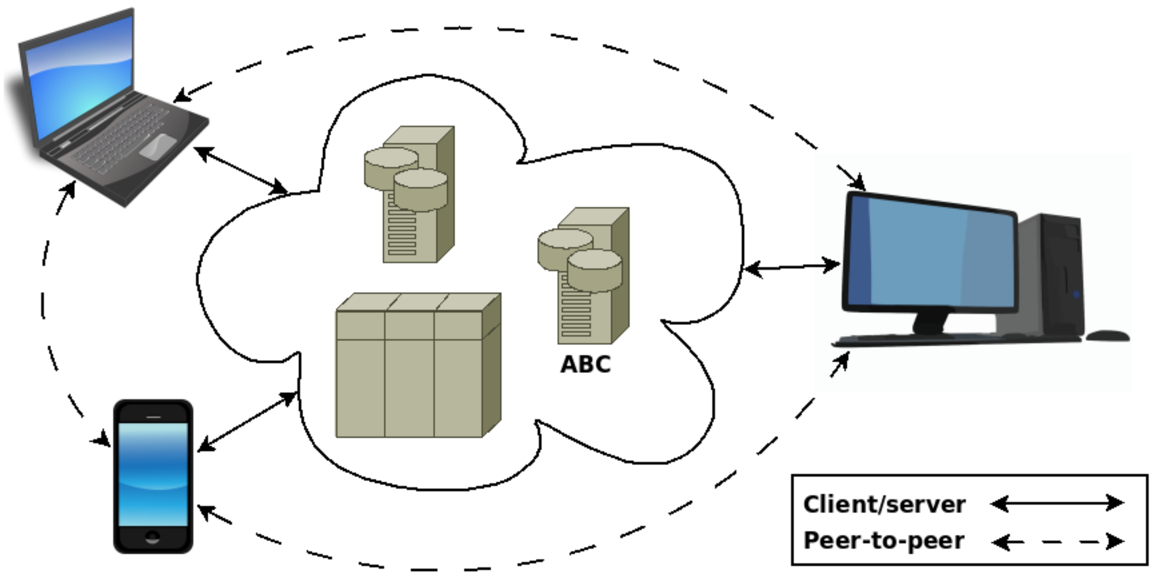


Figure 7.3: The architecture design for ABC in web browsers

For collaboration and synchronous sharing of states, the browsers can use peer-to-peer connections directly between the clients. The ABC framework already utilize peer-to-peer communication [24].

Different tasks can use different communication method. Low latency, non-critical tasks can use the peer-to-peer communication, like chat, voice communication, shared view, etc. While activity states, sessions and permanent information can use the client-server model.

7.1.8 Cross-browser compatibility

People tends to use several different web browsers during a day. It can be or instance Firefox on the computer, Opera on the smartphone and Chrome on the tablet. If people have different favorite web browsers depending on the devices, and still wants to enable Activity-Based Computing, the web browsers needs to communicate together.

Common protocol

One way is to develop a common shared protocol to handle Activity-Based Computing mechanisms. This protocol could be similar to the existing protocol used in the ABC framework - Activity-Based Computing Protocol (ABCP) [22, 24]. This protocol is based on a stateless HTTP-like protocol. It sends XML-based messages through GET and POST, and also PUBLISH and SUBSCRIBE methods for event-based communication.

The point of a common protocol is that the web browsers can implement each ABC concept individually and possibly different from each other, as long as the protocol remains the same. This ensures that the ABC concepts are kept regardless of which web browser a person uses.

Another bonus of this is that the concept of Activity Adaptation (See 2.4) becomes more obvious.

Plugins

Another way to communicate cross-browser is to use plugins. Most web browsers supports plugins. Plugins are small pieces of software that adds extra functionality in web browsers. The plugin acts as a layer between the web browser and the ABC service in the cloud (See figure 7.4).

A benefit with plugins is that they can be developed by a third party, and thus have no relation to any web browser developer. They are not dependant on a common protocol that web browsers can use, they can just use their own protocol and write plugins for most web browsers. This way, the different web browsers can communicate with each other through the plugins.

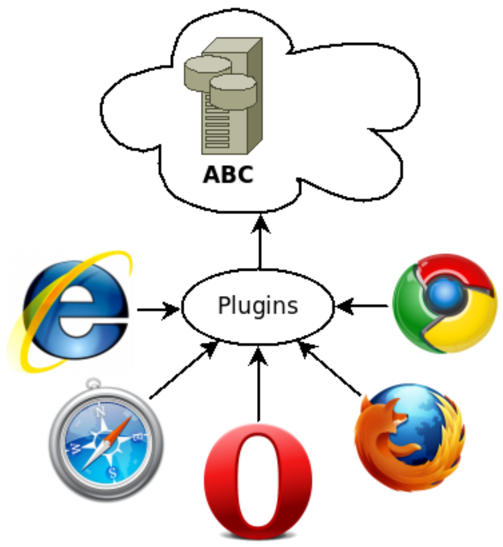


Figure 7.4: Plugins in web browsers to support ABC

Chapter 8

Discussion

This chapter will discuss the possible solutions and answer the previously stated research questions.

8.1 ABC concepts in current products

The first part of this thesis was to find ABC concepts in existing products and web browsers, hence the first research question:

RQ1: *To what extent do we find functionality similar to the concepts of Activity-Based Computing in current cloud computing services and web browsers?*

In short, the answer to this question and the summary of this product analysis is shown in table 8.1. The analysis of software products can be found in chapter 5 and web browsers in chapter 6.

Although several of the analyzed products are not a cloud computing service or a web browser, they are included in this product analysis because they illustrate some of the ABC concepts in a good way.

None of these products have any relationship with the ongoing research on Activity-Based Computing. Some of them are even older than the ABC research, like virtual desktops (See 5.1) and remote applications with X-forwarding (See 5.3.2). However, all these products incorporates features similar to one or several of the ABC concepts (See table 8.1).

	Product	Activity-Centered	Activity Suspend and Resume	Activity Roaming	Activity Adaptation	Activity Sharing	Activity Awareness
Virtual desktops	Workspaces	✓	(✓)	.	.	.	.
	Mission Control	✓	(✓)	.	.	.	.
Remote desktops	Microsoft Remote Desktop Services	(✓)	✓	✓	✓	.	.
	Windows Desktop Sharing	.	.	✓	.	✓	.
Remote applications	Microsoft RemoteApp	.	.	✓	(✓)	.	.
	X-forwarding	.	.	✓	(✓)	.	.
	X Persistent Remote Applications	(✓)	✓	✓	(✓)	.	.
Office suites	Google Drive	.	✓	✓	✓	✓	.
	Microsoft Office 365	.	✓	✓	✓	✓	.
File synchronization services	Dropbox	.	.	✓	✓	✓	.
	Microsoft SkyDrive	.	.	✓	✓	✓	(✓)
Other products	Evince	.	✓	.	.	.	.
	Twiddla	.	.	✓	.	✓	.
Web browsers	Firefox	✓	✓	✓	(✓)	.	(✓)
	Opera	✓	✓	(✓)	✓	.	.
	Chrome	(✓)	(✓)	✓	✓	.	(✓)
	Safari	(✓)	(✓)	✓	(✓)	.	(✓)
	Internet Explorer	(✓)	(✓)	(✓)	(✓)	.	.

Table 8.1: Summary of ABC concepts in current products and web browsers

8.1.1 Activity-Centered products

Few products implements the first ABC concept, which is Activity-Centered (See 2.1). This concept is mostly found in virtual desktops (See 5.1) and web browsers (See chapter 6). Although all of the web browsers have received a point for being activity-centered, most of them are only partially activity-centered. The same goes for Remote Desktop Services (See 5.2.1) and Xpra (See 5.3.3). The partial check mark comes from how the product is being used, not how it is designed. These products can be used in a way that gathers related programs and/or web applications to form an ”activity”, without the software knowing of the activity concept.

8.1.2 Suspend- and resumable products

Most products incorporates the ability to suspend and resume the ongoing work. A partial implementation enables the product to suspend and resume to a limited extent. With virtual desktops, this can be within a single log on session, and with web browsers it can be support for only a single session.

When completely implemented, the product supports a more persistent suspending and resuming, i.e. the user is able to resume the task exactly as he left it, regardless of the computer state. It survies a computer crash, reboot, etc.

8.1.3 Roaming products

Because of how the Internet has become an important aspect of working with computers today, and that almost any computer and device are connected to it, most services and products can be used from anywhere. Web applications are, by nature, available over the Internet, and thus supports the Roaming concept (See 2.3).

The only products, not supporting roaming, in this analysis are local desktop applications - Virtual desktops and the document reader, Evince (See 5.6.1).

The Roaming concept is the most common implemented concept in this product analysis.

8.1.4 Adaptable products

With the ever increasing use of mobile devices, both smartphones and tablets, software products must be adaptable to fit the different resources available. The main challenge is the different screen sizes. Most software products have its own mobile app for these devices. These apps are customized for use on smaller screens with touch interface.

The five web browsers described in chapter 6 have their own mobile version. A partial check mark indicates that the web browser is available on some mobile devices, but not all, e.g. Safari is only available on Apple's own iPhones and Internet Explorer is only available on Windows Phone. If the web browser is available on most large mobile operating system, it will get a full score, like Opera and Chrome.

The cloud services in this product analysis does also have their own mobile app, which works on most mobile platforms.

Remote dekstop and -applications does all get a partial mark. There are mobile applications which makes these products available on mobile devices. The applications in remote desktop and -application are usually full-sized desktop applications, and does not necessarily fit the small screens. They simply use zooming and panning to adapt to the smaller screens.

8.1.5 Sharable products

The Sharing and collaboration concept (See 2.5), is not widely incorporated into products. It is mostly available in the cloud service products. The reason for this is that it is less complicated to enable sharing in web applications and cloud computing services, compared to local applications, especially over the Internet. The single point-of-entry (e.g. an URL-address) makes these services easily available, and since web applications already runs on a server, the clients are simply connected to each other through it. If local applications should support sharing, the clients had to first find each other, and then connect to each other. This introduces several challenges, including network address translation and port forwarding for computers running on a network with only one external public IP-address, traffic relay through a server, etc. For this to be simple, a server with a known address had to be running to provide the clients with information about other clients, and the client connect to each others through the server. Bottom line is, a server makes sharing and collaboration easier, and cloud computing services already runs on a server.

The office suites (See 5.4) are a good example on products with live sharing and collaboration. Any number of clients can write at the same document simultaneously.

Although web applications with collaborative features runs in web browsers, the web browsers themselves does not include any collaboration feature.

8.1.6 Context-aware products

The last, and least implemented concept is the Activity Awareness (See 2.6). Context awareness is a very comprehensive subject. None of the products are fully context-aware.

Context awareness is mostly found in the web browsers. Although they are all partial, there are some functionality in web browsers that could resemble context awareness. The most similar is the geolocation feature found in HTML5. This reports the location of the web browser to the web sites, so they can be customized according to where the user is browsing the web from. This feature is useful when searching for local businesses, like restaurants.

8.2 Improvements for implementing ABC

The second part of this thesis was to come up with improvements for implementing ABC concepts in web browsers and cloud computing services.

RQ2: *How can current cloud computing services and web browsers be improved to support Activity-Based Computing?*

Web browsers are available on almost every computer and mobile device. This means that the user does not need to configure each computer for Activity-Based Computing, and thus makes web browsers a good base for implementing ABC.

To integrate ABC with cloud computing services would be more complex and still require a web browser. Either would the user be able to browse other web applications through a web application - like a web browser within a web browser, or the web application should handle each window and tabs in the web browser - which is basically the same as the web browser concept development (See 7.1). The difference is that instead of letting the web browser handle ABC, the cloud service handles it and controls the web browser through e.g. Javascript.

The improvements for integrating ABC in web browsers can be found in chapter 7.

8.2.1 Quick solutions

Based on my analysis of the different web browsers (See chapter 6), there are several possible steps towards a quick implementation of some of the ABC concepts.

Mozilla Firefox

Firefox (See 6.1) is the browser with the most similarities to Activity-Based Computing. With its tab groups, which symbolizes activities, Firefox can easily switch between a set of tabs. Each tab group can have a unique name. When restarting Firefox, the tab groups can be reopened again. Every tab group persists on the system through both computer reboot and browser crash. When using Firefox Sync, these tab groups are also available everywhere. Every open tab in Firefox on any computer can be opened on the current computer through Firefox Sync. This means that Firefox already supports functionality similar to the three first ABC concepts - Activity-Centered, Suspend/resume and Roaming.

By making Firefox available to other devices than those running Android, Firefox would also fit the Activity Adaptation concept.

Opera

Opera (See 6.2) supports saving sessions to the local harddrive. A session is a collection of tabs and windows. These sessions are not synchronized with Opera Link. However, for a single user/multiple devices scenario, it is possible to use cloud based file synchronization services, like Dropbox (See 5.5.1), to synchronize these local session files across several devices. This would improve the Roaming concept in Opera.

By using the sharing ability in Dropbox, these sessions can be used by multiple people. This will, however, only partially enable sharing of activities, because the sessions are not shared live. If a user opens a new tab, he must manually store the session, and other users must manually load the session again to get the new tab. This is a bit like asynchronous sharing of activities.

8.2.2 Web browsers

Many web browsers supports several of the ABC concepts already (See 6.6), but there are several possible improvements to fully integrate ABC.

Activity-Centered

The most fundamental and key-feature of Activity-Based Computing is that it is based on human activities. This means that the web browsers should handle activity objects, rather than just web sites and -applications. Some browsers have their own way of acting activity-centered, e.g. Firefox with its tab groups. However, the simplest general method would be to use the existing windows and tabs and let the operating system's window manager handle several activities as windows.

Suspend and resume

Most web browsers supports suspending and resuming of single sessions. The web browser must be able to store more than one activity. Activities need a permanent storage, so they can be used after computer reboots.

The biggest challenge with suspend and resume is logging in to services after long time of inactivity. The browser should be able to log in without interrupting the user or the activity. Ideally, this should happen in the background. However, there are security risks attached to this. For this to work smoothly, the web browser must be certain that the user who he says he is. Another aspect of this that the web application must implement a feature called *Background sign-in*, enabling the web browser to sign in without using the standard web page with username- and password input boxes. Alternatively, the browser opens up the root page of the web application in a hidden tab, in which it automatically fills in the user credentials, waiting for complete log in, closes the hidden tab, and then resumes the web application exactly how it was suspended.

Roaming

Roaming is enabled in many web browsers, with their own synchronization technique. However, with activities, both the activity information and the activity state must be synchronized between devices. The easiest solution is to store this as files locally, and then use a cloud computing service, for instance Dropbox (See 5.5.1), to keep these files synchronized across computers and devices. This is similar to the technique used in co-Activity-Manager (See 3.7). The web browser should constantly check the folder with these files and when a change occurs, it reloads the files and reconfigure itself accordingly. This way, activity states are synchronized instantly across the devices.

If the user gets disconnected, all activities and states are cached locally, similar to the ABC framework (See 3.2). However, this will also introduce another problem, which is that everything he does in the web browser is dependant on an Internet connection. Unless making web applications available offline, like Google Drive (See 5.4.1), this will stop all productivity.

Adaptation

Web browsers can be used on many computers and devices. When using web applications, it is often the web application's responsibility to adapt. With varying, and small screen sizes, mobile devices are the most challenging when it comes to adaptation. Mobile web browsers use zooming and panning when showing desktop web applications that are larger than the screen.

With no ability for multiple windows on mobile web browsers, the activity-centered concept from desktop versions with windows and tabs, can not be applied to mobile versions. Instead, they can just show one activity at a time, and have its own system for switching between activities.

With mobile apps available for most web applications, the idea of using apps inside the web browser instead of the web sites is just conceptual. This will just introduce several more challenges, like synchronize states between web application and app, the ability for an app to host another app, etc.

Sharing and collaborating

One thing all web browsers have in common, is the lack of sharing and collaboration abilities. Web browsers can use web applications which supports collaboration, but then it is on application level, not activity level.

When using shared state, the traffic between the different participants can be kept to a minimum. Shared state does only send information on what parts of the activity has changed. The actual change is computed locally. Compared to shared view, this gives a better viewing experience because of the reduced traffic. Another bonus with shared state is that it makes a more adaptable sharing for those with different size screens.

With shared view, images of the view are sent across the network. These images are more traffic intense than just sending state change information. A possibility with shared view is to keep track of which areas of the view has changed, and send only those. This is similar to the technique used in e.g. remote desktops (See 5.2). If the user has a small device, and use shared view, other participants must either zoom the images, making them blurry, or use the activity in a small window.

A challenge with sharing is when using web applications needing a user log in. If a participant does not have a username on the web application, he will not be able to sign in to collaborate. This also brings in a new problem, which is that participants in an activity, should also have the same permissions in the web applications in use. The connection between all of the web applications and the ABC service could be tedious to implement. A solution to this, is to use shared view when collaborating on a web application which only one user has access to. However, this does also introduce another problem, which is that the user with the permissions has to stay signed on to give other participants his view. In many cases, the simplest thing to do is to manual assign permissions in the web application.

Context Awareness

As context awareness is a complex subject, it is difficult to integrate Activity Awareness into web browsers.

Web browsers can use geolocation based on IP-addresses to know where in the world they are. This will help getting location-based search results searching. Another possibility is to keep track of where the user is most of the time. Based on that information, the browser knows if he is not home. If he is on vacation somewhere else in the world, the web browser will suggest starting activities like search for local restaurants, tourist attractions, gas stations, nice beaches to go for a swim, etc. It can also suggest activities like blogging, chat with friends at home, etc. A problem with this is the privacy and personal record tracking. This feature could be enabled and disabled in the web browser configurations.

Another way to make web browsers context aware, is to enable them to communicate with external hardware sensors, RFID readers, etc. However, this does introduce a challenge in transferring this information to the web applcations - if they need it.

Architecture

Because many users have many devices, and if they all are part of ABC, it would generate a lot of traffic to the ABC mainframe. By choosing to host ABC in the cloud, we get both a single point-of-entry, i.e. it is easy to find for all web browsers, and it is highly scalable according to needs.

By separating the traffic distribution into client/server- and peer-to-peer-models, it is possible to save server load and network traffic. Chat, voice communication, etc. can be sent between the clients using peer-to-peer. These services, especially voice communication, does also benefit from the lower latency since the traffic does not have to go through the ABC servers. However, this works fine up to a certain point, and that is when there are many, many participants in an activity, and all uses voice communication. Then the client must send the same voice data to everyone, and could be bottlenecked by the Internet connection. A solution to this is to set a specific participants threshold for when to switch voice communication over to the ABC servers. Servers usually have far greater network bandwith. Another possibility is to use multicast technologies, but this is limited to local networks.

Cross-browser compatibility

For a complete experience of Activity-Based Computing, people must be able to bring their activities to whatever device they have. This will most likely lead to a scenario where e.g. Firefox needs to talk to Safari. By introducing a common protocol for all web browsers, this can be achieved. Although possible, this is a bit unlikely. Most web browsers already have their own synchronization service, and the different web browser vendors might not want to enable this feature or establish cooperation with the competitors.

A solution to this is to use plugins. All the large web browsers supports plugins. The plugin developer can use the same protocol between the plugin and ABC, and thus make ABC available on almost any computer running any browser. Plugin support is not so usual on mobile web browsers, which could lead to a problem when using mobile devices.

If using plugins, the developer could be an independent company with no relations to the web browsers. It could also be one of the web browser vendors, making plugins to other web browsers, while integrate it into their own.

External cloud services

ABC can be hosted by any company, and does not need to be the web browser company itself. It could be for instance Dropbox, Google, Microsoft, Facebook, etc. A benefit with one of these providers, is the huge user databases they have. If a person uses Internet, it is very likely that he has an user account on one of these companies' services. By integrating ABC with one of these services, most users already have a user and does not need to register on a new site.

8.3 Evaluation of research methods

8.3.1 Product analysis

To answer research question 1, a product analysis was a good approach for getting an overview over features in several products.

In this product analysis, I have chosen several different product categories (See chapter 5) to analyze. I find these categories relevant to show existing software products with functionality similar to the concepts of Activity-Based Computing. However, products like virtual desktops, remote desktops and remote applications are not necessarily important to answer the second part of this thesis - improvements for integrating ABC in web browsers. They can be used as examples on how to implement some of the ABC concepts.

I have analyzed at least two products within the same category. In some cases, the features are quite similar, e.g. virtual desktops (See 5.1) and office suites (See 5.4). In these categories, it would propably be enough to analyze one product, but then again, I had to actually find out if they were alike. In other categories there are quite different results, like remote desktops (See 5.2) and remote applications (See 5.3).

8.3.2 Concept development

Concept development was used to come up with improvements in web browsers for integrating ABC concepts, which is an answer to research question 2. It is a good way to present possible solutions, and can be used at an early stage in a product development process.

In this thesis, I have based the concept development on implementing ABC concepts in web browsers. With more and more web applications available, it is possible to do more tasks in the web browser. Web browsers are also often used to handle multiple tasks at a time. This makes web browsers a good starting point for ABC. Another bonus is that they are available on almost any device, which is a part of ABC.

Another approach could be to develop a concept with ABC as an external cloud based service - as a web application. This is described briefly in 8.2.

My evaluation is that web browsers are best suited to handle Activit-Based Computing compared to a web application.

Chapter 9

Conclusion

This chapter contains the conclusion (9.1) and introduces suggestions for further work (9.2).

9.1 Conclusion

This thesis covers a product analysis to see which Activity-Based Computing concepts exists in current software products and web browsers. The conclusion is a summary of how the two research questions, stated in 1.4, are answered. These are discussed in chapter 8.

9.1.1 ABC concepts in current products

RQ1: *To what extent do we find functionality similar to the concepts of Activity-Based Computing in current cloud computing services and web browsers?*

A summary of all the analyzed products can be found in table 8.1 and the answer to this research question is discussed in 8.1.

The product analysis shows that many products already have functionality similar to several of the Activity-Based Computing concepts. However, none of the products incorporates functionality similar to all of the ABC concepts. The products that are closest to a full implementation of the ABC concepts are the web browsers.

Some product groups, like web browsers and office suites, have more in common with ABC than other groups, like remote desktops. Products belonging to the same product group has often the same similarities to ABC.

None of the products are based on the Activity-Based Computing research. They have developed similar functionality on their own, completely separate.

The product analysis is found in chapter 5 and 6.

9.1.2 Improvements to enable ABC

RQ2: *How can current cloud computing services and web browsers be improved to support Activity-Based Computing?*

The answer to this research question is discussed in 8.2.

Web browsers already exists on most computers and devices, and with web applications becoming more and more common, the concept development in this thesis is based on improvements in web browsers to support ABC.

The five web browsers analyzed in this thesis supports several features similar to the ABC concepts.

Windows and tabs can be used as activities, where one window is one activity, and the different tabs inside a window are the resources relevant to the specific activity.

An activity can store its window and tabs information, in addition to its state, and use it to be able to suspend and then resume it later.

By storing this information in the cloud, on an ABC service, the web browser can use this activity from anywhere, also known as roaming.

Most web browsers have a mobile version, which works on smaller screens, with touch input. By implementing synchronization of activities, including support for tab groups, and the to ability to zoom in on web applications, the web browsers can be adapted to fit mobile devices.

Sharing and collaboration support is non-existant in current web browsers. By implementing sharing of an activity state between web browsers, i.e. tabs, tab in focus, etc. the web browsers enables users to collaborate.

Web browsers can use geolocation to report its location to web services. The web browser should also be able to communicate with external hardware, like sensors, RFID readers, etc.

The improvements are described more in details in chapter 7.1 and discussed in 8.2.

9.2 Further work

The next logical step would be to actually integrate Activity-Based Computing within a web browser. Development of a system is the third, and next, research apporach according to Shaw (See 1.5).

Both Mozilla Firefox (See 6.1) and Google Chrome (See 6.3) is built on open source code, and thus makes a good starting point for further development and taking this project to the next level.

Bibliography

[1] Chloe Albanesius. *Mozilla Releases Updated Firefox for Android*. URL: `http://www.pcmag.com/article2/0,2817,2406317,00.asp` (visited on 06/25/2013).

[2] Amazon. *Amazon EC2*. URL: `http://aws.amazon.com/ec2/` (visited on 06/23/2013).

[3] Apple. *iCloud*. URL: `http://www.apple.com/icloud/` (visited on 06/30/2013).

[4] Apple. *iOS*. URL: `http://www.apple.com/ios/` (visited on 06/30/2013).

[5] Apple. *iPhone built-in apps*. URL: `http://www.apple.com/iphone/built-in-apps/` (visited on 06/25/2013).

[6] Apple. *Mission Control*. URL: `http://www.apple.com/osx/apps/all.html#missioncontrol` (visited on 06/18/2013).

[7] Apple. *OS X*. URL: `http://www.apple.com/osx/` (visited on 06/11/2013).

[8] Apple. *Safari*. URL: `http://www.apple.com/safari/` (visited on 06/24/2013).

[9] Michael Armbrust et al. "A view of cloud computing". In: *Communications of the ACM* 53.4 (2010), pp. 50–58.

[10] Opera Software ASA. *Opera browser - the alternative web browser*. URL: `http://www.opera.com/` (visited on 06/19/2013).

[11] Opera Software ASA. *Opera Link*. URL: `http://www.opera.com/link` (visited on 06/20/2013).

[12] Opera Software ASA. *Opera Mini*. URL: `http://www.opera.com/mobile/mini` (visited on 06/20/2013).

[13] Opera Software ASA. *Opera Mobile*. URL: `http://www.opera.com/mobile` (visited on 06/19/2013).

[14] Lee Badger et al. "Cloud computing synopsis and recommendations". In: *NIST Special Publication* 800 (2012), p. 146.

[15] E Bardram. "Activity-based computing: support for mobility and collaboration in ubiquitous computing". In: *Personal and Ubiquitous Computing* 9.5 (2005), pp. 312–322.

[16] Jakob Bardram, Jonathan Bunde-Pedersen, and Mads Soegaard. "Support for activity-based computing in a personal computing operating system". In: *Proceedings of the SIGCHI conference on Human Factors in computing systems*. ACM. 2006, pp. 211–220.

[17] Jakob Bardram, Afsaneh Doryab, and Sofiane Gueddana. "Activity-Based Computing–Metaphors and Technologies for Distributed User Interfaces". In: *Distributed User Interfaces*. Springer, 2011, pp. 67–74.

[18] Jakob E. Bardram. "Activity-based computing for medical work in hospitals". In: *ACM Transactions on Computer-Human Interaction (TOCHI)* 16.2 (June 2009), 10:1–10:36. ISSN: 1073-0516. DOI: 10.1145/1534903.1534907. URL: http://doi.acm.org/10.1145/1534903.1534907.

[19] Jakob E Bardram. "From desktop task management to ubiquitous activity-based computing". In: *Integrated Digital Work Environments: Beyond the Desktop Metaphor* (2005), pp. 49–78.

[20] Jakob E Bardram. "Supporting mobility and collaboration in ubiquitous computing". In: *Aarhus, Denmark, Tech. Rep* (2003).

[21] Jakob E Bardram. "The Activity-Based Computing Project". In: *AAAI Workshops; Workshops at the Twenty-Fifth AAAI Conference on Artificial Intelligence* (2011).

[22] Jakob E Bardram and Henrik B Christensen. "Pervasive computing support for hospitals: An overview of the activity-based computing project". In: *Pervasive Computing, IEEE* 6.1 (2007), pp. 44–51.

[23] Jakob E Bardram et al. "CLINICAL SURFACES–Activity-Based Computing for Distributed Multi-Display Environments in Hospitals". In: *Human-Computer Interaction–INTERACT 2009*. Springer, 2009, pp. 704–717.

[24] Jonathan Bunde-Pedersen, Martin Mogensen, and Jakob E Bardram. "The ABC adaptive fusion architecture". In: *Proceedings of the 4th international workshop on Middleware for Pervasive and Ad-Hoc Computing (MPAC 2006)*. ACM. 2006, p. 1.

[25] Cetrea. *Cetrea A/S*. URL: http://cetrea.com (visited on 06/23/2013).

[26] Zoho Corporation. *Zoho.com*. URL: https://www.zoho.com/ (visited on 06/11/2013).

[27] Paolo Costa et al. “NaaS: Network-as-a-Service in the Cloud”. In: *Proceedings of the 2nd USENIX conference on Hot Topics in Management of Internet, Cloud, and Enterprise Networks and Services, Hot-ICE.* Vol. 12. 2012, pp. 1–1.

[28] Nigel Davies, Daniel P Siewiorek, and Rahul Sukthankar. “Activity-based computing”. In: *Pervasive Computing, IEEE* 7.2 (2008), pp. 20–21.

[29] Shyam Kumar Doddavula, Ira Agrawal, and Vikas Saxena. “Cloud Computing Solution Patterns: Infrastructural Solutions”. In: *Cloud Computing.* Springer, 2013, pp. 197–219.

[30] Dropbox. *Dropbox.* URL: `https://www.dropbox.com/` (visited on 06/05/2013).

[31] Evince. *Evince - Document Viewer.* URL: `http://projects.gnome.org/evince/` (visited on 06/05/2013).

[32] Python Software Foundation. *Python.* URL: `http://www.python.org/` (visited on 06/28/2013).

[33] freedesktop.org. *freedesktop.org.* URL: `http://www.freedesktop.org/wiki/` (visited on 06/10/2013).

[34] GeoBytes. *IP Locator.* URL: `http://www.geobytes.com/iplocator.htm` (visited on 06/30/2013).

[35] Google. *Chrome Mobile.* URL: `http://www.google.com/intl/en/chrome/browser/mobile/` (visited on 06/20/2013).

[36] Google. *Chrome OS.* URL: `http://www.google.com/intl/en/chrome/devices/` (visited on 06/19/2013).

[37] Google. *Chromium.* URL: `http://www.chromium.org/` (visited on 06/20/2013).

[38] Google. *Google App Engine.* URL: `https://developers.google.com/appengine/` (visited on 06/28/2013).

[39] Google. *Google Apps.* 2013. URL: `www.google.com/a/` (visited on 06/04/2013).

[40] Google. *Google Chrome.* URL: `https://www.google.com/intl/en/chrome/browser/` (visited on 06/19/2013).

[41] Google. *Google Drive.* URL: `http://www.google.com/intx/en_uk/enterprise/apps/business/products.html#drive` (visited on 06/20/2013).

[42] The PHP Group. *PHP.* URL: `http://php.net/` (visited on 06/28/2013).

[43] D Austin Henderson Jr and Stuart Card. “Rooms: the use of multiple virtual workspaces to reduce space contention in a window-based graphical user interface”. In: *ACM Transactions on Graphics (TOG)* 5.3 (1986), pp. 211–243.

[44] Hightech Highway. *Cloud Computing: Yesterday, Today and Tomorrow.* URL: `http://www.hightech-highway.com/virtualize/cloud-computing-yesterday-today-and-tomorrow/` (visited on 06/21/2013).

[45] Steven Houben et al. “Activity-centric support for ad hoc knowledge work: a case study of co-activity manager”. In: *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* ACM. 2013, pp. 2263–2272.

[46] Steven Houben et al. *co-Activity Manager: ABC for Knowledge Work.* URL: `http://itu.dk/pit/abc/?page_id=215` (visited on 06/21/2013).

[47] Steven Houben et al. “Co-activity manager: integrating activity-based collaboration into the desktop interface”. In: *Proceedings of the International Working Conference on Advanced Visual Interfaces.* ACM. 2012, pp. 398–401.

[48] Joyent Inc. *Node.js.* URL: `http://nodejs.org/` (visited on 06/29/2013).

[49] ITU. *IT University of Copenhagen.* URL: `http://itu.dk/` (visited on 06/23/2013).

[50] Jaatun. “Evaluering av et aktivitetsbasert konsept for helserettede informasjonssystemer”. MA thesis. NTNU, 2008.

[51] Canonical Ltd. *Ubuntu.* URL: `http://ubuntu.com/` (visited on 06/05/2013).

[52] Canonical Ltd. *Unity.* URL: `http://unity.ubuntu.com/` (visited on 06/05/2013).

[53] Sean Ludwig. *Cloud 101: What the heck do IaaS, PaaS and SaaS companies do?* URL: `http://venturebeat.com/2011/11/14/cloud-iaas-paas-saas/` (visited on 06/05/2013).

[54] LXDE.org. *Lightweight X11 Desktop Environment.* URL: `http://lxde.org/` (visited on 06/17/2013).

[55] Microsoft. *ASP.NET.* URL: `http://www.asp.net/` (visited on 06/29/2013).

[56] Microsoft. *Internet Explorer.* URL: `http://windows.microsoft.com/en-us/windows-8/browse-web` (visited on 06/24/2013).

[57] Microsoft. *Microsoft RemoteApp.* URL: `http://www.microsoft.com/en-us/windows/enterprise/products-and-technologies/virtualization/rds.aspx` (visited on 06/18/2013).

[58] Microsoft. *Microsoft SkyDrive*. URL: `http://windows.microsoft.com/en-us/skydrive/` (visited on 06/17/2013).

[59] Microsoft. *Office 365*. URL: `http://office.microsoft.com/en-us/` (visited on 06/29/2013).

[60] Microsoft. *Office Web Apps*. URL: `http://office.microsoft.com/en-us/web-apps/` (visited on 06/29/2013).

[61] Microsoft. *Remote Desktop Services*. URL: `http://msdn.microsoft.com/en-us/library/windows/desktop/bb892075(v=vs.85).aspx` (visited on 06/19/2013).

[62] Microsoft. *Windows Azure*. URL: `http://www.windowsazure.com/en-us/` (visited on 06/28/2013).

[63] Microsoft. *Windows Desktop Sharing*. URL: `http://msdn.microsoft.com/en-us/library/windows/desktop/bb968809(v=vs.85).aspx` (visited on 06/19/2013).

[64] Mozilla.org. *Firefox for Android*. URL: `http://www.mozilla.org/en-US/firefox/fx/#mobile` (visited on 06/20/2013).

[65] Mozilla.org. *Firefox Sync*. URL: `http://support.mozilla.org/en-US/kb/firefox-sync-take-your-bookmarks-and-tabs-with-you` (visited on 06/20/2013).

[66] Mozilla.org. *Mozilla Firefox*. URL: `http://www.mozilla.org/en-US/firefox/` (visited on 06/19/2013).

[67] NTNU. *NTNU Software Farm*. URL: `https://innsida.ntnu.no/wiki/-/wiki/English/Software+Farm` (visited on 06/17/2013).

[68] OpenSSH. *OpenSSH*. URL: `http://www.openssh.org/` (visited on 06/10/2013).

[69] OpenStack. *OpenStack Cloud Software*. URL: `http://www.openstack.org/` (visited on 06/23/2013).

[70] Oracle. *Java*. URL: `http://www.java.com/en/` (visited on 06/28/2013).

[71] Ormberg. "Activity Based Computing: Health workers and the principles of ABC". MA thesis. NTNU, 2009.

[72] Steven Ovadia. "Syncing bookmarks: An overview of current options". In: *Behavioral & Social Sciences Librarian* 31.1 (2012), pp. 76–79.

[73] The Linux Information Project. *The X Window System: A Brief Introduction*. 2013. URL: `http://www.linfo.org/x.html` (visited on 06/09/2013).

[74] Rackspace. *Rackspace.* URL: http://www.rackspace.com/ (visited on 06/23/2013).

[75] Remmina. *Remmina.* URL: http://remmina.sourceforge.net/ (visited on 06/19/2013).

[76] Salesforce.com. *Salesforce.* 2013. URL: http://www.salesforce.com/ (visited on 05/21/2013).

[77] Mary Shaw. "The coming-of-age of software architecture research". In: *Proceedings of the 23rd international conference on Software engineering.* IEEE Computer Society. 2001, p. 656.

[78] Mary Shaw. "Writing good software engineering research papers: minitutorial". In: *Proceedings of the 25th international conference on software engineering.* IEEE Computer Society. 2003, pp. 726–736.

[79] Kshitij Sobti. *Google Chrome Finally Comes to Android.* URL: http://www.thinkdigit.com/Internet/Google-Chrome-Finally-Comes-to-Android_8702.html (visited on 06/25/2013).

[80] Soft32. *Web Browser Roundup.* URL: http://www.soft32.com/blog/platforms/windows/web-browser-roundup/ (visited on 06/21/2013).

[81] StatCounter. *StatCounter Global Stats.* URL: http://gs.statcounter.com/about (visited on 06/24/2013).

[82] StatCounter. *StatCounter Global Stats - statistics from 2008-2013.* URL: http://gs.statcounter.com/#browser-ww-monthly-200807-201305 (visited on 06/24/2013).

[83] ABC Research Team. *Activity-Based Infrastructures and Interfaces.* URL: http://itu.dk/pit/abc/?page_id=158 (visited on 06/21/2013).

[84] ABC Research Team. *What is ABC?* URL: http://itu.dk/pit/abc/?page_id=10 (visited on 06/15/2013).

[85] The Go Team. *Go.* URL: http://golang.org/ (visited on 06/28/2013).

[86] Darkside Technologies. *X Server.* URL: https://play.google.com/store/apps/details?id=au.com.darkside.XServer&hl=en (visited on 06/10/2013).

[87] Paul Thurrot. *Welcome to Windows 8.* URL: http://winsupersite.com/article/windows-phone-8/windows-phone-8-144662 (visited on 06/25/2013).

[88] Twiddla. *Team Whiteboarding with Twiddla.* URL: http://www.twiddla.com/ (visited on 06/17/2013).

[89] John Viega. "Cloud computing and the common man". In: *Computer* 42.8 (2009), pp. 106–108.

[90] Team Viewer. *Team Viewer - the All-In-One Software for Remote Support and Online Meetings*. URL: `http://www.teamviewer.com/` (visited on 06/19/2013).

[91] Lizhe Wang et al. "Cloud computing: a perspective study". In: *New Generation Computing* 28.2 (2010), pp. 137–146.

[92] Aaron Weiss. "Computing in the clouds". In: *networker* 11.4 (2007).

[93] WOLF. *Cloud computing*. 2013. URL: `http://www.wolfframeworks.com/cloudcomputing.asp` (visited on 05/22/2013).

[94] X.Org. *X.Org Foundation*. URL: `http://www.x.org/wiki/` (visited on 06/10/2013).

[95] xpra.org. *X Persistent Remote Applications*. URL: `http://xpra.org/` (visited on 06/10/2013).

[96] Qi Zhang, Lu Cheng, and Raouf Boutaba. "Cloud computing: state-of-the-art and research challenges". In: *Journal of Internet Services and Applications* 1.1 (2010), pp. 7–18.