The Basics For Futureproof RIA User Interface Design

Analyzing trends when moving towards Rich Internet Applications

MASTER THESIS BY
Wouter Vermeulen B.Sc.

UNIVERSITY
Vrije Universiteit, Amsterdam

SUPERVISOR
dr. Gerrit van der Veer

SECOND READER
Name XXX

DATE
August, 2010

1 Introductory annotations

VERSION INFORMATION
Version 1, published August 2010

CORRESPONDING MASTER
Information Science

Faculty of Science (Multimedia & Culture)

Free University, Amsterdam, The Netherlands

2 Abstract

ABSTRACT TEXT

In this thesis, W.Vermeulen B.Sc. will analyze the change from traditional web applications to Rich Internet Applications (RIA) while the primarily focus is at user interface (and interaction) design. The analyses will include what RIA's are by defining the term, why they exist and more so why there is a "need" for RIA's nowadays.

After some essential terminology, containing definitions and basic knowledge of hard- and software technology, the author introduces a model, the EORIA-model, that extends the definition and helps user interface designers and information engineers to decompose RIA's and have an overview to overcome challenges when designing a RIA.

From there, we will discuss trends and changes when moving from traditional web applications to RIA's. Also a comparison has been made with traditional web applications (or user interface web design) to finally conclude what the changes are for anyone interested in designing and/or developing a user interface (and its interactions). Subsequently a deconstruction of a RIA user interface is made and design patterns are defined and compared with traditional web applications to see the difference.

As a bonus and as a case study, and to put it in to practice, a usability research has been carried out at a large Dutch TV Broadcaster to see whether some of the findings and conclusions found in literature are correct and if something might have been overlooked.

The conclusion notices that there, indeed, are noticeable changes concerning user interface design when moving towards RIA.

KEYWORDS

Rich Internet Applications (RIA), Web 2.0, Usability, User Interface Design, Interaction, Design Patterns, Flex, Flash, AJAX

3 Table of contents

APPENDICES

4 Introduction

The Web is, fortunately, constantly evolving. Since the first webpage in 1991 (at *CERN*)[1] we have gone through multiple phases of the development, of which the internet bubble (or *dotcom bubble*) has been recognized as significant[2]. Others deny the existence of the dotcom bubble (Kearns, 2004) and say that the evolution of the Web is just continuous (Bouras, 2003). At least the last 10 years (since 2000) of this evolution have been marked by a phase with the, later on introduced, significant and well-known term "Web 2.0".[3]

The Web became social, more interactive and is being used all the time for professional purposes rather than just visiting self built hobby-websites of some old schoolmate. Think of banking, shopping, auctions and social platforms.[4]

This change placed a strong emphasis on user interface design. Building websites became serious business and names like Nielsen[5], O'Reily[6] and Krug[7] have greatly influenced the way we think about user interface design on the Web.

This was of great necessity, especially because building a website is a way of designing and developing software accessible to wide public, rather than only to those that are trained and skilled and the levels of knowledge are therefore widespread. Add to this that within the browser everything, if technically possible, is allowed, even if that brings along terrible user interfaces and bad interaction.

The author believes that we are currently at the beginning of a phase in web evolution where we will be hearing more and more of a new phenomena: Rich Internet Applications (or RIA in short). They are to set trends concerning user interfaces for years to come.

This thesis should clearly define and explain RIA's and everything involved, as a great help for user interface designers (and anyone interested in the field), to anticipate upon the changes that exist when moving towards this kind of *web applications*.

[1] CERN, see http://info.cern.ch/
[2] Cassidy J. How America Lost it Mind and Its Money in the Internet Era (2002)
[3] O'Reily Tim, What is Web 2.0? (2005)
[4] Knemeyer Dirk, There are only four things that people do on the Web (2003)
[5] Nielsens' biography, http://www.useit.com/jakob/
[6] Tim O'Reily, http://www.oreillynet.com/pub/au/27
[7] Steve Krug, http://www.sensible.com/

4.1 Research questions

This section describes all the questions of central concern in this research, which we'll try to answer as well as possible, leading to our final research question.

We are going to have a conversation about Rich Internet Applications. But it is not totally clear where the boundaries of the concept "RIA" are defined. Let's not be confused by today's buzz-word-marketers and define the term so that we are able to work (scientifically) with it. This is why we pose the following questions:

- What is a Rich Internet Application (RIA)?
- Why RIA?
- How should a RIA user interface be (de)constructed?
- What user interface design patterns for RIA exist (or none at all)?

Because we expect that there are certain particular technologies for building RIA's, we ask ourselves which and, more important, what their restrictions concerning usability are. This leads us to the following questions:

- Which technologies for RIA's exist?
- What are the usability issues with these technologies for user interface designers?

It might as well be the case that there is no difference, compared with the current situation for user interface designers, while designing web applications. Therefore we carry out a comparison and ask ourselves the following questions:

- How do RIA and traditional web applications differ, regarding interface design?
- Do the RIA user interface and design patterns differ to these for "traditional" webpages?

FINAL RESEARCH QUESTION

All of the above will result in the following formulated research question:

- Are there any noticeable changes for user interface designers while designing a RIA?

The answer will be given in the conclusion, where you will find a summarized explanation why.

4.2 Thesis structure

CHAPTER 1-4: INTRODUCTIONARY MATERIAL
Basic thesis-opening content inclusive research questions and introduction.

CHAPTER 5: TERMINOLOGY
We'll start by describing some basic terminology and concepts we will be using to answer some of our questions.

CHAPTER 6: THE EORIA-MODEL
Shows Extendable Open model for RIA's, that helps to decompose them and see the whole. Helpful for designers to set-up patterns and take note of what they need to pay attention to.

CHAPTER 7: TRENDS & CHANGES
In this chapter we discuss the background of our subject. We start with trends that will answer the question "why RIA?" and help us predict future comings, a comparison with traditional web applications is made.

CHAPTER 8: DESIGN PATTERNS FOR RIA USER INTERFACE
This chapter will do a deconstruction of the RIA user interface and illustrate some design patterns especially for RIA which will be compared with current ones to emphasize the change.

CHAPTER 9: CASE STUDY: RESEARCH IN PRACTICE
In addition to the literature study, we do a case study in the form of a usability research to a Rich Internet Application at a large Dutch TV broadcaster.

CHAPTER 10: CASE STUDY: RESULTS
The results of the case study are shown including an analysis and a conclusion to see what the added value for this thesis is.

CHAPTER 11: CONCLUSION
The conclusion is the place where we end our journey through fascinating literature as well as the case study and answer our research questions in short, finally concluding with the answer to our main question.

CHAPTER 12: AFTERTHOUGHTS
Some afterthoughts on the good and the bad within this study and some suggestions for further research.

Chapter 13: References

The references that we have been using for this thesis divided in paragraphs of literature, websites, books and figures.

Chapter 14: Acknowledgements

My sincere gratitude.

Appendices

In the back of this thesis, you'll be able to find a thick bundle of appendices mainly belonging to the case study (results).

4.3 Reading guide

The target audiences which can be defined for this thesis are:

- Designers, user interface and interaction designers;
- Researchers and students, in the field of usability and interface design;
- General audience, furthermore anyone who is interested in the field of usability and especially Rich Internet Application (RIA);

Below, I define a per-chapter audience focus that might be of help to you, especially when you have less time. I still strongly recommend to read the whole of this brilliant thesis! ☺

CHAPTER 1-4: GENERAL AUDIENCE
Basic thesis-opening content inclusive research questions and introduction.

CHAPTER 5: GENERAL AUDIENCE
This chapter is an obligation for everyone trying to understand the core of the subject.

CHAPTER 6: DESIGNERS, RESEARCHERS & STUDENTS
For anyone willing to have a clear perspective toward RIA's.

CHAPTER 7: GENERAL AUDIENCE
For anyone willing to know the cause of RIA.

CHAPTER 8: DESIGNERS
You must read this if you currently make frequent use of design patterns.

CHAPTER 9: CASE STUDY: RESEARCHERS & STUDENTS
Especially interesting for those planning or (willing to) to conduct a research study.

CHAPTER 10: CASE STUDY: GENERAL AUDIENCE
The results of the case study are shown including an analysis and a conclusion to see what the added value for this thesis is.

CHAPTER 11: GENERAL AUDIENCE
For anyone willing to know the final conclusion.

CHAPTER 12: RESEARCHERS & STUDENTS
For further research highly recommended.

CHAPTER 13: GENERAL AUDIENCE
It's also recommended for everyone to explore the reference list to see if it contains something of your interest.

5 Terminology

Because much terminology, even for professionals, could be unknown or unclear, and since we want to have a clear understanding of what we're having a conversation about with everyone who's reading beyond this page, we'll use this chapter to explain important terms and techniques which are part of the core of this research.

Before continuing we should make one remark that certainly not all terms are defined here. A definition list for all italicized words in this thesis can be found in appendix I.

The terms and their definitions and (or) their explanations you'll find next are ordered in categories first and subsequently in alphabetical order. The categories are concepts, software and hardware.

5.1 Concepts

Main concept is "Rich Internet Applications", with which we will start. Furthermore we'll describe Web 2.0, which is a closely related concept in this field of research.

5.1.1 Rich Internet Applications

Let's get down to the core definition at once: Rich Internet Application, which is of course our major concept. Before we define one, let's see what others have stated about this concept.

Wikipedia defines it as:

> *"Rich Internet Applications (RIAs) are web applications that have many of the characteristics of desktop applications, typically delivered either by way of a site-specific browser, via a browser plug-in, or independently via sandboxes or virtual machines."*[8]

Many others state it like:

> *"Rich Internet Application - a Web 2.0 (AJAX) based web application that reacts to user actions without needing to redisplay the whole page, perhaps just updating one small part. RIAs have much greater responsiveness than traditional web-based systems which required either navigation to new a page or at least refreshing of the current page to fetch new data or perform back end actions."*[9]

And also this one:

> *"Rich Internet Application (RIA) — a web application which functionality and interface capabilities are identical or close to its desktop analogs. The other important specific of RIA is that it can work in offline mode."*[10]

For all understanding a last definition from Schroeder (2002)[11]:

> *" Rich Internet Applications extend the web and HTML without replacing them, enabling the development of applications that offer significantly more intuitive, responsive, and effective user experiences."*

[8] Wikipedia, http://en.wikipedia.org/wiki/Rich_Internet_application

[9] BurntJet, The glossary of Terms for Programming (2008)

[10] Program-ACE, Glossary (2010)

[11] Schroeder, et.al. Enhancing User Interaction in Pet Market. (2002)

This should give us a little bit of a clue. Now we are able to retrieve a few specific properties of RIA, namely:

1. They are web applications
2. They have interface capabilities close to desktop analogs
3. They have rich/responsive interactions
4. They have the ability to load partial content
5. They have the ability to work off-line (if temporary no internet connection available)

We should state that the second mentioned source (9) is one that contains confusion that is seen a lot. The term Web 2.0 is mixed up with RIA. As we are going to see in the next paragraph, AJAX is certainly not the only technology that can construct a RIA. So that definition is even for Web 2.0 not true. The only detail that partial content is loaded is a characteristic for as well RIA as Web 2.0. Read more about Web 2.0 in 5.1.2.

Let's state a basic definition to work with:

> *"A Rich Internet Application is a web-based application that entails a rich and responsive user interface which behaves closely as desktop analogs and contains the ability to load partial data, but also to store data if no internet connection is available."*[12]

See chapter 6 for the EORIA-model, which extends our definition and clearly shows the boundaries of a RIA system helping designers to note points of attention.

[12] Definition by Wouter Vermeulen, B. Sc.

5.1.2 Web 2.0

As stated in the previous paragraph, the concept of which much has been written about, even more than Rich Internet Applications, is Web 2.0.[13] Although the concept of Web 2.0 is much better known than RIA they sometimes get confused. Understandable because they are tightly related.

If the terms in this chapter were ordered chronologically, this one should have been before RIA. Web 2.0 is a buzz-word as well as a good defined term which works scientifically and hype-proof. The bachelor research of the author of this thesis has a lot to say about Web 2.0.[14]

Think of Web 2.0 as a state in the evolution of the Web with strong emphasis on the social element, enabled by advanced technology and techniques such as AJAX and better performing user hardware. What we have read in the previous chapter is true: content should be partially loaded (mostly using AJAX, which is explained in paragraph 5.2.4) to enhance the Web 2.0 experience. This also worked against usability, for example the back-button gets another meaning.[15] This is because users think they go back one step, but they go back one step in the browser history stack. Later on, we'll discuss that the same sort problem arises with other technology (5.2.1.1).

We will not go too much into Web 2.0 because it's not our main focus. For all convenience we'll state the Wikipedia definition:

> *"The term Web 2.0 is commonly associated with web applications that facilitate interactive information sharing, interoperability, user-centered design, and collaboration on the World Wide Web."*[16]

Please read through the reference list at the end of this thesis to find more literature on Web 2.0 if that is of your interest.

[13] Deloitte, Change your World, or the World will Change you (2008)
[14] Vermeulen W., Boer den W., De web browser interface (2008)
[15] Garett Jesse James, Ajax: A New Approach to Web Applications (2005)
[16] See Wikipedia - http://en.wikipedia.org/wiki/Web_2.0

5.2 Technology - Software

We're glad to explain some software terms in this chapter. This is software or techniques user interface designers are most likely to be confronted with, if they are designing a RIA and more likely often used by anyone executing a RIA.

This should be some basic knowledge to have subject matter when talking about RIA's.

5.2.1 The Adobe Flash Platform

Adobe® Systems is the leading software company in particular in the multimedia and creativity industry, the largest and most known.[17] The biggest commercially launched platform for building and executing RIA's is the Adobe Flash platform. The platform consists of multiple products which are all generally based upon the Flash format. They will be explained in the following paragraphs.

The Flash platform has been criticized a lot [18] and not only usability but also indexabillity has been a tough point. But Adobe worked together with companies like Google[19] on a SEO-procedure[20] to improve on the matter.

The good qualities of the Adobe Flash Platform include however:

1. Animation, the ability to create highly interactive and animated interfaces (websites)
2. Easy import, from other (Adobe)(market leading) graphical products
3. Low learning curve, even programming for designers
4. Video, tremendous possibilities concerning video (see Huurdeman, 2007)
5. High availability, plug-in penetration of about 97,5%[21]
6. Cross platform, platform independent[22]

The high animation ability while maintaining a small file size is due to the fact that it is a vector-based tool. Because of its characteristics it's not only used for webpages but also for games, presentations and advertisements.[23]

On the next page, we will discuss the products originated from the Adobe Flash Platform, starting with the foundation of them all: Flash.

[17] Wikipedia, http://en.wikipedia.org/wiki/Adobe_Systems
[18] Schaller David T. (et al.), To Flash or Not To Flash? (2004)
[19] Adler Ron, Google Learns to Crawl Flash (2008)
[20] Piyasirivej, Pilun, Towards usability evaluation of Flash Websites (2004)
[21] Millward Brown (for Adobe), Flash Player Version Penetration (2010)
[22] Except for the 64-bit versions of the plug-in at the moment of writing, see the product website.
[23] Huurdeman H.C., Interactive Video in Serious Games (2007)

5.2.1.1 Adobe Flash

We start with Flash™ [24] because that is the most known and also the oldest shoot in the family. All other techniques following are based upon the same file structure, having a .swf extension.

In 1996, the first version of Flash was launched as a vector based drawing tool. With the Internet becoming more popular and FutureSplash being a rival to Macromedia's® Shockwave, Macromedia acquired FutureSplash and launched it under the banner of Flash. Later on, in 2005 Adobe acquired Macromedia.[25]

When talking about Flash, one can mean either one of two things: one, the Flash authoring tool or two the plug-in that is needed to execute Flash (within or not within the *web browser*).

A short explanation of *the authoring tool* is in place here. The interface consists of a timeline containing multiple layers of frames and key frames. At the key frames changes are made. The tool contains some smooth transitions that can be used between key frames. Graphical objects can exist stand alone or in movieclips that can be animated. For more on the authoring tool I recommend reading Adobe Flash CS3 Professional Bible[26]. Current version of the authoring tool is CS5.

In addition to the timeline, Flash has its own programming language called ActionScript (AS). The first widely adopted version of the language was version 2.0 that can be compared to JavaScript. Later on, Adobe introduced version 3.0 (current) because there was a need for greater stability, more structured coding and easier reuse of code. AS 3.0 allows for *object-oriented programming* (OOP) and can be compared with Java.

As said, Flash has always struggled with a bad reputation[27] regarding usability.[28] Especially in the early days because then there were no techniques like Swfaddress[29]. These sort of techniques keep track of the state of the *application* due to a communication between Flash and JavaScript (sometimes continued to the back-end). Without keeping track of these application states the user would, for example, use the back button assuming he's going back, but he's resetting the Flash application to the beginning. This makes the browser user interface basically useless and allows no possibility for deeplinking.

Much more usability issues with Flash are present. Please read the references as stated in the text if they are of your interest.

[24] Flash official product page, http://www.adobe.com/products/flash/
[25] Wikipedia, http://en.wikipedia.org/wiki/Adobe_Flash#History_2
[26] Reinhardt Robert, Down Snow, Adobe Flash CS3 Professional Bible (2007)
[27] Nielsen J., Flash 99% Bad (2000)
[28] Holzinger Andreas, Interaction and Usability of Simulations & Animations (2003)
[29] See SWFaddress by Asual, http://www.asual.com/swfaddress/

5.2.1.2 Adobe Flex

The next tool in line is, chronologically, Adobe Flex™. Adobe Flex is a software development kit released by Adobe Systems for the development and deployment of cross-platform rich Internet applications based on the Adobe Flash platform.[30] Flex applications can be written using Adobe Flex Builder or by using the freely available Flex compiler from Adobe.

The main difference between Flex and Flash is that Flex has no support for the timeline anymore. A basic Flex application consists of AS 3.0 code (preferably arranged in packages) and one or more MXML file(s) specifying the objects that are displayed. MXML is Adobe's own *XML* format describing properties of user interface elements (for example width and height of a text box).

The basic authoring tool for Flex is based upon Eclipse[31], a professional developers tool, but also alternatives like FTD[32] and notepad can be used to code.

A Flex application must be compiled. This can be done with the free Flex SDK.[33]

5.2.1.3 Adobe Flash Catalyst

Relatively new to the Adobe family is Adobe Flash Catalyst™.[34] This is a tool that allows visual designers to work closely with user interface designers and then send the files to Adobe Flex developers for building the actual application. The tool can be used excellently for *Rapid Application Development* (RAD) and *Prototyping*.

In my opinion, software like Flash Catalyst marks the beginning of a new era of application development. Let's take a look at the distinguishing workflow of this authoring tool.

The workflow consists of three steps:

1. **Import visual design file**
   The imported files should have the file format of Adobe's leading graphical software suites (Adobe Photoshop and Illustrator). All layers are directly converted into usable objects for adding interaction. If no visual design file is present one can go into wire-frame mode and use Adobe's defined user interface elements.
2. **Design user interface and its interactions**
   One of the most important things to note is that Flash Catalyst is able to produce dynamic interactions without adding a line of code.

[30] Flex product page, http://www.adobe.com/products/flex/
[31] See the Eclipse software development page, http://www.eclipse.org/
[32] See the Power Flasher FDT product page, http://www.fdt.powerflasher.com/
[33] See the official Flex SDK website, http://opensource.adobe.com/wiki/display/flexsdk/Flex+SDK
[34] Flash Catalyst product page, http://www.adobe.com/products/flashcatalyst/

3. **Export**

   Now you are ready to export the file to a working application in the .swf format, or when the application is more complex, export the file for a Flex developer for more advanced development such as complex communications with a database server, user verification etc.

All of this is done in the Flash Catalyst authoring tool, being part of the latest release of suites, CS5.

5.2.1.4 Adobe AIR

That Adobe is focusing on development of RIA's is shown by their launch of Adobe AIR[35]. AIR™ stands for Adobe's Integrated Runtime and is a cross-platform environment for building RIA's using the Flash platform (as described here), HTML or AJAX. Applications then can be deployed as desktop applications, easily be launched from the taskbar and (as RIA's should be) available when working in offline mode.

Even though there are no statistics of AIR being known or being used, it's doubtful that it's by a large number of people. We should say that it has passed over 100 million installations[36], probably mainly due to the fact that it installs silently with other Adobe products.

5.2.2 Microsoft Silverlight

Seeing the success of Adobe, Microsoft® came up with a competitive product[37] Silverlight™ [38], based upon their .NET framework. Silverlight does basically the same as Flash with-in the browser and is now (just like the Adobe Flash Platform) extending to the mobile market, mostly focused on Windows Phone 7.

Silverlight is clearly part of the Microsoft family, which brings along great integration with other Microsoft products. Advantages of Silverlight (compared to Flash) include:

1. Stability and high results in performance tests[39]
2. A great community of .NET developers
3. Great compatibility with Microsoft products and WMP streaming services

The disadvantages include:

1. Supposed higher learning curve
2. Not so much integrated with other creative products
3. Lower penetration of plug-in[40]

[35] Adobe AIR official product page, http://www.adobe.com/products/air/
[36] Ludwig Adrian, AIR passes 100 million Installs (2009)
[37] See the Silverlight product page on, http://www.microsoft.com/silverlight/
[38] See the Silverlight community, http://www.silverlight.net/
[39] For example, Shinedraw, Flash vs Silverlight fps Meter Stress Test (2008)

Due to the high frequency of requirement changes it's undoable to make a detailed up to date comparison. I suggest Googling "Silverlight vs. Flash". Also a good comparison can be found at Smashing Magazine[41], for example.

5.2.3 Google Web Toolkit

Google® Web Toolkit[42] or Google GWT is an open source Java framework that enables developers to design JavaScript applications and build them in Java. Version 1.0 was launched by Google in 2006.[43] Developers now can build rich applications in Java, which takes care of a stable foundation and then the code can be ran through the GWT compiler which in turn produces neat and browser ready JavaScript.

The entire application is built in Java, and therefore relies on Java development tools. The compiler library includes AJAX (see below) frameworks like jQuery.

5.2.4 AJAX

AJAX stands for Asynchronous JavaScript and XML. This is not a manufacturer dependant technology, but it consists of a group interrelated web-development techniques executed on the clientside in order to create a rich user experience and interactive web applications.[44]

It should be quite clear that the main techniques involved are JavaScript and XML. The reason it's called asynchronous is because of the request to the web-server which is (or can be) done Asynchronous, that is "later", after loading the page. However the requests do not need to be asynchronous.

There are multiple AJAX-frameworks like YUI, jQuery and MooTools. You'll find enough comparisons of the different frameworks on the Web.[45] They basically all do the same:

1. Allow developers to do more with less code
2. Simplify coding so that it becomes within fingertips of designers
3. Enrich the user experience by smoothing and animating the interactions the user has with the interface

[40] See RIAstats, http://www.riastats.com/
[41] Alam Muhammad Usama, Flash vs. Silverlight: What Suits Your Needs Best? (2009)
[42] Google Web Toolkit, http://code.google.com/intl/nl-NL/webtoolkit/
[43] Wikipedia, http://en.wikipedia.org/wiki/Google_Web_Toolkit
[44] Snook Jonathan, What is AJAX? (2005
[45] Like this one on Wikipedia: http://en.wikipedia.org/wiki/Comparison_of_JavaScript_frameworks

5.2.5 HTML 5

HTML 5 is not so much a specialized technology, but a currently under development HTML standard. Because of the clean markup and added tags, for instance the (widely known) video tag which allows video to be played without needing external plug-ins, it's said that HTML 5 will seamlessly integrate with the above mentioned AJAX frameworks and other to-be developed techniques.

Especially since the 2nd quarter of 2010 there is much media noise stating that HTML 5 is going to bring an end to Flash and Silverlight basically because "it does the same, without plug-in(s)".[46] This in particular became more widely spread when Apple opened the fire on Adobe whilst defending themselves that the popular iPhone™ had no support for Flash (nor Silverlight).[47]

Let's put some things straight here, because I think there's a tendency of copy-paste behavior among journalists concerning the matter and they were not always thinking further when writing about HTML 5 replacing Flash. The suggestion of HTML 5 totally replacing Flash (and Silverlight) at first looks true, but that is only if HTML 5 is considered the HTML 5 video tag. If someone looks further he or she might have to ponder about a few things.

HTML 5 is, as said, a standard under development and should only be finished in 2022![48] The specification[49] is far wider than the video tag only. Add to this that there are today no comparable tools for visual designers by which they are able to express their creativity in such splendor.

Other advantages of HTML 5 over Flash, like performance are not always better and depend on development quality. But with Flash products, one has an easier job avoiding performance issues.

So is HTML 5 a better solution than working with plug-ins? Yes! Will it replace Flash (or same kind technologies) within the first 5 years? No! Others think the same about this matter.[50]

[46] Huff Bex, HTML 5 Versus Flash/Flex (2009)
[47] Jade Charles, Steve Jobs: Thoughts on Flash (2010)
[48] James Justin, HTML 5 Editor Ian Hickson discusses features, pain points, adoption rate, and more (2008)
[49] See the HTML 5 specification, http://dev.w3.org/html5/spec/Overview.html
[50] Goldman Jake, Will HTML 5 Replace Flash in the next 5 years? (2010)

5.2.6 Apple Gianduia

Many software techniques for RIA are now arising. We should lastly mention Apple Gianduia, introduced at the WWDC June 2009. Just like the Java based Cappuccino, Gianduia obtains a Cocoa-inspired name (Cocoa is itself a Java-inspired name) to describe its role as a way for Cocoa developers to bring their skills to rich online applications built using web standards, without the need for a proprietary web plug-in (like Flash or Silverlight).[51]

[51] Dilger Daniel Eran, Apple developing Flash alternative named Gianduia (2010)

5.3 Technology - Hardware

In addition to the software, let's write a piece about hardware (technology) that has been very helpful at the development of RIA. Only one big concept is explained here, for further trends and changes concerning hardware, we refer to paragraph 7.1.1.

5.3.1 Cloud computing

A technology that's a great help in advancing RIA's is cloud computing. Commercially sold software based on cloud computing is often called Software as a Service (SaaS), where business-models allow clients often to subscribe monthly to an online web-product that has the same quality and user-experience as a desktop application. The rise of this new business-form is a trend which is part of the answer to our question "Why RIA?" and we will describe this change in chapter 7, let us here explain the hardware part of the technology.

The easiest way to explain cloud computing is by means of the picture in Figure 1. The term "cloud" becomes now clear, it should be approached as a metaphor to the Internet.[52]

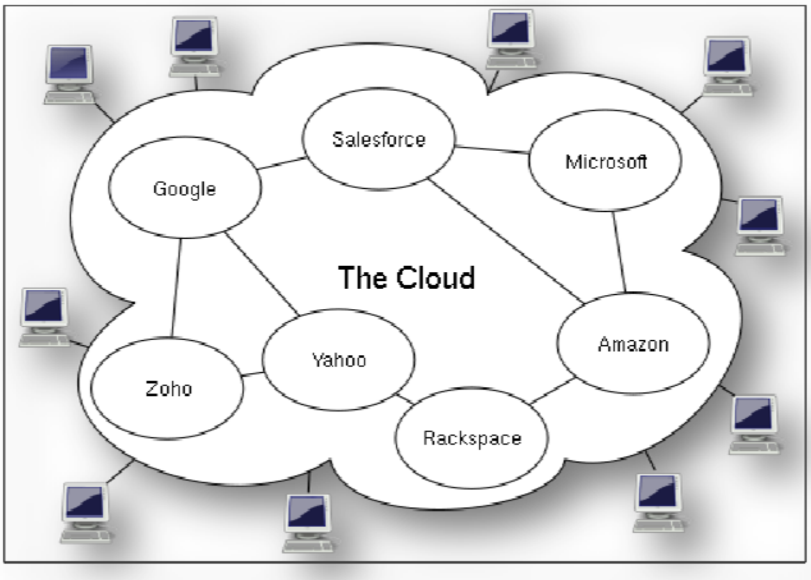

Figure 1: Cloud computing as a metaphor

[52] Berkun Scott, Cloud computing is a bad metaphor (2010)

Cloud computing describes a new delivery model for IT-services based on the Internet, and in general involves over-the-Internet provision of dynamically scalable and often virtualized resources.[53] This frequently takes the form of web-based tools or applications that users can access and use through a web browser as if it were a program installed locally on their own computer.[54]

Common cloud computing providers deliver regular business applications online that are accessed through a Web browser, while the software and data are stored on servers. A facet of main importance in cloud computing is the creation of a user-preferred experience and customization of the interface and application.

Most cloud computing infrastructures consist of services delivered through common centers and built on servers. The clouds often emerge as single access-points for all consumers' computing needs.

The major cloud service providers include Microsoft, Salesforce, Skytap, HP, IBM, Amazon and Google.

[53] The Economist, Cloud computing: clash of the clouds (2009)

[54] Cloud computing defined, http://www.cloudcomputingdefined.com/

6 The EORIA-model

In my opinion, there is much more to say about RIA's. They are an exciting new phenomena that mark the beginning in a phase of web evolution that sets standards concerning application-building that are there for years to come. Every user interface designer or specialist should profit from the model given in this paragraph by adopting it in its early days.

I will call the model EORIA, which stands for Extendable Open Rich Internet Application -model. Extendable, because due to the continuing change in web evolution, standards may be changed and extended. Open, because it's open to everyone to do that. It really should be a model that works for science as well as on the work floor.

6.1 Defining the model

Let's be more specific on the analogue behavior to standard applications and define some more principles. First let us rehearse our definition given in the previous chapter, then add some properties and specifics of RIA's with which we are going to work in the model.

> *"A Rich Internet Application is a web-based application that entails a rich and responsive user interface which behaves closely as desktop analogs and contains the ability to load partial data, but also to store data if no internet connection is available"*

A Rich Internet Application should:

1. Comply with the given definition
2. Be device independent
3. Be platform independent
4. Be browser independent
5. Be Context-aware
6. Be Person dependent (allow for personalization)

Now we'll use some time to explain the enumeration list above. The first item contains a profound foundation to work with and it should be of no question that we are working upon our definition. The following five items are shown in subsequent paragraphs to illustrate what the consequences are for each of the specific items.

6.1.1 Device independence

As you will see in the next chapter, where we describe trends and changes, the upcoming of the mobile market will boost RIA. In addition to that, we are going to see more (new) devices like set-up boxes, tablets, mini-laptops able to execute RIA's.

Consequences for this item are mainly:

1. Multiple screen resolutions (e.g. now including mobile and TV)
2. Multiple ways of input (e.g. now including touch and gesture)
3. Differing device specification

Let's explain each of them.

6.1.1.1 Screen resolution

Before we cannot see the woods for the trees, because there are so many resolutions let's help ourselves, for the sake of our model, by defining the screen resolutions in classes:

1. Small (for mobile kinds)
2. Medium (laptop, pc and tablet kinds)
3. Large (TV and projector kinds)

Display specifications as view-angle and -distance are then related to those classes, which will result in certain design patterns.

Furthermore, design choices as aspect-ratio and orientation etc. belong to these collections of patterns.

It should not be a good idea, although it's tempting, to set certain x- and y-values for the resolutions belonging to one class here. These are time related and are best set occasionally by the designer.

6.1.1.2 Input types

The designer of the application should be aware of the types of input that are available for that device. Is it done by keyboard, mouse (multi) touch, gesture or voice? Or a combination of these? We should think about the constraints and requirements for these choices.

6.1.1.3 Differing device specification

Last consequence is what we call here "differing device specification". This means that every device the application depends on should be available on the device where it's executed. For example a microphone or camera are today not (yet) available on a TV. Devices that are not connected (or built in) cannot be used.

6.1.2 Platform independence

A RIA should be platform independent, because the application then only brings true honor to its name. This means that, at best, two things can be done:

1. No OS specific interface design and interactions are used
2. All OS specifics are known and supported
   The application detects underlying OS and makes the application behave coherent for the user.

We should clearly state that there also could be a third option when there is no underlying OS with user interface. For example when displaying the application on a TV running on an interactive media box.

I think that platform independence is so important that "web-based application" as described in the definition should imply it.

6.1.3 Browser independence

Cross browser support should be part of a RIA. For most user interface designers for the Web, this equals to browser-compatibility. So this should be nothing new.[55] Although an important thing to note is that the amount of browsers is most certainly, going to explode due to built in browsers on mobile phones, set-up boxes and game-computers etc. [56]

Another new item to pay attention to is that the application could have a state that is browserless. Great examples are the applications now being designed for Apples iPhone™ and Google's Android™.[57]

6.1.4 Context-awareness

A lot is being written about context-awareness.[58] It basically means that the application "knows" its context and thereby predicts what the user wants mainly through a *knowledge base* (KB) and learning. A rule in this KB could be "only show me the weather information if it is raining" or "if my friends are nearby, show their pictures".

We have a basic form of context-awareness in use nowadays on a large scale: location services on our mobile phone (when GPS is supported). Applications can determine (if all well after our permission) our exact location and retrieve relevant data for us.

[55] Wikipedia, http://en.wikipedia.org/wiki/Cross-browser
[56] Wikipedia, http://en.wikipedia.org/wiki/Comparison_of_web_browsers
[57] See the App Store on the Web, http://www.apple.com/iphone/apps-for-iphone/
[58] Zimmermann Andreas (et al.), Personalization and Context Management (2005)

The concept of context-awareness is believed to evolve much more, especially in the field of *Smart Homes* (SH).[59] When going deeper into the field of context-awareness you are to touch Artificial Intelligence and Philosophy. Interesting and challenging research is done in this matter.[60]

What are the results of context-awareness for us? In the first place we should notice that much more system events are going to be fired (in the next chapter, we talk a little bit on events), requiring more complex interactions. For example, when a user is carrying out a task and a rule in the KB ensures to change the interface, these events should be handled. The application should ask the user what to do (requiring back and forth interaction) or the event should be quietly handled in the background, leaving open the possibility to still switch to that new state after the user has finished his task.

In the second place, concerning user interface design, more patterns should be thought of. For example, in a fictive news-application, in one occasion five user interface components are used to display weather information, in a second occasion three components are used. This is based upon the context the application has been being placed in. This issue is tightly related to personalization, where the same issues take place and user interface designers have to think about multiple views for their application.

6.1.5 Personalization

Personalization is the last item we pay attention to. We should notice that many people, even within a family, use the same application, but have different needs and preferences. For convenience, personalization is using technology to accommodate the differences between individuals[61] and has been a great challenge for user interface designers.[62]

As stated earlier, this multiplies the number of design patterns in use. There are now more than one ways to design the same application. This means we should be thinking of multiple lay-outs and corresponding, well-designed interactions. Just as we've learned a long time ago.[63]

A last thing to note is that context-awareness can be person (user) dependant.

[59] See stichting Smart Homes, http://www.smart-homes.nl/

[60] Kaswar Fahim (et al.), A portable toolkit for supporting end-user personalization and control in context-aware applications (2009)

[61] Wikipedia, http://en.wikipedia.org/wiki/Personalization

[62] Kramer Joseph (et.al), A user-centered design approach to personalization (2000)

[63] Koch Nora, Rossi Gustavo, Patterns for Adaptive Web Applications (2002)

6.2 The finished EORIA-model

Now that we've seen the consequences for each of the properties, we are ready to construct our model. Shown in Figure 2 you will see our final model. On the next page we will explain the figure and show how to use it.

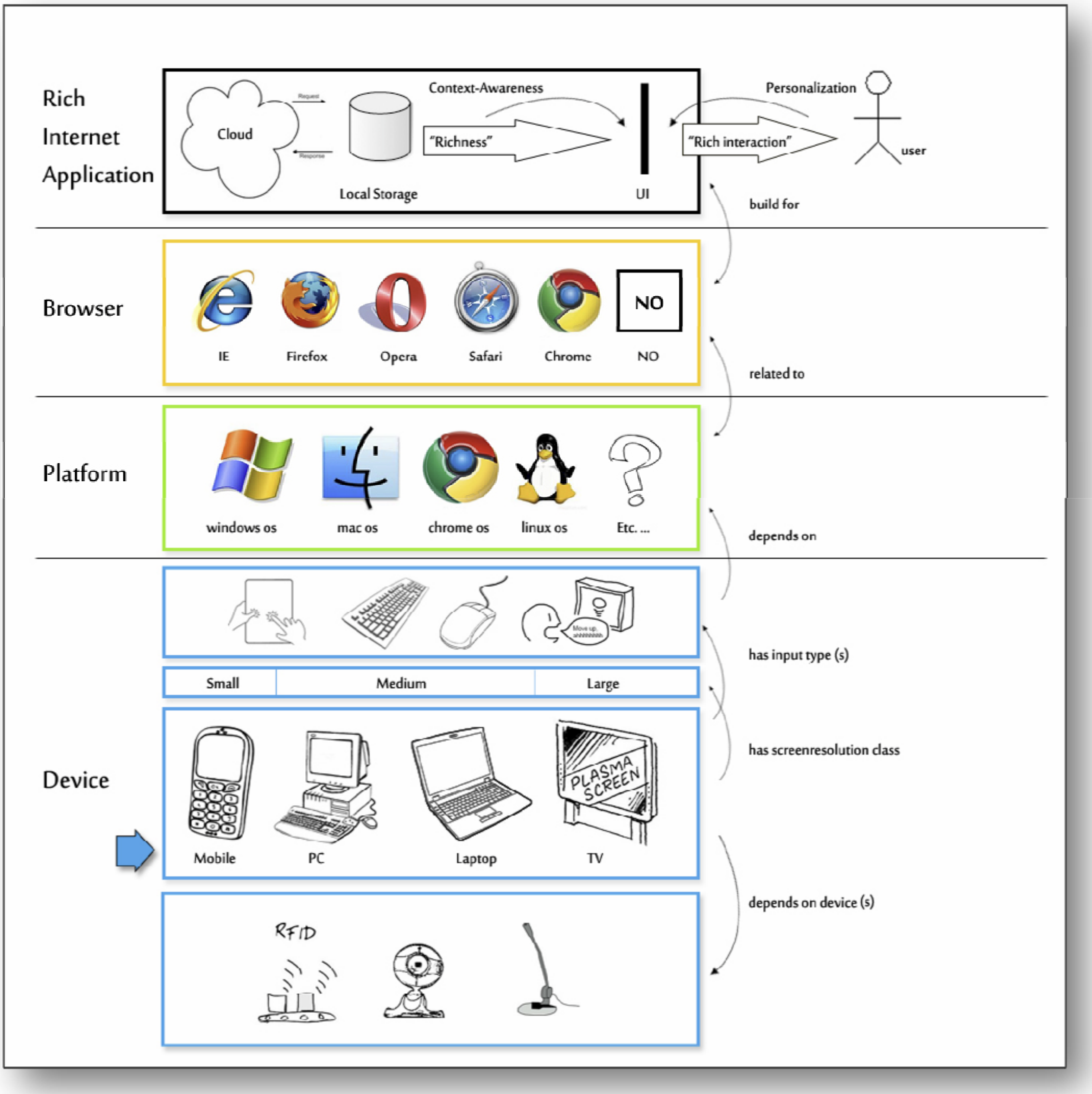

Figure 2: The EORIA-model

6.2.1 Explanation of the EORIA-model

Clarification of the model is as follows:

It consists of 4 layers (e.g. "platform" is on top of "device"). Each layer has been built on the previous layer. You could say that for each application design each layer is dependent on the previous layer. The small arrows show the relation between the (sub) layer (s) and their corresponding names next to them.

The model as shown in Figure 2 clearly illustrates all elements user interface designers should take into account. The EORIA-model contains a clear overview to overcome design issues and helps by making difficult design choices.

6.2.2 How to use the EORIA-model

The model can be used in multiple ways:

The basic way of usage is to start at the thick blue arrow and choose a (one) device to design for. Then the designer goes down to think of the used devices and he can make some notes on how the application should behave if these device extensions (which makes the device different from another) were not present.

Then he goes up and sees the display resolution class he is bound to for designing. He's now able to choose patterns for designing according to that display class.

Next, the designer can do the same thing with the input type(s), he would be using. He can think of patterns belonging to the input types, but also what the fallback behavior of the application should be if one, or more, input type(s) is (are) not present.

Now, the designer can go to the next layer and think of the behavior of the application in relation to the platforms he's going to design for. As said in 6.1.2, he can make choices on how the user interface should behave according to the OS.

The next layer tells the designer to think of the browsers he should accommodate (or none if using the browserless state). Again, he can now look for browser related design issues and think of the best design patterns needed. As mentioned in 6.1.3 all browser compatibility problems that existed with traditional web applications should still be thought of.

Now that the designer has been going through all layers, the last layer is where the application is really to be designed. It tells the designer to think of elements within the interface and how they are influenced by context-awareness (6.1.4) and personalization (6.1.5) Of course, for both, if present in the to-be designed application. The model clearly shows how context awareness and personalization are going to influence the user interface and the designer is alerted to take note of these influences.

Furthermore, the connection to the internet (or not) must be taken into account and also the rich interaction and how this is realized.

When the designer has finished going through the model, he has made profound design decisions and is ready to build the actual user interface (and its interactions), see chapter 8 for a focus on the user interface and the design patterns.

Another way to use the model is when the designer draws a red line from the bottom to the top, through all elements he thinks he's going (or required) to use. He has now a one eye-catch-overview-chart of many of the design choices to be made.

6.3 Conclusion

We have introduced a new model for designers to easily create RIA's and shown how to use it. It's a quite abstract and basic model which should be of great help when designing a RIA.

Nevertheless, we should mention that it may be extended and it's open to everyone to do so. For example, we could imagine that screen resolution classes are going to be changed or added. Everyone is free to make the model suit his or her need. That is part of the open character of the model and should also be its power.

From now on we should try to make a clear separation between, what we will call "traditional web applications" and RIA's as defined here complying with the EORIA-model. In the chapters to come we will work with this model.

7 Trends & Changes

In this chapter we discuss more of the background in the field, starting with an analysis of the trends and changes that bring forth this time period of RIA's. We'll illustrate why RIA's exist and we lay a profound foundation for understanding the change in web evolution so that one could easily predict changes to-be.

Secondly, we will make a comparison (focused on user interfaces) between traditional web applications and RIA's to accentuate differences and design challenges.

7.1 Trends

Now we should have some time to analyze and discuss some trends in this phase of web evolution that all together create an extremely suitable situation for the presence of RIA's as defined in the EORIA-model.

The trends/changes described in this chapter should illustrate that these are points to ponder and how they influence user interface design(ers), thus what the related consequences are.

The trends/changes are for convenience divided into three groups: hardware, software, users.

7.1.1 Hardware trends

Some converging hardware trends are the partial cause of why RIA's exist. Let's take a look at them here.

7.1.1.1 Internet availability

In addition to the world wide growing internet availability[64], the world is now quickly being covered with mobile internet connections. This means that chances increase that users always have access to the Internet. Although RIA's should have the possibility to work temporarily in off-line mode, most of the time they should require an Internet-connection.

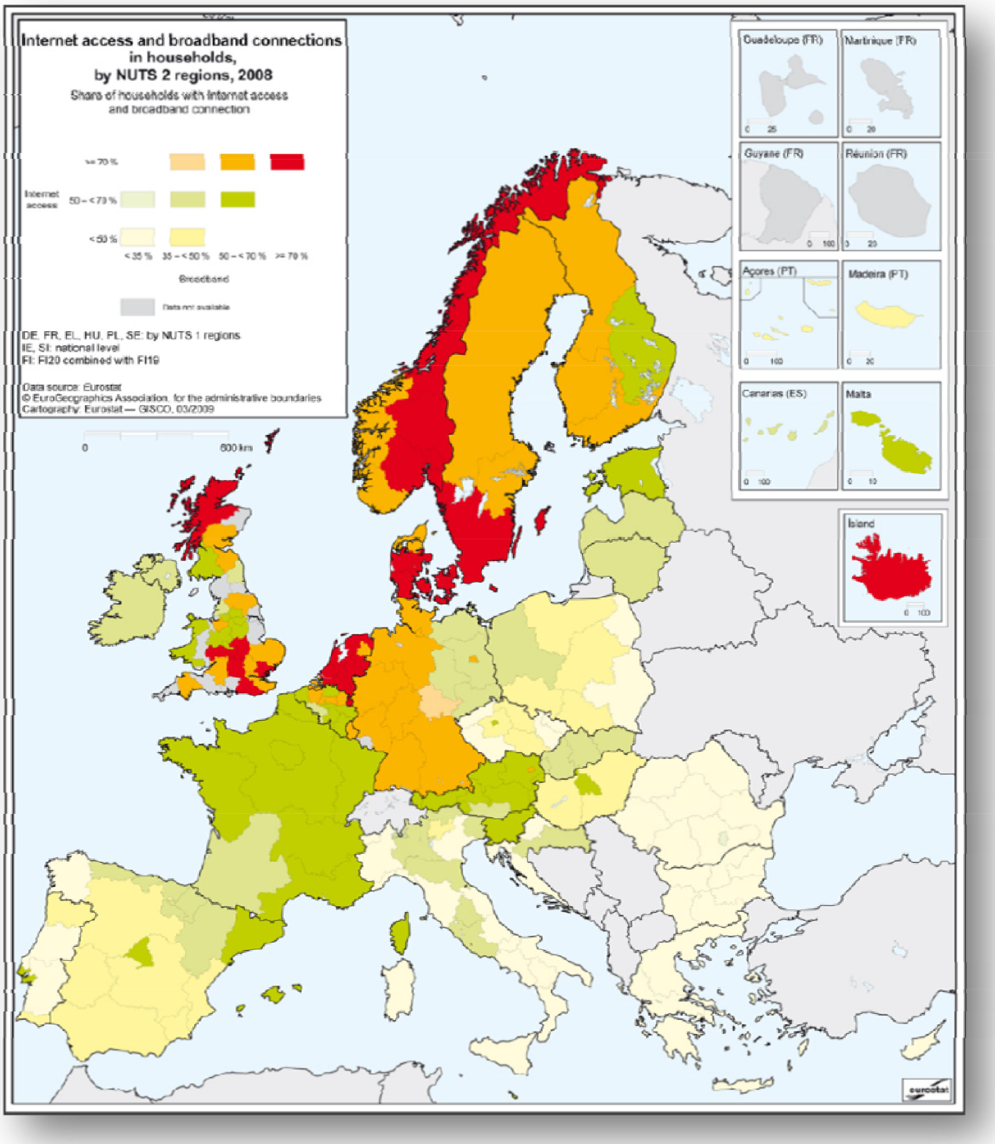

Figure 3: Internet access and broadband connections in households in Europe

[64] Eurostat, Information society statistics at regional level (2009)

7.1.1.2 Higher bandwidth

Higher bandwidth goes together with the previous mentioned characteristic. At the moment of writing, average speed of broadband internet in the Netherlands is 5,7 Mbps[65]. Average speed for mobile is worldwide, just below 7,2 Mbps. High speed and reliable internet transmissions can contribute to smooth working RIA user interface interactions. Furthermore for applications using video, or large-size media files, a higher and stable bandwidth is required to maintain *Quality of Service* (QoS).

7.1.1.3 Advancing hardware

Advancing hardware includes more graphical performance and processor power for PC's, but also the arise of touch tablets, all properties that RIA's use gladly. The "Rich" element of the user interface requires mostly processor power on the client-side (within the web-browser).

On the other hand, applications require much less hardware resources when they are executed in the browser, as cloud computing. Thus users have more power available, but use relatively less, for performance.

7.1.1.4 More powerful mobile devices

Mobile devices have become significantly more powerful concerning processor power in the last few years. An average (newly sold) mobile device, fitting in the palm of your hand, can easily beat a powerful game computer.[66] Add to this that these devices have internet connection everywhere.

7.1.1.5 Web on TV

Another trend we are seeing is the one of television with built in internet connections.[67] Support for a new web language, widget support etc. Also more set-up boxes support web applications.[68] Furthermore there is an *ARM* processor[69] being able to run Adobe Flash/Flex application that is being implemented within many of these set-up boxes (and mobile devices).

[65] Akamai, The State of The Internet Q4 - 2009 (2009)
[66] Karl B., Telltale employee says iPhone "much more powerful" than Wii (2009)
[67] Like the Philips Aurea, http://www.philips.nl/c/televisie/aurea-40-inch-1080p-full-hd-digitale-tv-40pfl9904h_12/prd/nl/
[68] The PopBox HD for example, http://www.webtvwire.com/popbox-hd-review/
[69] See http://www.arm.com/about/newsroom/26062.php

7.1.2 Software trends

This paragraph will lead us through the main trends and changes for software (on and for the Web). The premier trend is caused by, an earlier explained hardware technology, cloud computing. This hardware technology leads us toward some trends in software areas.

7.1.2.1 SaaS

Software leading companies like Google and Microsoft have introduced new software models, as described in our paragraph on cloud computing. This goes mainly under the name Software as a Service (SaaS). SaaS is software that is offered as an online service. Users do not buy the software but pay a reasonable amount per month for usage. The provider takes care of installation, maintenance and management. Great advantage is that users do not have to take care of these things and purchase costs are low. For the reseller, this model is pleasing because marketing and sales costs drop greatly.[70] Gartner states that in 2011, 25% of business software will be SaaS.[71]

SaaS requires that the built applications are comparable with desktop applications and at least offer the same stability, which is not (yet) always the case.[72] It's also stated by some that browsers (as for today) are not built to run applications, this is concerning memory- and process-management after reportedly crashed browsers.[73]

Please also note that this trend concerning SaaS fits perfectly in our EORIA-model.

7.1.2.2 Easier software development (authoring) tools

Another trend in software that boosts RIA is easier design. Great example is the earlier mentioned Flash Catalyst. But also widely adopted techniques as code hinting, built-in help functionality, frameworks and extended animation libraries are part of these development tools that make building RIA's easier.

[70] Source SaaS.nl, http://www.saas.nl

[71] Desisto Robert P. (et.al), SaaS Delivery Challenges On-Premise Software (2006)

[72] Staten James, Is Cloud Computing Ready For The Enterprise? (2008)

[73] Boutin Paul, Cloudy Judgment (2008)

7.1.3 User (needs) change

With the turn-out of Web 2.0 we could clearly discover and analyze the change in user-behaviors and needs while using the Web. Much research has been done in this field.[74][75]

Most literature is about the increasing alter-ego which exists through the intercommunion of supply and demand of user-generated-content (ugc). We want to consume exactly what we desire at our own preferred times. The term "social", when talking about Web 2.0, is often proposed as an illogicality and the truly social part actually seems to achieve the opposite. Web 2.0 has often been considered the cause for hardening our society.[76] The Dutch Queen stated this also in her yearly talk addressing the Nation.[77]

We shall leave that discussion for others and remain focused to user interfaces. Are there any changes explicitly for user interface design? Maybe these are correlated with the phenomena mentioned in the previous paragraph.

STATEMENT 1
When technology innovates and/or advances and it reaches the fingertips of users, a greater demand originates to this advanced technology when offered at the same (or lower) cost as other (previous) solutions.

STATEMENT 2
Innovation and advancing technology equals performance (speed), usability (ease) and flawlessness.

STATEMENT 3
Let's rehearse what users truly want (need) in a nutshell based upon literature sources as Lightner (2003) and Constantine (1995) (et.al.):

- Speed
- Ease (of use)
- Flawlessness (of applications)

Logically, statement 3 is included in 2 (I abstracted the literature for you a bit). Statement 2 says that innovation and advancing technology imply statement 3. If statement 1 and statement 2 are true, we can make up the conclusion based upon these premises that the users' need is a thriving force for this Web evolution towards RIA's as we describe them.

[74] Daugherty Terry (et.al), Exploring Consumer Motivations For Creating User-Generated Content (2008)
[75] Cheong Hyuk Jun, Morrison, Margaret A., Consumers' Reliance on Product Information and Recommendations Found in UGC (2008)
[76] Tomesen Remco, De Pers, 'Web 2.0 is zelfbevrediging' (2007)
[77] Christmas speech Queen Beatrix, 25 December 2009, See: http://www.volkskrant.nl/binnenland/article1331555.ece/

7.2 Traditional versus RIA interface design

Since we're bringing focus into user interfaces, let's compare the user interfaces of traditional web applications and RIA's in this chapter and illustrate the complements each time.

7.2.1 Browserless applications

The first thing we should notice when approaching a RIA according to our model is that there could be an application state that is browserless, being not the case with traditional web applications. The application is no longer in the safe environment of the browser and cannot use its functions or user interface. When the designer of the application intends to use this functionality, he should incorporate these in his interface design. Why don't we have a more detailed look at what the consequences are when abandoning the "known" design within the browser.

The consequences when leaving the browser-interface are:
Solution (recommendation) in italics.

1. **No back button**
   The back button is the most used button in the web-browser interface, recommended read[78]. Studies show that 42% of the actions that a web browser user carries out are done with the back button.[79] My earlier study also shows that the location of the back button is exactly the right one, that's that users don't expect another place for it.
   The application should be designed in such a way that:
      i. *Users know where they are at all times.*
      ii. *The back button behavior and visual design is consistent throughout the entire application.*
      iii. *Users always have the possibility to go back.*
      iv. *Navigation back and forth should feel "natural" and perform browser-like.*
2. **No other navigational elements**
   All actions that a user would undertake concerning navigation will be different and should be altered. This will include the back button (mentioned especially above, because of it's importance) the forward button and address bar, but also there must be thought clearly about linking to external pages.
   The forward button has to be considered (solution just like in 1) and the designer should note that, if the application doesn't offer an address bar, the user has to abandon the application if he/she want to type in an URL. Furthermore, the application needs to include browsing history in the design and if preferred favorites. These should also be confirming iv. (above).

[78] Vermeulen W., Boer den W., De web browser interface - Is een beter design mogelijk? (2008)
[79] Cockburn A. (et al.), Pushing back: evaluating a new behavior for the back and forward buttons in web browsers (2002)

3. **External pages**

   Also, when linking to external pages, one must realize that some of these pages do rely on the web browser interface. When opening a link from within the RIA, most architectural designs will cause to open the standard browser from the operating system.

   There must be thought clearly about what happens when external urls are called or even embedded. Can you create a clear and unique browser experience within your application? In most cases, it will be better to handle this via the OS, however then you break the experience of your application.

4. **Embedded objects and plug-ins behavior is different**

   Plug-ins on which you (as a designer) would rely on may not be available and behave differently.

   Controls for the concerning plug-ins should be incorporated in the design or a well-considered design choice for handling the plug-in outside of the application must be made.

7.2.2 Device

According to our model a RIA should be able to be executed on different types of devices. Compared to traditional web applications the differences (and points of attention) are:

1. The different input types
2. Dependant devices

   (screen resolutions are discussed in the next paragraph)

Good and helpful sources on handling multiple input (device) types are for example Lu Xudong (2007)[80] and Paterno (2005)[81].

As for the second, one should not apply design patterns for devices that are not there (e.g. microphone, camera) and come up with creative ideas leaving a clean visual design and well performing application. This of course is also true for traditional web applications.

[80] Lu Xudong (et al.), A New Approach for Multi-device User Interface Design (2007)

[81] Paterno Fabio, Santoro Carmen, Designing Multi-Device Interactive Services through Multiple Abstraction Levels (2005)

7.2.3 Screen resolution

Again let's use our model to see the difference with traditional web applications. RIA as defined in our model (chapter 6) should be capable of handling three classes of resolutions: mobile (small), computer (medium), and TV (large) (the small and large class display resolutions are new).

The basic lessons we've always learned, for example by Brown,[82] about resolutions for web-designers, concerning usability and screen-resolution still apply. Think of rules like: leave enough whitespace, keep a maximum width for text (to benefit readability) and acceptance of minimum resolutions of the users of your application's target audience. Many rules can be found for example at Nielens' website[83].

When comparing RIA's to traditional web applications, attention must especially be paid to the small and large display class.

For mobile, good usable patterns are quickly arising and can be adopted.[84] For TV (large screens) however not so much. Research has been done[85], but it has hardly been applied.

There is another issue, that should be paid attention to: more people will be sharing one screen while using that application. This point is of course of another kind, but I think it's worth mentioning.[86]

[82] Brown C. Marlin, Human-Computer Interface (1999)
[83] Nielsen J., Screen Resolution and Page Layout (2006)
[84] Nudelman Greg, Design Patterns for Mobile Faceted Search (2010)
[85] Robertson George, Tan Desney (et al.), Large Display User Experience (2005)
[86] Lekakos George (et al.), Information systems in the living room: A case study of personalized interactive TV design (2001)

8 Design patterns for RIA user interfaces

We'll use this chapter to deconstruct a RIA user interface to get some more understanding of the elements involved when constructing a RIA user interface. This will also help to understand design patterns.

The design pattern(s) (collections) we will show in 8.2 will be designed for RIA's as well as traditional ones that can be adopted. Furthermore, adaptation of current used patterns is what we will take a more detailled look at to see if problems arise when we do so.

8.1 (De) constructing the RIA user interface

Before we assemble design patterns for RIA, let's break apart the user interface of a RIA and have a closer look at specifications, requirements, behaviors, feedback, events and contexts. The first three just mentioned are grouped together and discussed in the first subsequent paragraph because they all concern visual design. The other three are each discussed in the paragraphs thereafter.

See an overview of this deconstruction of the RIA user interface in Figure 4 at the end of 8.1.

8.1.1 RIA Interface components

Each RIA user interface can be defined to be a set of components. It consists of (0-N):

1. Input components
2. Button components
3. Display components (all other visual components)

Think of specifications as, size, width, height, position, x, y, color, font, alignment, border, opacity etc. Everything that can be visually seen can be included in the specification of these components. All of these components are placed in a context (see 8.1.4) that may change the specifications.

When talking about requirements, the same patterns as in traditional web applications apply. Think of "text should fit in textbox" for input components and "a user should know that an item is clickable (touchable)" for buttons.

Visual design of behaviors, like shadow when moving over an object that's clickable should be consistent with traditional web applications.

8.1.2 Feedback

The components can provide feedback which we'll, for good understanding, split apart in four types of feedback:

1. **Visual**
   All feedback visible on the screen, through icons, animation and visual design (changes).
   (or a combination of these)
2. **Audible**
   All feedback given through sound.
3. **Textual**
   All feedback that contains text.
4. **Haptic**
   All feedback that's given through touch.
   Most devices are not capable of giving haptic feedback nowadays, but it's seen in the mobile market a lot

We should also note that feedback mostly consists of a combination of the above four.[87]

8.1.3 Events

We should also mention that the feedback is generated through events. We should distinguish 2 types of events:

1. System events (errors and messages)
2. User events

(These are derived from the Flex framework[88], but I think that this is the best approach or perspective when talking about events). One should now see (and learn) that, in this approach, a RIA user interface is always event driven.

For a designer understanding events is important because they make him think when these (can) happen and how they can interfere in designed interactions.

[87] Schneiderman Ben, Plaisant Catherine, Designing the User Interface: Strategies for Effective Human-Computer Interaction 4th Edition (2004)

[88] See "About Events", http://livedocs.adobe.com/flex/3/html/help.html?content=events_02.html

8.1.4 Contexts

When studying software engineering and in particular user centered design approaches, application states nowadays become more usual to work with while designing an application. This is also absolutely true for the Adobe Flex frame work and Flash Catalyst.

The designer defines states where his application is transformed to. If a user fires an event (for example through a button), that means that the state could change. But also through a, on beforehand programmed, (system) timer state can change. A change in application state implies a change in visual appearance of the application.

Contexts, as described here, are groups of components and are a subcomponent of application states. You could say that it is a part of state on which the designer focuses and applies consistently, specifications, requirements and behaviors (for example a commenting area). Thus, a text field, that is part of our display component can behave differently in multiple contexts.

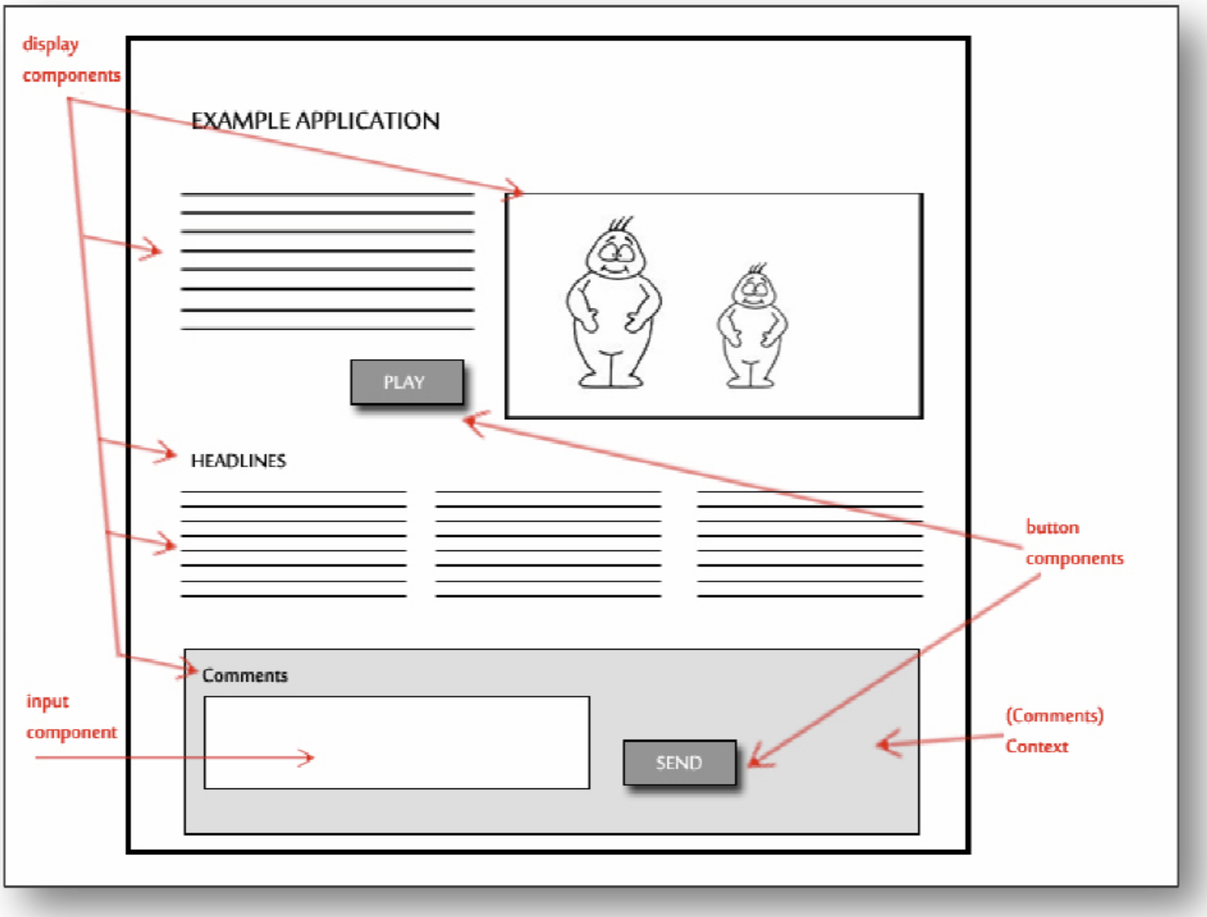

Figure 4: Example application, showing an application state with all described components of a RIA user interface

8.1.5 Wrapping it together

Figure 4 shows a state of an application wherein all components are showed. This illustration should also make it clear that the send button is allowed to have different visual appearance from the play button because it's in another context. The application shows an image belonging to the text, but if the user presses the play button, this image is being replaced with a video component. The application is then in a different state. See Figure 5 for an illustration.

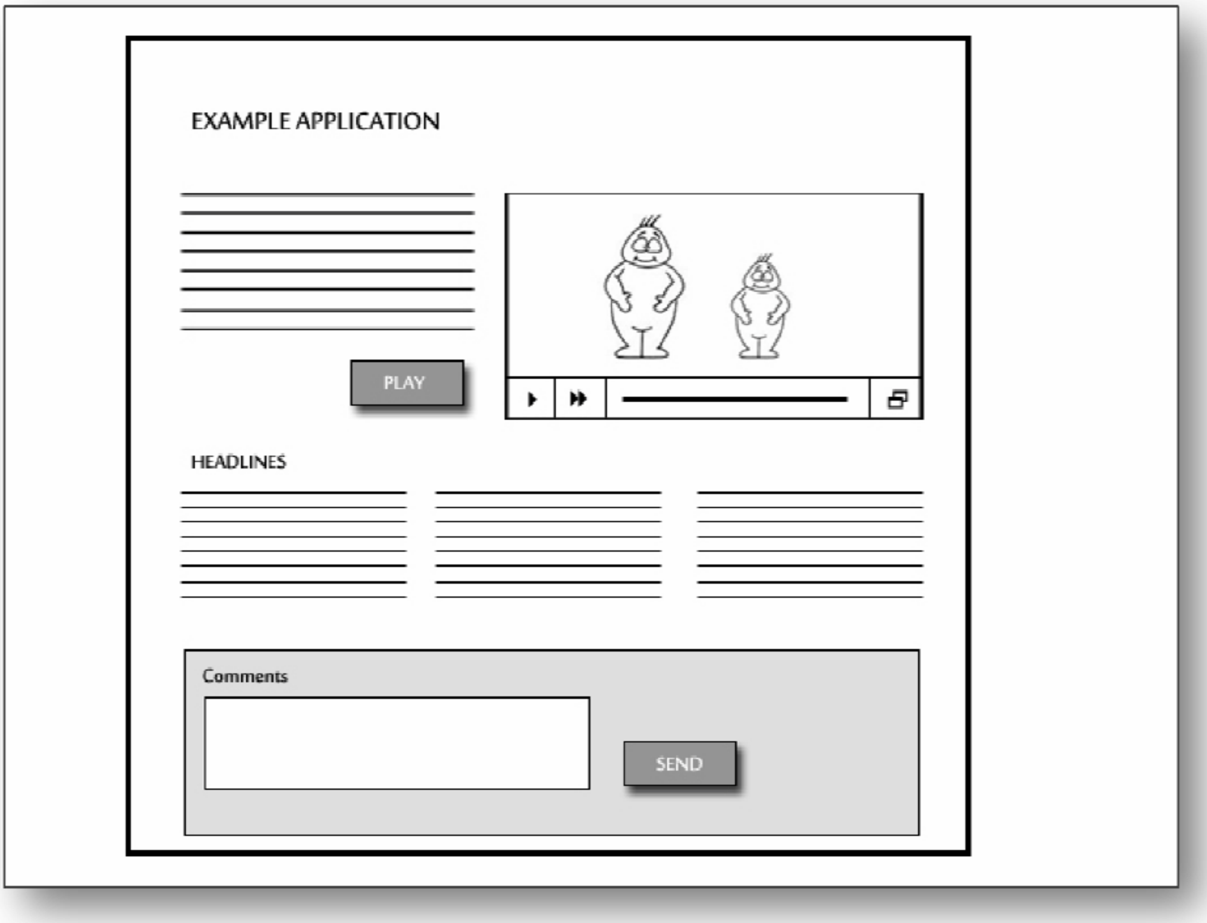

Figure 5: Same application as in previous figure, except in different state

8.2 Design patterns for RIA user interfaces

But before we continue we must clear up some probably originated confusion. We are not talking here about design patterns[89] as in the field of software-engineering[90], but design patterns as described for user interfaces and interaction.

Good. This is actually part-b of the story in this chapter. It would be impossible for us to generate a complete list of design patterns for such a new phenomena as RIA's. Mainly because there's not really one standard like HTML and/or CSS and RIA's are subject to rapid changes.

The many traditional design patterns we know for Web usability were also generated (and researched) over years.[91] Beautiful, but negotiable, examples of design pattern libraries can be found, including Patternry, Endeca, Welie and UIpatterns, in paragraph 13.4 where **we collected all patterns** in this thesis including their corresponding web links.

What we will do however is show a few that are neck stabbing and have some noticeable differences with current patterns. These patterns that are especially designed for rich interfaces, SaaS applications and complex interactions.

Secondly, in 8.2.2, we will see if we can adopt "old" patterns without modification (or maybe they have become due) and if any other issues arise when doing so.

[89] Wikipedia, http://en.wikipedia.org/wiki/Design_pattern_(computer_science)
[90] Van Vliet Hans, Software Engineering - Practice and principles 3rd edition (2008)
[91] Welie Martijn van, van der Veer, Gerrit C. & Eliens Anton, Patterns as Tools for User Interface Design (2000)

8.2.1 Notable design patterns for RIA user interfaces

As said, we are not expecting lots of RIA patterns. Our recommendation is to look mainly to patterns for AJAX and or Rich applications. Bill Scott has held a great talk on the matter.[92]

The book "Designing Web Interfaces"[93], fortuitously authored by the same man, (B. Scott, T. Niel, 2009) has an accompanying website containing multiple patterns for RIA interfaces.[94] Among them are master pages, search results, forms and ports from desktop applications (e.g. spreadsheet). See Figure 6 for examples.

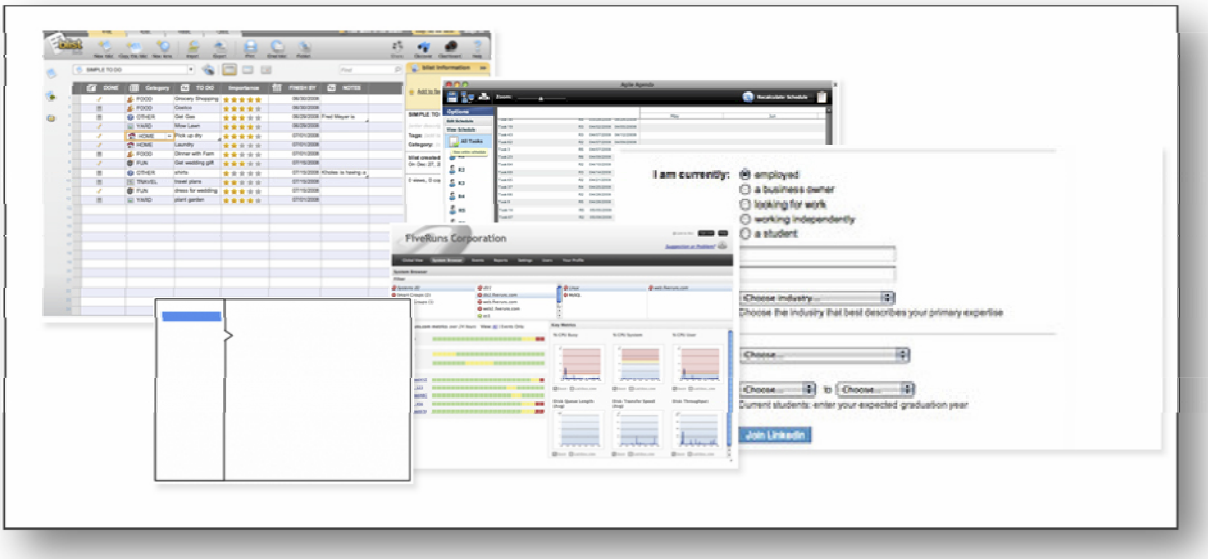

Figure 6: Examples of RIA patterns

Another great place to start is the Yahoo! Design Pattern Library.[95] They include a category "rich interaction" which is excellent for RIA.

Another great book that has as target audience RIA user interface designers is "Designing Interface"(Tidwell, J., 2005) Which contains some special RIA patterns.

Finally, I want to end with two great talks from another person we already discussed. The first one includes short YouTube video's on patterns and also make problems more clear when having a browserless application state. This presentation talks about almost all patterns for rich interaction and makes my adding superfluous. Highly recommended. [96]

[92] Scott B., Designing for AJAX, http://www.slideshare.net/billwscott/designing-for-ajax/
[93] Scott B., Niel Theresa, Designing Web Interfaces (2009)
[94] See, http://designingwebinterfaces.com/designing-web-interfaces-12-screen-patterns
[95] YDPL, http://developer.yahoo.com/ypatterns/everything.html
[96] Niel Theresa, Designing Rich Applications (2008)

The second one is called "RIA Screen layouts." And contains a beautiful and complete overview of patterns for RIA with corresponding live examples. The figures below show examples that should awaken your appetite. Also highly recommended. [97]

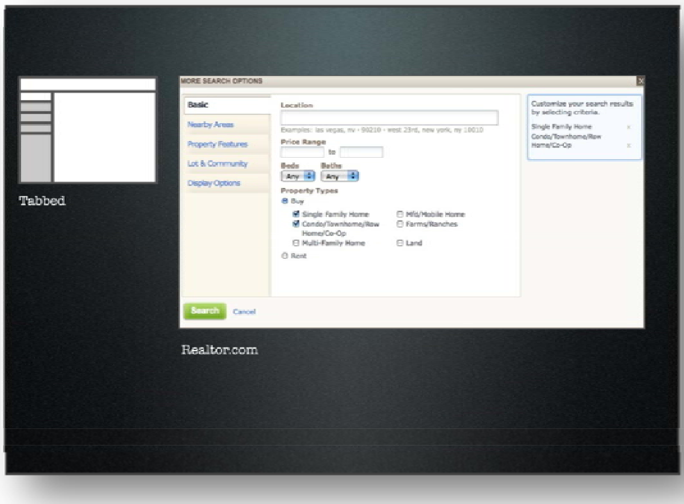

Figure 9: Example of RIA design patterns for Tabs

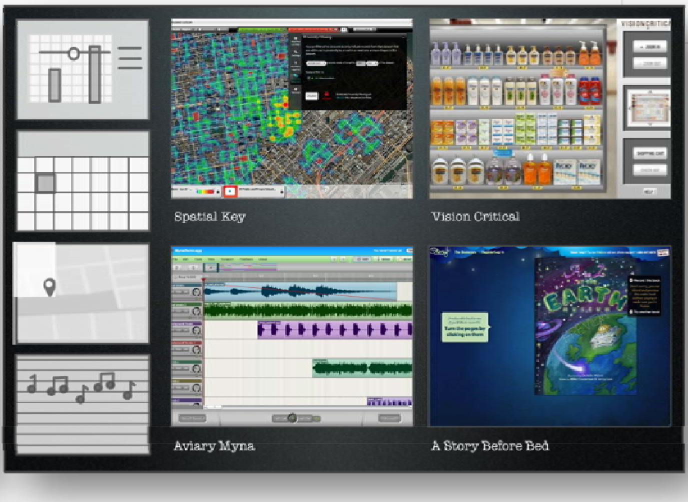

Figure 10: Examples of RIA design patterns for Interactive Design

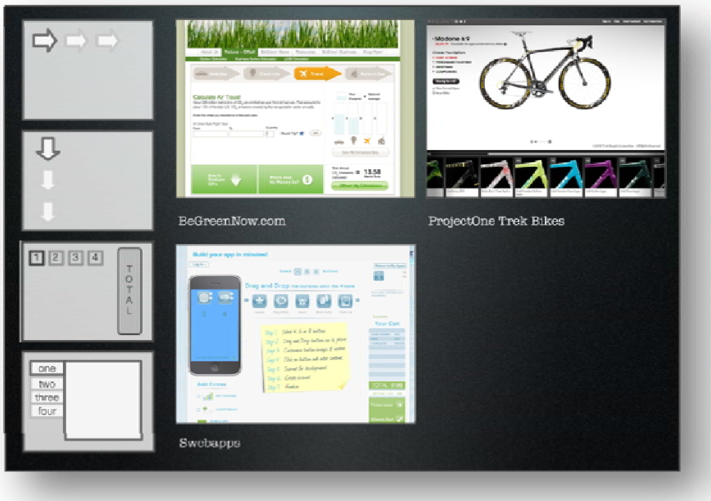

Figure 8: Examples of RIA design patterns for Wizards

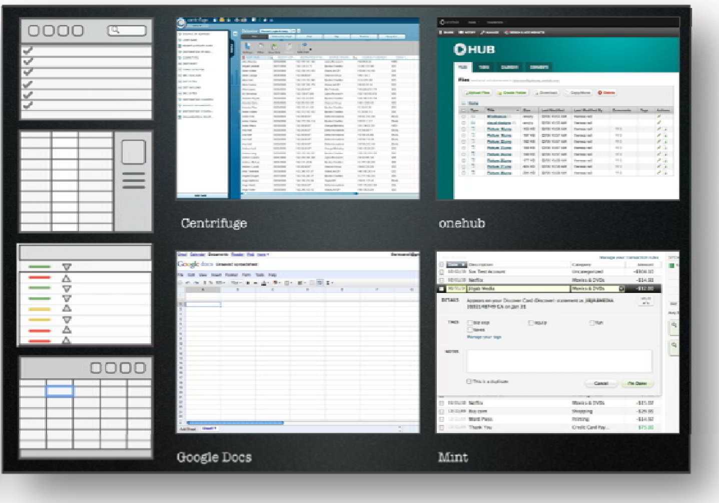

Figure 7: Examples of RIA design patterns for Spreadsheets

[97] Niel Theresa, RIA Screen Layouts (2010)

8.2.2 Adoption of current design patterns

As you could have seen in 8.1 where we (de)constructed a RIA user interface, there should be many lessons we've learned from building traditional web applications that still are valid for RIA's. So we now can say that we can adopt current design patterns and keep on using the beautiful ones given in the introduction of 8.2 (and in 13.4).

RIA's can make good use of existing patterns as long as the designer realizes that more advanced patterns should get priority.

User interface designers should be conscious of the fact that these pattern(s) (libraries) are going to extend and increase as cause of:

1. More complex interactions
2. Browserless application states
3. More types of devices and screen resolutions

9 Case study: Research in practice

Let's see if we could get some findings and results of our preceding literature study in practice. As part of the master accreditation of the author the case study described here is a helpful means in accomplishing this research.

I had the great possibility to join in (and lead) a usability research to a RIA application embedded on a website at a large media company in the Netherlands.

This chapter describes the case study background, challenge and the experimental set-up. For your convenience, the next chapter contains the results and conclusions.

9.1 Background

Before we enter into this case study, we'll discuss a little bit more about the background of the company I worked with and the assignment I carried out.

9.1.1 The company

The company where the case study was carried out is a large Dutch nationwide television broadcaster called Evangelische Omroep (EO). Founded in 1967, the EO has a large experience of being part of the Dutch public broadcasting. Since 1992 (until now), the company holds the A-status (which is member based in the Dutch public broadcasting) and has a considerable impact on the society through radio, television and the Web.[98] Their website is found at http://www.eo.nl.

Figure 11: The EO logo

[98] See http://nl.wikipedia.org/wiki/Evangelische_Omroep

9.1.2 The assignment

In December 2009, the EO, being the first among the Dutch public broadcasters, launched their own web-TV player with to disclose (within the coming years) all their produced content of the last 20 years. As an extra the web-TV player had a built-in web-only channel (wisely called EOxtra).

Because this product is under continual development and still in infancy there was a need for a usability research on this RIA (for more information on the product see below). In short[99], the EO wanted to know whether:

1. They were coursing in the right direction if it came to user friendliness of the disclosure of all their archived content.
2. The design and interaction of the interface satisfied users' needs.

They also added more requirements that needed to be part of the research which have no value for this thesis. However, the results are still included in belonging appendices. Maybe they are of inspiration for someone carrying out research.

9.1.3 The product

Finally, the product which applied to the usability research performed is called (internally) EOx and advertised as "EO web-TV". It's an embedded Silverlight object, measuring 640x360 pixels, placed on the homepage of the EO.

The product, built by Lightmaker[100], Amsterdam, was intentionally developed to launch a web-TV channel and for disclosing of the archive. However, since the EO had no experience in building this sort of applications and requirements weren't clear while developing they immediately admitted that the user interface had become a disaster.

An important notice is that they immediately admitted, the user interface and application structure were spared a lot of well thought-trough time, because other priorities where higher.

Searching and browsing for content, retrieving information about content and controlling the media are all done within the small embedded object. A good place to start a usability mess. See Figure 12 for a picture of the product.

[99] The official description of the assignment can be found in appendix VII
[100] See http://www.lightmaker.com

9.2 The challenge

The challenge we really have for the case study is that we are able to reveal issues we might have overlooked in literature and by being confronted with a real life situation find facts that we must pay more attention to.

We should be able to test some design patterns and design issues, although we should be realizing that there would be many issues in this usability research only applicable to the company and not so much for this thesis.

Another part of the challenge for this case is that we are going to find users reactions to the application and we're able to see whether the trends described in chapter 7 are correct.

Of course all added information we'll discover is invitable.

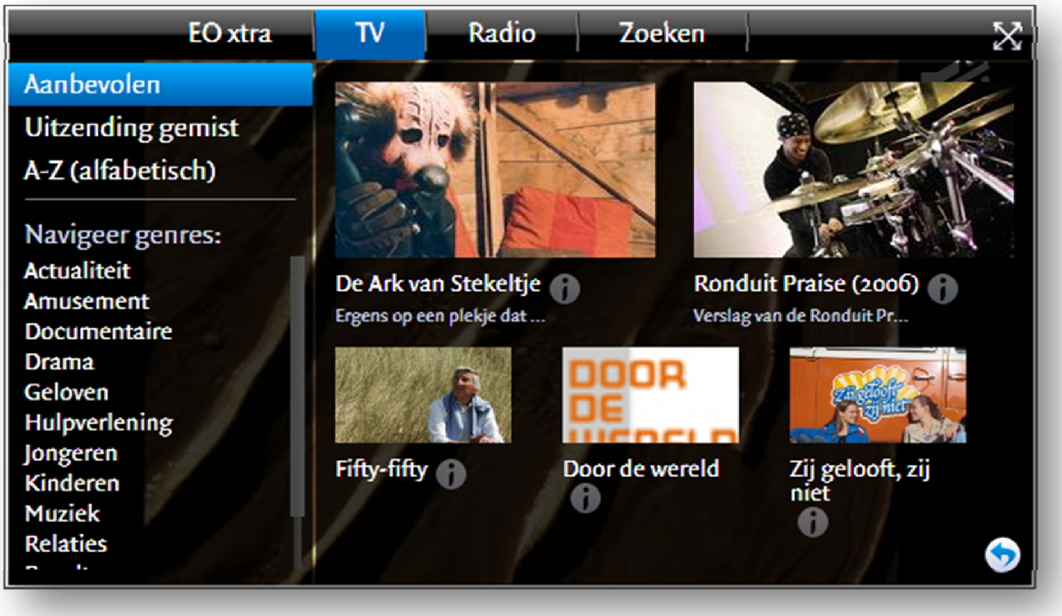

Figure 12: Example of the product (EOx) that was researched

9.3 Experimental design

In this paragraph and the included subparagraphs we will illustrate how the research came into being and discuss the problem statement, conditions, used resources and software.

Please note that the full design of the interview and survey could be found in appendix II.

9.3.1 Problem

The problem stated by the company is that they have no clue how well or bad they are doing concerning their application. Icons, design patterns and application lay-out are not thought of properly and therefore they need a usability research confirming the right parts.

9.3.2 Research design

After discussing with my conductors, the research should consist of interviews and surveys. The interviews should basically be the same as the survey and should serve as a pre-test (see below). The actual research should consist of the survey.

After some basic questions (like age, sex, location, see questions 1 to 23 in appendix II) I had the respondents to carry out multiple scenario's (4) that are chained (which we shall call chained scenarios). In this way there are 4 main subjects(tasks) we can focus on for task analysis, which are carried out after each other so that a real user experience comes to pass.

Each scenario was setup with a special goal and containing a topic:

1. **Scenario 1 - Searching and Browsing**
   First scenario in the chain contains the assignment: "Search for the episode about the disappearing of Joanne Noordink of the broadcasting 'Het zal je maar gebeuren'". The goal of the scenario was to know more about: how respondents search and browse, which buttons they use and why and whether the player behaved as expected. This scenario consists of 5 questions.
2. **Scenario 2 - Information processing**
   The next scenario in line is one that may be equivalent to one task. We ask respondents to play the found episode. Goal of this scenario is to discover how buttons and links within the application are used (read: have it confirmed that it's wrong). This scenario consists of 1 question.

3. **Scenario 3 - Fullscreen**

   The third scenario is one where the respondents are asked to go to fullscreen mode for the video they've just started. Goal is to find out how users are doing that and if everything within the application behaved as expected. Furthermore respondents were being asked if they knew more ways to carry out the same scenario. This scenario consists of 4 questions.

4. **Scenario 4 - Radio**

   Last scenario in the chain tests if respondents are able to execute an easy assignment. The assignment was: Listen to the episode of "man/vrouw radio" of January the 2nd 2010. Respondents were asked to write down every step they carried out. This scenario consist of 1 question.

The topics of the research included:

- Internet usage (in the living room)
- EOx (the product)
- Media usage
- The homepage
- Readability and visual design
- Keyboard and mouse
- The scenario topics described above

The time for one respondent is approximately 15-20 minutes, but the interview might take some longer if additional questions are asked. For more information on how this research was carried out, see 9.3.3 for the research approach.

9.3.3 Research approach

The group of respondents was composed by the department of marketing of the EO and consisted of 2000 people, randomly mixed variables of age, sex and geographical location. All the people agreed to voluntarily participate in surveys of all kinds for the EO.

The head of the internet department signed to send the e-mail I composed (see appendix IV, Dutch) to the whole file with respondents.

Additionally when there should be too few respondents, project leader and conductor advised me to design a banner to place on the homepage. So I did, see Figure 13.

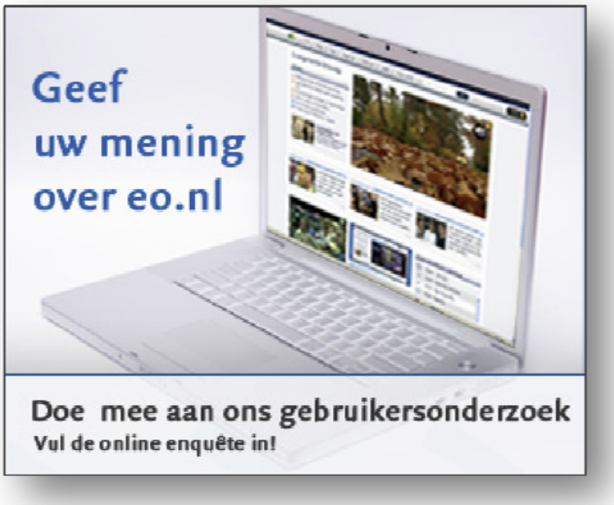
Geef
uw mening
over eo.nl
Doe mee aan ons gebruikersonderzoek
Vul de online enquête in!

Figure 13: Designed banner for attracting respondents

The survey was available for three weeks from 23th of February until the 18th of March. Then the analyses was done for approximately another three weeks. While the research was being carried out, we would keep track of any unexpected results so that we could intervene if necessary.

Now a few words on the used software. All (accompanied) texts were composed with Microsoft Word, the survey was constructed with Examine[101] and the results were analyzed with SPSS[102] and Microsoft Excel (and some common sense).

After the analysis, I have given a presentation for the whole internet department explaining important lessons learnt concerning usability in RIA's. Respondents who had indicated that they wanted to receive more information, received a summary of the results (See appendix V, again this is in Dutch).

[101] See https://examine.vu.nl/
[102] Statistical software, see http://www.spss.com/

9.3.4 Pre-test (interview)

As stated, we use the interviews as pre-test. In this way we have the ability to observe the users' behaviors and ask additional questions when needed. Thus we have the possibility to make changes to the survey. For the interview, we use the Nielsen-5 rule, which basically says that testing 5 people should be enough to produce the biggest flaws in user interface design.[103] In this research we project this rule on the pre-test.

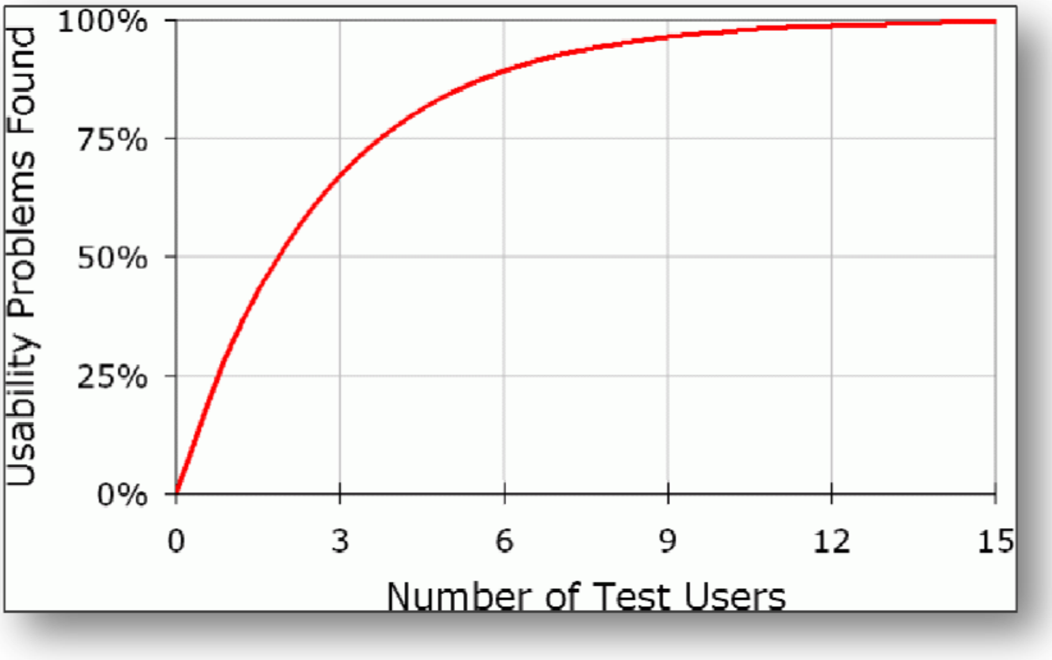

Figure 14: Diminishing returns for usability testing, the curve bends around 5 thus that is the recommended number of participants

The results of the pre-test can (also) be found in the next chapter.

9.3.5 Research (survey)

The survey consisted (just like the interview) of a maximum of 70 questions depending on answers given in earlier questions.

Clear and simple introductionary texts should explain the respondent what he/she could expect. (See the design of the screen texts in appendix VI).

There was an indicator showing how far the respondents were stating that there is light at the end of the tunnel. Not that this research was so annoying, but we should be aware that there are people participating that are not so great computer fans as we are.

[103] Nielsen J., Why you only need to test with 5 Users (2000)

The "tricky" part of this research is that it is carried out wherever the respondent wants, so they could easily be distracted, look for comparable solutions, use the Web to find answers etc. Although we know that this is a though one, we should also see that it is of positive meaning because respondents should do more or less what they are doing in everyday life and secondly, this is of great meaning for the EO because in this way they can test compatibility with multiple browsers and operating systems and how it affects usability.

However the following things have been done to prevent this issue of being a breakdown:

- Introductionary texts clearly explain that respondents should close all applications and stop all other activity.
- Introductionary texts clearly indicate that respondents should open only two windows or tabs; one with the survey (current) and the second one with the application.
- Designing the survey in such a way that the respondent has some amount of freedom in switching from survey to application window.

9.3.6 Final notes

Before handing over to the result let's use this whitespace to make some final notes.

The first one is that the RIA we are about to test is not in agreement with our EORIA-model, nor with our definition. Luckily, some definitions apply, and still there is a good lesson to learn.

10 Case study: Results

See the appendix II. The results are placed in the appendix not to clutter up this thesis, even more because some parts of the usability research were only of interest for the employer.

In this chapter, we'll discuss the results from the pre-test and add some notes to it. Secondly we'll go into the basic results of this case study. After that we'll analyze the results and drilldown into the top-5 issues found. Finally we close this chapter with the conclusion to this case study and see what the added value is, as well for this thesis as for further research.

10.1 Results pre-test

As stated 5 people took the pre-test. No unexpected answers or behaviors were given, so that would be a "go" for the research.

Most remarkably, results were that 2 (out of 5) respondents indicated that they had removed the homepage of the EO as start page because they were having performance issues leading to annoyances. These issues appeared since the introduction of EOx and (of course) signs for a bad (read opposite) user experience.

There were no remarkable comments from respondents on the research which could implicate that the survey would be too difficult to carry out on one's own. On the other hand, there was some concern about whether the respondents wouldn't be too distracted while taking the survey (and that this could not be verified). This was known to us and we have discussed this dilemma in the closing of 9.3.5.

What about the further content of results, there were slight implications that some of the semiotics in the user interface were not clear. But this was not enough to modify the research set-up and/or survey.

We therefore could doubt the Nielsen-5 rule only is good enough if used as pre-test. See paragraph 9.3.2.

10.2 Results Main-Research (survey)

Because the response-ratio was rather low, we used the designed banner (mentioned earlier) to attract an even wider audience, and placed it on the homepage for 3 days. Yet this did not help much. The final response ratio was below 20%. There were 327 respondents, 162 male and 165 female.

Finally there were 537 reactions, but 210 only read the introduction.

As stated, not all of the research applied to this thesis and was only of interest for the EO. However there were some amazing results.

See appendix III for graphs of the basic results. Below are the remarkable facts we can conclude from those graphs.

SEE GRAPH 1-7

10.2.1 Respondents

A large group 88% of respondents was older than 41 years (total sum of three classes). On average the correspondent gives himself an 8 (on a scale of 1-10) concerning involvement to EO. The same applies for computing experience. 3% of respondents had eo.nl as their most used website. The largest group uses the Internet 6-15 hours a week. For a media company it was interesting to know how much usage of their content is consumed via the internet, the largest group (38%) used it less than 5 hours a week, but there are two partitions, combined 32%, that say they never watch TV or listen to the Radio using the Internet.

SEE GRAPH 8-9

10.2.2 Web visits

Frequency of visits to eo.nl is less than once a month for 39% and for 13% its daily. From this group, a relative big part does come to watch TV programs, another main task on the website is looking for information about programs and events.

SEE GRAPH 10

10.2.3 Browser usage

Some things on browser usage. The biggest group (23%) had IE8, the whole IE-family had 56% and the Firefox-family 12%. About 25% of respondents did not know which browser they had.

SEE GRAPH 11-12

10.2.4 Icon recognition

Our first icon (see Figure 15), we asked for a meaning of, was mainly answered right (58%), 17% selected our misleading suggestion, stretch player above the right answer. Very interesting was the open answer (25% of correspondents chose for that). Most of this group expected this icon to mean "move" the player.

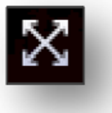

Figure 15: Icon that is used for going to fullscreen

Our second icon (see Figure 16), was answered right by 89% of the respondents, and should confirm existing design patterns.

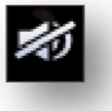

Figure 16: Icon that is used on the button for muting the sound

SEE GRAPH 13-17

10.2.5 EOx

Most people (68%) did not know EOx. From this group 25% said not to be interested and 50% said that they are not willing to watch TV via the computer. Among the respondents that did use EOx was a relatively big group of 26% that used the application daily. Only 11% used the application less than 5 times a month. When watching, most people watch somewhere within 20-30 minutes.

Among the most requested features for the application were "link to program-website", "link to chat", "Care/support after and during programs", and also multiple requests for programs, but that had nothing to do with our research goals.

SEE GRAPH 18-20

10.2.6 RIA in the living room

A little bit more than 50% claimed to have no computer in their living room. The average size of the largest screen they have in the living room is 50-70 cm (44%), second is 70-90 cm (34%). Almost three quarter of the population indicates not to use Internet on their TV, not through their laptop, media center, set-up box or streaming.

For the following results of the scenario's there are no graphs. We again refer to appendix II, where an overview of the questions including the answers is given. All remarkable findings of the scenario's are found below.

10.2.7 Scenario 1 - Search and browse

The average time for this scenario was 20 seconds (long). Most respondents use the search functionality. Respondents searched on program as much as on episode, but also on "her name" (Joanne Noordink). The media player responded for 65% as expected. Silverlight installs go wrong as well as many reported performance problems during this scenario.

(See questions 24 to 28 in appendix II).

10.2.8 Scenario 2 - Information processing

Some 35% users will click the photo, 21% the title to start a video. There is a strong demand to visually represented buttons and links (as expected).

(See question 29 in appendix II).

10.2.9 Scenario 3 - Fullscreen

This assignment was not too difficult, but also not easy. 69% of the respondents uses the button to go to fullscreen instead of choosing one of the other options, 25% does not succeed in getting the application to work fullscreen.

(See questions 30 to 33 in appendix II).

10.2.10 Scenario 4 - Radio

For many people this assignment was too difficult (almost 40%): Listen to the episode of "man/vrouw radio" of January the 2nd 2010.

(See question 34 in appendix II).

10.2.11 General conclusions

We close here with some general conclusions before we proceed analyzing these results given. We must note that not all results are discussed here. We once again refer gladly to appendix II.

Some conclusions on the survey that we want to include here are:

- On average the assignments (scenario's) were experienced as difficult.
- Browsing structure of the application is working as expected.
- Respondent satisfied with usability (7/10), responsiveness, speed by which the streams are loaded.

10.3 Analyses of results

Below we will describe the interesting things we found when analyzing the results given on the previous pages. On the next page, we'll discuss the top-5 issues found while analyzing this case study.

First thing we should notice is the relationship between the given computer experience and the answers given concerning browser choice. One quarter of the respondents did not know what their browser was. So this may be a good reason to think that the actual level of computer experience is a little lower than the stated average 8.

Next, a remarkable finding that the respondents conclude that the application is working as expected, is amazing because the application is from a user perspective not well designed.

Furthermore, the results of application speed and loading time for video content were quite slow but they were finally reported as satisfying. This could mean that "fast enough" could exist or the population of respondents was used to these speeds in general.

It's interesting to find that many respondents (nearly 10%) when executing the first scenario stated that they used as search term "Joanne Noordink". This is the person the program is about. So an important note to make, not only to RIA designers, is that users do not exactly search on the fields you intended them to search on. In this case, "program-title" and "broadcast-title" were the ones we suggested, but the interesting conclusion was that users most likely will remember persons associated with the thing they are looking for better.

10.3.1 Top 5 issues

Drilling down in the analyses, in this paragraph we'll describe the top 5 issues found within EOx in the research.

1. **Performance**
   Maybe I shouldn't be this glad that the biggest issue, although not a design (or interaction) mistake, affects usability. There were many complaints of bad performance affecting the total load time of the Webpage (homepage) where EOx is placed at. This leading to no response from button clicks, and hence bad interactivity.
   Also within the RIA users were experiencing tons of interactivity problems because busy processors did not handle the input event, a nice issue to investigate for further research. The complaints varied from have "no sound" to "player doesn't work at all" and "it made my browser crash".
   A good extra lesson to learn here is that we should not make our RIA's to heavy (this is a common usability design guideline, however when building RIA it's more tempting to add special effects etc. at the cost of performance).
2. **Program offerings**
   The number two issue isn't actually of any importance for this thesis, so we'll spend not too many words and precious bits on this. The program offerings were, one, incomplete (too many people are searching for old TV programs that are not in the archive (yet)) and two, the selection criteria for programs in the archive were unclear. I guess the best lesson to learn even with state of the art user interface design an application will fail if the content isn't what people expect. The best practice would be to place a guide of the selection criteria that helps people to save costly time searching for things that cannot be found and thereby improve the user experience of the application.
3. **Quality of old recordings**
   For the third biggest issue, the same thing (as for the second) applies, so not too much words will be splurged here. The best lesson for us to learn here is probably that content should be of good quality for the success of the application.

4. **Tooltips**

   Tooltips are an amazing feature that can simplify usability. It's known almost since *graphical user interfaces* (GUI) exist, for example since Windows 3.11.[104] They are also part of the WAI specification[105], so that they appear when the user hovers over an image using the alt attribute (better known as alt-text) or over an anchor tag with the title attribute. Though, this standard is being violated when using the Adobe Flash platform or another RIA software (earlier described in paragraph 5.2.1) and is most likely violated when using complex JavaScript. In the last mentioned cases, the tooltips have to be incorporated in the user interface design. This has become a best practice.
5. **Mac compatibility**

   This, again, is an issue that exists for a long time and is a crime for Web designers: web browser compatibility. Since the RIA in this case study is within the browser (not in a browserless state), the same rules concerning browser compatibility apply as in traditional design apply. But also cross-platform (platform independence as stated in the EORIA-model). Respondents with a Mac operating system complain about not working fullscreen buttons and there were multiple complaints about the media player not working at all.

10.3.2 Lessons learned

The lesson we've learned is one to seriously regret. Namely that a great part of this usability test cannot be applied to the literature of our main interest. This is primarily due to the fact that business objectives are different from science objectives. For future research I would look further for another case, probably at another company, were they would allow you to actually create or fulfill your model which will match our literature more closely. However, it was a great and profound usability research which brought up many things of our interest and consequently many things for future research.

[104] Wikipedia, http://en.wikipedia.org/wiki/Windows_3.0

[105] See, http://www.w3.org/WAI/

10.4 Case Study Conclusions

We can conclude that EOx is not a well working application, but users are satisfied on how the application is working.

10.4.1 Practical fixes

The following ten things must be fixed in order to create a better user experience:

1. Performance issues must shrink in order to create better user experience.
   This can be done by:
   i. better programming
   ii. loading not so much items on the same page
   iii. move the RIA object to another page
   iv. not starting the video automatically (together witch loading the paged, this requires much CPU)
2. Add content and improve quality thereof. Also make very clear which content is included and why. This makes that people won't search for something that isn't there.
3. Add tool tips that tell people where they are hovering over and suggest what they are about to click on.
4. Make the application Mac compatible, or better cross platform and cross browser compatible.
5. Let buttons be buttons, so that they are visually notable and are expected to be buttons.
6. Use standard design patterns to improve on icons. No standard patterns are used currently and some of them are working confusingly.
7. Use existing design patterns to thoroughly design the entire application and apply one of breadcrumb (e.g.) and the given suggestions concerning the back button (see 7.2.1).
8. Include required features: links to content and an in-program button for care.
9. Provide the full application in Flash technology, or better without the need of a plug-in.
10. Improve on search by:
    i. giving more suggestions
    ii. add more tags on concerned persons in the episode
    iii. do not return result if they are not there
    iv. apply filters as year, type and category
    v. apply the design patterns as given in our literature for browsing

10.4.2 Added value for this thesis

Points of most added value for our literature study is:

1. We found that performance is a not to be underestimated value when building a RIA because it affects the user experience greatlly. We did not cover it so much in our literature, but performance and stability is for RIA even more important factor to work on than for desktop applications. Much literature about this can be found related to cloud computing. Interesting for further research.
2. Browser compatibility issues as described in our literature studies are confirmed, mainly through problems with Mac computers (platforms).
3. Software plug-in problems are bigger than we expected and wrote about. Reported failing Silverlight installations asks for an alternative technology. (As HTML5?)
4. User needs as described in 7.1.3 are confirmed.

These are in addition to all usability issues and statistics we found and confirmed patterns, common knowledge in the field and our literature.

11 Conclusion

We're now ready to setup the conclusion of this thesis. We'll run through the answers of the subquestions proposed in paragraph 4.1 and finally answer our main research question.

We've clearly been defining what RIA's are by comparing definitions and creating one ourselves. In addition we introduced the EORIA-model that extended our definition and also gives some handles for user interface designers to approach the concept and easily overcome design issues.

Now we know that there are some trends that thrive RIA's. We've seen that these include hardware trends as Internet availability, higher bandwidth, advancing hardware, more powerful mobile devices and web on TV and software trends included cloud computing and easier software development (authoring) tools. The last thriving force we discussed is the change in user needs which places a greater demand on this technology.

In chapter 8, we did a profound (de)construction of the RIA user interface and thereby showed all elements involved. This breaking apart gives designers more insight and helps by understanding and constructing design patterns.

Next, we saw an enumeration of multiple design patterns and concluded that many of these are still in evolution. We have seen that patterns categorized as "rich", "ajax" or "Web 2.0" can be well used. We also saw a few special libraries for RIA and some amazing examples of interactions. RIA's can make good use of existing patterns as long as more advanced patterns get priority. The patterns we were used to are not going to change, we should only note that the pattern collections are going to extend.

The multiple technologies that exist to build and execute a RIA include the Adobe Flash Platform, Microsoft Silverlight, Google Web toolkit, AJAX, HTML 5, and Apple Gainduia. We have seen their qualities and some of their usability issues.

We have compared traditional web applications and found that, in addition to changing design patterns, the browserless state, the multiple devices on which the application is displayed and screen resolution as well as other components mentioned in the EORIA-model differ.

The case study added valuable information on user needs and -preferences. Even more, the case study showed that for a web application performance it is a bigger issue than for its desktop equivalents, because much more disturbing factors come into play.

Now let's rehearse our main research question:

- Are there any noticeable changes for user interface designers while designing a RIA?

As you already could have guessed, the answer will be yes! Through our literature studies we have seen that some of the subjects of our subquestions on the previous page have changed and that there are new patterns for RIA that should be used.

In addition to that, user interface designers need to be aware of the characteristics of the application that are easily displayed through the EORIA-model.

It's not so much that designers need to alter their ways of working, but more that they need to extend it. Multiple devices, different input devices, screen resolution classes and browserless state are points of main concern which they need to take into account and apply the corresponding design patterns to.

Furthermore, they have to keep up with the described software and are especially influenced by the way applications are built (authoring and development tools), like Google Web toolkit and Flash Catalyst, more and more without adding, by many designers, large difficult blocks of code.

12 Afterthoughts

Now we've done our studies, a few afterthoughts and recommendations for further research.

The first thing that comes to my mind is that it is a privilege the work with such an exciting concept as RIA. While doing this study I wanted to do more, explore more and learn more. The beautiful design patterns challenged me to pick-up some design work in this field again. It gave me a pleasurable feeling of all the good things to come.

Second, the great amount of literature that can be found for such a (relatively) unknown concept amazed me. So many fields of expertise come together in RIA's. That's why I think they will set trends for user interface designers for years to come. Eventually we may not call them RIA anymore, but their characteristics will remain.

Some ideas for further research include:

- Deepen the EORIA-model. Work out the classes for devices and screen resolution, classify the operating systems and create a exhaustive list of the design patterns that belong to these classes. Or better, an online wizard to go through the model.
- As seen in our literature studies the usage of RIA by multiple users, especially in the living room may be interesting.
- The principle of context-awareness should be worked out to more clear and concise real-life examples.
- As seen in the case study, very important for the success of a RIA is the performance. More research could be done in the field of process management within the browser, the appropriateness of browsers to execute full-blown RIA's and the importance of performance issues for interface and interaction design.

13 References

This list of references begins with the list of figures and subsequently shows the resource websites (articles and facts) and the literature and papers used in this thesis.

13.1 List of figures

Below is the list of figures:

13.2 Websites

Below you'll find website articles and blog posts used in this thesis. Product- and Wikipedia pages are excluded.

Adler Ron, Stipins Janis, Google Learns to Crawl Flash (2008). Retrieved from http://googleblog.blogspot.com/2008/06/google-learns-to-crawl-flash.html

Akamai, The State of The Internet Q4 2009 (2009). Retrieved from http://www.akamai.com/stateoftheinternet/

Alam Muhammad Usama, Flash vs. Silverlight: What Suits Your Needs Best? (2009). Retrieved from http://www.smashingmagazine.com/2009/05/09/flash-vs-silverlight-what-suits-your-needs-best/

Berkun Scott, Cloud computing is a bad metaphor (2010). Retrieved from http://www.scottberkun.com/blog/2010/cloud-computing-is-a-bad-metaphor/

Boutin Paul, Cloudy Judgment (2008). Retrieved from http://www.slate.com/id/2188015

Burnt Jet, The Glossary of Terms for Programming (2008). Retrieved from http://burntjet.co.uk/programs/help/glossary.php

Dilger Daniel Eran, Apple developing Flash alternative named Gianduia (2010). Retrieved from http://www.appleinsider.com/articles/10/05/07/apple_developing_flash_alternative_named_gianduia.html

Eurostat, Information society statistics at regional level (2009). Retrieved from http://epp.eurostat.ec.europa.eu/statistics_explained/index.php/Information_society_statistics_at_regional_level

Goldman Jake, Will HTML 5 Replace Flash in the next 5 years? (2010). Retrieved from http://www.cmurrayconsulting.com/ask-a-web-strategist/html-5-flash-future/

Huff Bex, HTML 5 Versus Flash/Flex (2009). Retrieved from http://bexhuff.com/2009/06/html-5-versus-flash-flex

Jade Charles, Steve Jobs: Thoughts on Flash (2010). Retrieved from http://theappleblog.com/2010/04/29/steve-jobs-thoughts-on-flash/

James Justin, HTML 5 Editor Ian Hickson discusses feature... and more (2008). Retrieved from http://blogs.techrepublic.com.com/programming-and-development/?p=718

Karl B., Telltale employee says iPhone "much more powerful" than Wii (2009). Retrieved from http://www.qj.net/qjnet/wii/telltale-employee-says-iphone-much-more-powerful-than-wii.html

Kearns Dave, Microsoft over Google? Keep searching (2004). Retrieved from http://www.computerworld.com.au/article/16331/microsoft_over_google_keep_searching/

Ludwig Adrian, AIR passes 100 million Installs (2009). Retrieved from http://blogs.adobe.com/air/2009/01/air_passes_100_million_install.html?sdid=EENCL

Millward Brown (for Adobe), Flash Player Version Penetration (2010). Retrieved from http://www.adobe.com/products/player_census/flashplayer/version_penetration.html

Niel Theresa, Designing Rich Applications (2008). Retrieved from http://www.slideshare.net/theresaneil/designing-rich-applications

Niel Theresa, RIA Screen Layouts (2010). Retrieved from http://www.slideshare.net/theresaneil/ria-screen-layouts

Nielsen J., Flash 99% Bad (2000). Retrieved from http://www.useit.com/alertbox/20001029.html

Nielsen J., Screen Resolution and Page Layout (2006). Retrieved from: http://www.useit.com/alertbox/screen_resolution.html

Nielsen J., Why you only need to test with 5 Users (2000). Retrieved from http://www.useit.com/alertbox/20000319.html

Nudelman Greg, Design Patterns for Mobile Faceted Search (2010). retrieved from: http://www.uxmatters.com/mt/archives/2010/04/design-patterns-for-mobile-faceted-search-part-i.php

O'Reily Tim, What is Web 2.0? (2005). Retrieved from http://oreilly.com/web2/archive/what-is-web-20.html

Program-ACE, Glossary (2010). Retrieved from http://www.program-ace.com/glossary/

Shinedraw, Flash vs Silverlight fps Meter Stress Test (2008). Retrieved from http://www.shinedraw.com/mathematics/flash-vs-silverlight-fps-meter-stress-test/

Snook Jonathan, What is AJAX? (2005). Retrieved from http://snook.ca/archives/javascript/what_is_ajax

Tomesen Remco, De Pers, 'Web 2.0 is zelfbevrediging' (2007). Retrieved from http://www.depers.nl/economie/127990/Web-20-is-zelfbevrediging.html

The Economist, Cloud computing: clash of the clouds (2009). Retrieved from http://www.economist.com/node/14637206?story_id=14637206

13.3 Literature and Papers

Bouras Christos, Konidaris Agisilaos, Estimating and eliminating redundant data transfers over the Web: a fragment based approach (2003)

Brown C. Marlin, Human-Computer Interface (1999)

Cheong Hyuk Jun, Morrison, Margaret A., Consumers' Reliance on Product Information and Recommendations Found in UGC (2008)

Cassidy John, Dot.con: How America Lost its Mind and Its Money in the Internet Era (2002)

Cockburn A., McKenzie B. and Smith M. Jason, Pushing back: evaluating a new behavior for the back and forward buttons in web browsers. - Human–Computer Interaction Lab, Department of Computer Science, University of Canterbury, Christchurch, New Zealand. (2002)

Constantine, Larry L., Engineering Usability into Software (1995)

Daugherty Terry (et al.), Exploring Consumer Motivations For Creating User Generated Content (2008)

Deloitte, Change your World, or the World will Change you (2008)

Desisto Robert P. (et al.), SaaS Delivery Challenges On-Premise Software (2006)

Garett Jesse James, Ajax: A New Approach to Web Applications (2005)

Holzinger Andreas, Ebner Martin, Interaction and Usability of Simulations & Animations -A case study of the Flash Technology (2003)

Huurdeman H.C., Interactive Video in Serious Games (2007)

Kaswar Fahim (et al.), A portable toolkit for supporting end-user personalization and control in context-aware applications (2009)

Koch Nora, Rossi Gustavo, Patterns for Adaptive Web Applications (2002)

Knemeyer Dirk, There are only four things that people do on the Web (2003)

Kramer Joseph (et al.), A user-centered design approach to personalization (2000)

Lekakos George, Chorianopoulos Kostas, Spinelis Diomidis, Information systems in the living room: A case study of personalized interactive TV design (2001)

Lightner Nancy J., What users want in e-commerce design effects of age, education and income (2003)

Paterno Fabio, Santoro Carmen, Designing Multi-Device Interactive Services through Multiple Abstraction Levels (2005)

Piyasirivej, Pilun, Towards usability evaluation of Flash Websites (2004)

Reinhardt Robert, Down Snow, Adobe Flash CS3 Professional Bible (2007)

Robertson George, Czerwinski Mary, Baudisch Patrick, Brian Meyers, Robbins Daniel, Smith Greg, and Tan Desney, Large Display User Experience (2005)

Schaller David T., Allison-Bunnell Steven, To Flash or Not To Flash? Usability and User Engagement of HTML vs. Flash (2004)

Schneiderman Ben, Plaisant Catherine, Designing the User Interface: Strategies for Effective Human-Computer Interaction 4th Edition (2004)

Scott Bill, Niel Theresa, Designing Web Interfaces (2009)

Schroeder W., Perfetti C. & Spool J.M., Enhancing User Interaction in Pet Market - Bradford, MA: User Interface Engineering (2002)

Staten James, Is Cloud Computing Ready For The Enterprise? (2008)

Tidwell J., Designing Interfaces (2005)

Vliet Hans van, Software Engineering - Practice and principles 3rd edition (2008)

Vermeulen W., Boer den W., De web browser interface - Is een beter design mogelijk? (2008)

Welie Martijn van, van der Veer, Gerrit C. & Eliens Anton, Patterns as Tools for User Interface Design (2000)

Zimmermann Andreas, Specht Marcus & Lorenz Andreas, Personalization and Context Management (2005)

13.4 List of Design Pattern Libraries

Below is a list of design pattern libraries used in this thesis. These are mixed for RIA and traditional web applications (because most libraries are extending) and are highly recommended.

1. AJAX patterns
   http://ajaxpatterns.org/
2. Design Pattern Chooser
   http://interface.fh-potsdam.de/infodesignpatterns/patterns.php
3. Design Solutions on Flickr
   http://www.flickr.com/photos/guspim/collections/72157600047307884/
4. Designing Web Interfaces
   http://designingwebinterfaces.com/ria-screen-layouts
5. Endeca User Interface Design Pattern Library
   http://patterns.endeca.com/content/library/en/home.html
6. Konigi Interface elements
   http://konigi.com/interface
7. Martijn van Welie's Interaction Design Pattern Library
   http://www.welie.com/patterns/
8. Pattern tap
   http://patterntap.com/
9. Patternry
   http://patternry.com/
10. Quince, the community driven User Interface Pattern Library
    http://quince.infragistics.com/#/Main
11. Search Patterns on Flickr
    http://www.flickr.com/photos/morville/collections/72157603785835882/
12. Social Patterns
    http://www.designingsocialinterfaces.com/patterns/Main_Page
13. Theresa Niels Libraries on Slide Share
    http://www.slideshare.net/theresaneil
14. UI Pattern Library on Flickr
    http://www.flickr.com/groups/namics_uipatterns/
15. UI scraps
    http://uiscraps.tumblr.com/
16. UI-Patterns
    http://ui-patterns.com/

17. UIpatterns

    http://www.uipatterns.net/

18. Usability Institute Gallery

    http://www.usabilityinstitute.com/resources/learnabilityGallery.htm

19. UXmagazine RIA Screen design

    http://www.uxmag.com/design/rich-internet-application-screen-design

20. Yahoo Developer Network - yPatterns

    http://developer.yahoo.com/ypatterns/everything.html

14 Acknowledgements

I would like to express my sincere gratitude to:

Gerrit van der Veer, for supervision and throughout my study giving me the opportunity to express my creativity and participate in incredible projects.

Second reader XXX, for the time and effort invested in the second evaluation

My family, friends and co-students, for their continuing support and encouragement

De EO, for accommodating the case study, adding value to this research

The King of Knowledge, Wellspring of Wisdom, the reason I live, Jesus Christ

Appendices