UNIVERSITY OF GOTHENBURG

Rich Internet Applications with Real-time push mechanism

Shahzeb Muhammad Iqbal

Bachelor of Software Engineering & Management Thesis

Report No. 2009-059
ISSN: 1651-4769

University of Gothenburg
Department of Applied Information Technology
Gothenburg, Sweden, May 2009

Rich Internet Applications (RIA) with Real-time push mechanism

Shahzeb Muhammad Iqbal

Applied Information Technology, IT University of Göteborg
Göteborg University and Chalmers University of Technology

SUMMARY

From last two decades the internet has became a house hold source of information, education, sales, marketing, advertising etc. It has become a part of everyone's life up to some extent. Everyday more and more people are getting introduced to internet and use it for different reasons. The reason could be as simple as paying bills or anything else. There are different user groups using internet with different level of knowledge, and for this reason, the tasks they want to perform should be design in such a way that all the user groups no matter what knowledge background they come from, should feel comfortable performing them. This brings a responsibility on the developers of websites and web applications to implement well designed products which are suitable for usage of everyone.

This report addresses the traditional web application development, and brings new methods in discussion which could potentially make the development process and implementation easier and richer. The report introduces the audience to the new terminology called as RIA. RIA uses a different technology stack for development of web applications then the traditional methods used currently. The report discusses the benefits and limitations of the RIA technologies and what can be achieved through it. Further the author talks about the efficiency which can be gained in terms of execution time and richness which could be gained in terms of graphical user interface through the usage of RIA technology stack.

The author discusses a particular web application called as GWT, and develops a small part of it using RIA technology stack as a prototype, and compares the efficiency in terms of "execution time" with the same component when developed using traditional methods.

The author concludes with the discussion on the results of the tests performed on the development prototype. The report is an interesting introduction to RIA and would server as an initial step to learn about the technologies under discussion.

Acknowledgement

First of all I would like to thank Ideal Systems Nordics for providing me with such a wide and interesting topic for my thesis. Secondly I would like to thank the R&D team of the Ideal systems in Brussels Belgium for training me on their in-house developed products, and equipping me for the basic requirements of writing this thesis and giving me a good start.

I would like to thank my Supervisor and course coordinator for all their help and motivation to make my thesis better in every aspect, providing me with very valuable examples, and giving feedback on the contents and structure of the thesis.

Last but not the least I would like to acknowledge my family and my friends at IT University of Göteborg for their help and support during my entire educational period in Göteborg (Sweden).

Disclaimer

The topic of this thesis revolves around web development and how to make it better. The nature of the topic is so wide that one cannot simply stick to traditional literature references to prove points, the author have taken help from and, thus, reference several different websites and published papers to clarify and explain the terms.

Great care has been taken in an attempt to mention all the URLs of the websites in the reference list, and to tag each and every part of this paper with the reference to the original content. The author will take complete responsibility if any of the text in this paper is left un-tagged referring to the original content. If you feel that your work has been used without proper attribution, please contact the author so that the information can be accredited correctly.

Contents

1.0 Introduction

The idea of this thesis traces it origins back to a blog, created with the objective of discussing all the aspects of web based applications such as:

1- Current technologies being used in order to develop web based applications
2- Frameworks being used such as MVC (Model View Control)
3- The efficient combination of different technologies

One specific entry which influenced the topic of this thesis included a quotation from Kurt Willams:

“Lately everyone in the industry is falling all over themselves to offer "Rich Internet applications." We want our applications to be "richer." What does that mean? At its core, "rich" means "more like the GUI apps we used to write ten years ago [1].”

The discussion was regarding a proposal in which the Java platform would be used in a different way called “Java Web Start” comprising a different technology stack for web application development. Currently most of the web applications are being developed with the traditional markup-based method, the post in the blog was a comparison of the technology stack which was needed for the traditional markup-based web applications compared to applications deployed using Java Web Start.

Web based application development is a complex process, and can become even more so taking various business requirements into account. The development itself starts out in a simple manner, but in this process the limitations quickly become apparent as the complexity of the business requirements are uncovered. It can be surprisingly difficult, in the beginning of a new project, to estimate the complexity of the final application, making it equally hard to determine if the product would serve best as a standalone application, or could benefit from being implemented using web technologies.

1.1 Purpose

This thesis will conduct a study about web based applications development in the context of a contact center soft phone web based application, developed by using web technologies by Ideal systems Nordics, named as GWT (Genesys Web Toolbar). The study will include implementing a small prototype of this application using the Rich Internet application specific technologies using real a time push mechanism.
The core question which this thesis will try to answer is as follow:

Is it possible to gain efficiency by developing a web based application with RIA specific technologies than traditional web technologies, on the grounds of execution time, specifically in the case of GWT ?

(Note: the author doesn’t claim that an efficiency is possible, but will make an attempt to see if it is or not)

In the light of the core question of this thesis, there are some additional questions which would be answered to support the core questions, which are as follows:

- What problems can arise when developing web based applications with RIA?
- What are the limitations of using RIA-specific technologies?

1.2 Motivation

The motivation behind this thesis is to investigate some new methods of implementing web based applications with RIA technologies in order to benefit companies like Ideal systems. This thesis will help such companies with their future products and would provide a platform on which to base future studies and work.

1.3 Personal Motivation

Various kinds of applications have been implemented using RIAs architecture but designing an agent desktop application using RIA has, to the author's knowledge, not been attempted before. It thus provides both an interesting as well as a technological challenge to solve. This thesis will act as a platform and try to answer some commonly asked questions; the prototype solution developed alongside the thesis will act as an example for further development of such applications.

1.4 Outcome

The outcome of this thesis could provide a foundation for the R&D team of Ideal Systems, and other such companies, to develop agent desktop application using a RIA architecture; further investigation could be done within the following topics:

- Investigating the advantages of different RIA architectures and their benefits
- Investigating and learning about the new web technologies associated with RIA desktop applications
- Investigating different RIA architectures, finding possible solutions for different kind of applications

2.0 Method

From the onset, this thesis was planned to end up as a contribution to the ongoing investigation into RIA systems within Ideal System Nordics AB, and as such needed to conform to the pre-existing research having been conducted using the “Action Research” methodology. Though the author have tried best to follow all the guide lines of “Action Research” methodology, but have yet fell short in some of the phases to do so. It have been fairly difficult to shape some of the information under the guild lines of the “Action Research” in general which will be discussed in the chapter below.

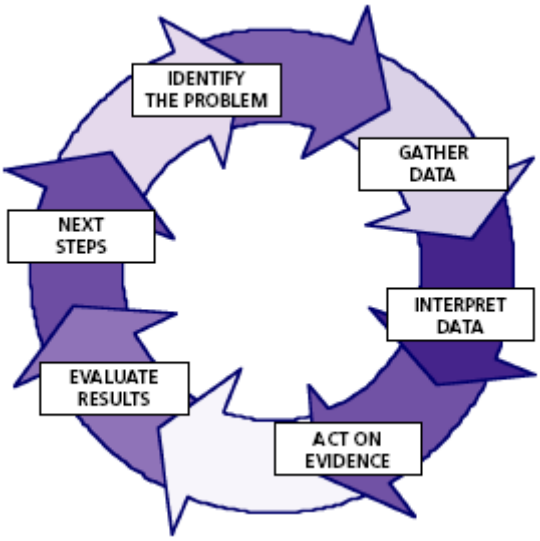
“*Action research is a* reflective process *of progressive* problem solving *led by individuals working with others in teams or as part of a "*community of practice*" to improve the way they address issues and solve problems. Action research can also be undertaken by larger organizations or institutions, assisted or guided by professional researchers, with the aim of improving their strategies, practices, and knowledge of the environments within which they practice [2]*”

One of the reasons why action research fits the topic of this paper is due to the process of studying the results of the tests being performed by the author.

“*Action research is a process in which participants examine their own educational practice systematically and carefully [3]”.*

Action research provides many methods of examining the results of the tests being performed in order to watch the differences between the systems being studied. In addition to that, it provides the author with the means of how to express the results in a scientific manner, and from this be able to identify significant differences, and how these differences affect performance. Though most part of this thesis focus on answering the questions regarding RIA instead of actually providing a solution, the prototype solution will act as an example of how the discussed applications can benefit from RIA.

This paper focus on providing an answer to the question posed in section 1.1, using the simple cycle of the action research shown in the diagram defined by *[4]*.



The first step to approach “Action Reach” methodology is to identify the problem and plan an approach to address it. The problem of this thesis has been identified, and core question which simplifies the problem have been put forward in the section 1.1. Further on a plan has been formulated about how such problems can be addressed; the motivation behind this investigation as a whole, and what the author expects in the outcome, and how the outcome will serve for future continuity if conducted.

The second step entails gathering data about the problem domain, and, using the gathered data to answer the research questions. The author has conducted an investigation on many RIA related solutions, and gathered some reports from surveys conducted in order to access the benefits of RIA from development point of view, and user point of view. The gathered data has then been used to justify the approach of developing contact center soft phones with RIA based technologies, and what benefits would be gained in doing so.

The third step is to interpret the data and come up with an example prototype of how the solution can be implemented. Once the author has made an attempt to justify the approach, the next thing is to plan the implementation of the prototype to make a stronger point based on the previous discussion. The author discusses the software of which a small part will be developed using the RIA technology stack, explains some of the background of that software, its current architecture, and what will be done during the prototype.

Next step is to implement a solution to the problem using the decided method and gather results by testing the implementation. Before reaching this step, the author has already discussed the prototype approach and architecture in details, next thing which has been done is to a try out a prototype, discuss the implemented code and perform tests. The author has implemented a small part of a soft phone web based application using RIA technologies stack, and then used a test bench to access the performance in executing time. It's always a challenge to perform tests and isolate the test from the environmental factors; in addition to that, it's never easy to know how many tests are enough to prove a point. The author has performed the test number of times, though the results has proven positive all the time, he has only shown three of the tests which were quite impressive.

Last but not least, the tests must be evaluated to observe the differences. In the Final section of this thesis, the author has drawn a conclusion regarding all the tests performed, the results of the tests and the efficiency with respect to time. Some of the other factors have also been discussed which are some of the leading discussion points when it comes to RIA, which could prove potential investigational points for future.

The motivation behind this thesis have been stated in the introductory section, and the thesis was conducted in the following manner

2.1 How action research was used and shortcomings

From the very beginning the author regarded action research as a methodology which would help him in both action outcomes and research outcomes. As a research initiative, the author's intention was to gain an appropriate understanding of the technological components which are involved in this thesis. The primary objective of the author was to keep the emphasis on the research (understanding) of the topic and related technology components, and keep the action part as a fringe benefit.

Initially the author struggled with formulating the main question in a presentable format, but action research provided enough flexibility to allow a “fuzzy” beginning. The cyclic nature of the methodology helped in refining the main questions by “action --> reflection --> action” method, which means getting response and feedback from technical and presentational supervisors step by step, which guided the actions in the right direction and finally towards an agreeable outcome.

The author starts with an explanation of the RIA technology to create an understanding of what exactly RIA means and what the benefits of such technology are. The author goes on explaining the technology stack which is related to RIA, and also explores some of its limitations.

Finding relevant literature for related research and accompanying literature review was troublesome, mostly due to the fact that the topic itself is relatively new, technology-wise, and had not been used much in the exact context of this thesis. The author decided to continue anyway by doing a literature review technology-wise than context-wise, i.e. the author tried to present the success rates achieved by developing web applications where RIA specific technologies had been used, and how RIA had benefited companies in different areas. The purpose was to give the reviewers and readers a small overview of what kind of work have already been done in RIA, what have been the success rates and in which areas the work have been done.

The author didn’t conduct the Action part (prototype implementation) quite as it should have been under the guidelines of “Action Research”. Action Research expects the action to be cyclic and to conduct action experiments more than once in different manners to see the improvement difference. But in this thesis the author didn’t conduct an implementation of the prototype more then once using different parts of the RIA technology stack to see whether how much difference in efficiency is gained, whether the difference is positive or not with different technologies. For example, the author used AJAX technology to prove his point and support his argument, which gave successful positive results efficiency wise, but on the other hand, it might be a good idea to implement the same prototype with RIA technologies such as “Microsoft Silverlight” or any other technology to see if the efficiency prevails, which could have been done. The reason behind this was that the code which was being rewritten for experimental purposes is part of a system which is entirely written in Java, and AJAX is one of the technologies with is best compatible with java, whereas using other technologies could have made it more complex, and maybe out of the context of this thesis. Secondly it would have taken longer time.

One of shortcomings of this thesis is the fact that the topic under discussion is much too broad. The thesis discusses the efficiency gained by using RIA technology, but when we talk about a technology, especially a web technology, it contains enormous methods of doing something in different ways, and there is no way that each and every method can be tried for the sake of argument. The point is, the author have successfully contributed to the knowledge and change through this thesis, which are strong points, but the author could not claim that attempts to disprove them and find other alternatives will not be possible. The author has also discussed this in the conclusion of this thesis.

Other than these, the author have made a wholehearted attempt and made his best effort to keep the flow of the information transfer as smooth as possible, to avoid any confusion for the readers and reviewers.

2.2 Preparation for the thesis

In order to undertake the thesis it was necessary to study the current state of web-oriented technologies, to understand what was being used today, how it was used, and why. Web applications are typically created using several complementary technologies; a thorough understanding was needed of those basic building blocks. The technologies which were studied as a part of these thesis prerequisites are as follow:

- HTML, CSS and Javascript
- Web Standards
- AJAX
- Java web start
- Action based MVC frame
- Different kind of architectures of web applications
- Interceptors
- Form validation
- Techniques of testing web applications

2.3 During thesis

After developing a significant knowledge of the traditional web applications, the way they are developed, tested, and some of the important technologies which are involved, the next step was to take the initiative to study RIA itself.
This entailed understanding the concept, studying the technologies being used to develop RIAs, observing the limitations in general and in terms of the work (the prototype under development.)

In order to understand the development of the prototype using RIA, and to develop an understanding of the technologies being used in RIA, different short workshops were arranged by Ideal Systems Nordics and, in addition to that, valuable knowledge about the in-house developed software was provided, in the initial stages of the thesis.

The technologies for which workshops were arranged are as follow

- Rich Internet Applications
- XUL – XML User Interface Language
- Silverlight
- OpenLaszlo
- Flex
- JavaFX

2.4 Thesis finalization

The final part of the thesis documents all the activities performed throughout the entirety of the thesis. Since there is lot of information available on the Internet about the technologies studied in this thesis, and as such it is the author's opinion that describing them in any greater detail would be a waste of the readers' time. The focus has been kept on answering the core questions, and only defines RIA itself in general terms as it is the center of gravity for this thesis.

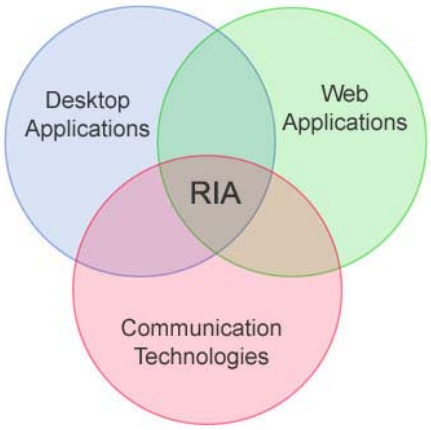
The final part is also spent describing the work done in the prototype, examining the results, drawing conclusions about the results and discussing the entire thesis process.

3.0 Rich Internet Applications

Rich Internet applications (RIA) are web applications that have the features and functionality of traditional desktop applications. RIAs typically transfer the processing necessary for the user interface to the web client but keep the bulk of the data (e.g. application state, domain-specific information, etc.) on the application server *[16]*.

Rich Internet Applications combine the best user interface functionality of desktop software applications with the broad reach and low-cost deployment of web applications and the best of interactive, multimedia communication. The end result is an application which provides a more intuitive, responsive, and effective user experience *[17]*.

Real-time pushing is an architecture in which a web server sends data to a client program (normally a web browser) asynchronously without any need for the client to explicitly request it. It allows creation of event-driven web applications, enabling real-time interaction otherwise impossible in a browser. This is also known as the term Comet *[18]*.



Real-time applications use long-lived HTTP connections between the client and server, which the server can respond to lazily, pushing new data to the client as it becomes available.

Many technologies have emerged to fill up the shortcomings of HTML-based web applications and provide richer interfaces for users. Some of the examples of these are Adobe Flash and client technologies both for Java and Microsoft platforms *[7]*. Combined, the applications developed by using these technologies are called as Rich Internet Applications (RIA). AJAX is commonly also regarded as a technique for implementing RIAs.

The RIA architecture has been around for quite a while now. It was first introduced in a paper back in 2002 by macromedia *[8]*, though unofficially the term have been around for a longer period with the names of

1. Remote Scripting, by Microsoft, 1998
2. X Internet, by Forrester Research in October 2000

4.0 Difficulties, limitations and benefits of web application development using RIA in general

4.1 Difficulties using RIA

Traditional web application development with standard HTML is often performed using simplistic software architectures, designed using a very limited set of development options which, however, are relatively easy to manage, design and test.

With the introduction of RIA, development has become more advanced, while at the same time growing in complexity. The companies developing web applications using RIA-oriented frameworks and technologies face more problems due to the increased complexity in design, testing, measurements and support.

The companies using RIA frameworks and technologies for application development are, in addition to the development problems, also faced with several service level management challenges as well, many of which have not yet been solved at present time.

The service level management is of no issue to the developers, and is rarely considered as a problem by the users of those applications, but they are still important in terms of successful delivery of such applications. Some of the different aspects of the RIA that complicate management processes *[9]* are explained below.

Traditional web applications can be seen as a number of different web pages and each of them need to be downloaded separately by an http request. This model is called as the *Web page paradigm*.

In the case of RIAs, the web page paradigm is totally invalidated; instead RIA introduces a new way of asynchronous server communication which often support and enable a more responsive user interface. In RIAs, the time taken to download a page may not be of that much importance to the users; this is because the client engine may be in a mode of constant download to fetch the contents for further use.

To advance the field of RIA-oriented development, and solve some of the service level management problems, a standard tool to measure the response time of RIAs could, and indeed should, be developed.

Currently, due to the absence of such tools, most of the time the developers must craft their application code to produce the measurement data needed for service level management.

Measuring response times between client and server in a traditional web application can easily be done by placing a computer, somewhere between the two on the network, with the expressed mission to observe the flow of traffic produced by the network on the TCP and HTTP levels. Because of the fact that these protocols are predictable and synchronous, any kind of a packet sniffer can read these and interpret them on the packet

level. This would however increase the response time experienced by the user by tracking the messages from HTTP and measuring latency of underlying packets of the TCP and acknowledgments.

The interesting fact about RIA architectures is that these reduce the power of the packet sniffing approach, because the client engine works in a slightly different way by breking up the communication between the user and the server into to two different cycles operating in different ways, a foreground (user-to-engine) cycle, and a background (engine-to-server) cycle. Both of these cycles are very important, because neither of them stands alone; it is the relationship between them which define the behaviors of the application.

This relationship, however, is completely dependent on the design of the application, which cannot be inferred by a measurement tool, especially one that can observe only one of the two cycles. Therefore the most complete RIA measurements can currently only be obtained using tools which reside on the client and observe both cycles.

4.2 Limitations of Web Application Development using RIA

All technologies have limitations, RIAs are no exception, but it is always up to the design and development team to keep the shortcomings of the used technology in mind, and in check, before designing an application, so that no hurdles are faced during the actual implementation.

One of the limitations of RIA web based desktop applications is that they run in a sandbox, a locked down environment which restrict access to the resources of the system. In the case of Ideal Systems, they deploy a desktop web application. Any assumptions made about the resources of the system in use could be incorrect, in which case the RIA could fail to operate, which would restrict the user from using the application.

Javascript have always been a trouble with many web technologies, and it is one of the languages which are required for RIAs as well. Javascript errors can make the development of RIAs very complex since they are hard to catch and sometimes even the Javascript debuggers cannot catch the errors. The usage of Javascript for agent desktop applications would prove an even greater hassle since there is a lot of event handling in such applications, which could easily be broken and difficult to fix. An example of a problem which could occur on the client is if the user has disabled active scripting in their browser, whereupon the RIA may not function properly, if at all.

One of the limitations which could result in consequential loss of performance would be the usage of client-side scripts written in languages which are interpreted, such as Javascript. This is not an issue with compiled client languages such as Java, where performance is comparable to that of traditional compiled languages, or with Flash, Curl, or Silverlight, in which the compiled code is run by a Flash, Curl or Silverlight plugins.

Many manufacturers of the web browsers have already made a release or are working towards releasing a solution for Javascript engines which would solve the issue of loss of

performance, for example TraceMonkey, the Javascript engine used in Mozilla Firefox, is already using "Trace Trees" in version 3.1 of the browser.

But with this, another limitation arise, which is the restriction of usage of browser with the application. The application shall be used only with the browser which supports some typical functionality, which could become a business limitation as well.

As compared to the Mozilla Firefox, Google's web browser has the V8 Javascript engine, which also accelerates Javascript execution.

4.3 Benefits of using RIA

In order to create a successful application, is must satisfy the users needs and provide them with the functionality they need to accomplish their tasks. If the application does this, and provides a stable and predictable environment, there is a great chance that users will continue using it, and promote it among their colleagues. People mostly turn towards the companies which can provide them better services, which can help them to accomplish their goals as efficiently as possible. If customers are dissatisfied by your services, they turn to your competitors who offer the same business you do, and possibly with better services, so at the end of the day, it is all about serving your customers. If you invest a little bit extra in giving your customers the services they are looking for and expecting, in return they can become your greatest sales people, producing a chain reaction of repeat-business.

Distributors and retailers can improve the revenues of their companies by improving the user experience by providing simple interfaces to complex methods and tasks. This is where RIA-based systems come into play, providing a great opportunity to simplify business processes by their ability of making simple, highly interactive interfaces for web applications. More and more online-business owners and retailers are beginning to invest in RIA-based technology development in order to help them differentiate their offering from their competitors.

The organizations and businesses which specialize in the areas of service providing can easily make a great impact on their business by enhancing their customer services with RIAs. The primary reason is to provide their customers with the service they are looking for, service which can guide them through complex decision making processes. The main benefit in these cases is to make customers self sufficient in solving their own problems, and answering their questions, which brings down the call center costs for the service providers.

By using some of the processing power of the computers they run on, RIAs have the same speed and performance as any other desktop application. Since RIAs does not need a full page refresh all the time unlike traditional web applications, this increased interactivity and responsiveness improve the end-users experiences, resulting in greater productivity.

5.0 Litrature Review

Rich Internet applications are all over the place nowadays, it is the most talked about branch of web application development at present time. According to a survey, by 2010 at least 60 percent of new application development projects will include RIA technology, and at least 25 percent of those will rely primarily on RIA *[10]*.

What make RIAs so widely adoptable in all kind of applications is its richness and capabilities. The three biggest reasons which have made people storm towards RIA-oriented solutions for their businesses are:

- Increased sales
- Improved service and cost reductions
- Improved productivity

According to an article, Rich Internet Applications are the next stage in the evolution of software and web technology. They offer a range of benefits to users, software developers and businesses alike, such as:

- Improved and enriched user experiences
- New opportunities for enhanced products and services
- Reduced cost of deployment and support

[10].

Since RIA platforms are going through continuous research, all the companies and organizations are having a research in implementing there desktop applications according to the RIA architecture specifications. But RIA itself is not a strict architecture or specification. It has open boundaries which can be modified and adapted, depending on the application and its required performance.

"Rich Internet applications are well suited for financial services interactions that are extremely complex, rely on graphics and models, and often require data from multiple sources. Customer experience professionals at financial services institutions should identify moments of truth where RIAs can have a big impact and connect these richer experiences into the overall flow of their sites [11]

In light of the above reference, this thesis focus on describing and explaining RIA and its limitations towards web application development in context of agent desktop applications. The benefits of designing an agent desktop application on top of a RIA platform will be shown by developing a small prototype and evaluating the results as compared to the same application developed with traditional web applications, and how performance differ between the two. This thesis would benefit the ongoing research in RIA within many organizations for developing agent desktop applications using RIA specific technologies.

"While a variety of applications have already been ported over to RIA environments, not all of them have been successful. Just because RIA technology is available, doesn't mean that it's the right solution for every process:" [12]

In the light of the above reference, the author will attempt to answer whether or not the RIA platform can be a possible solution for agent desktop applications. To study whether the implementation using RIA-technology would enhance the performance of such applications and if it can open up for the possibility of people to think of such applications in a different way.

"The RIA platform is best suited for applications such as

- *Product Catalogs and Product Selectors*
- *Product Confi gurators*
- *Productivity Applications*
- *Entertainment Applications*

[12]

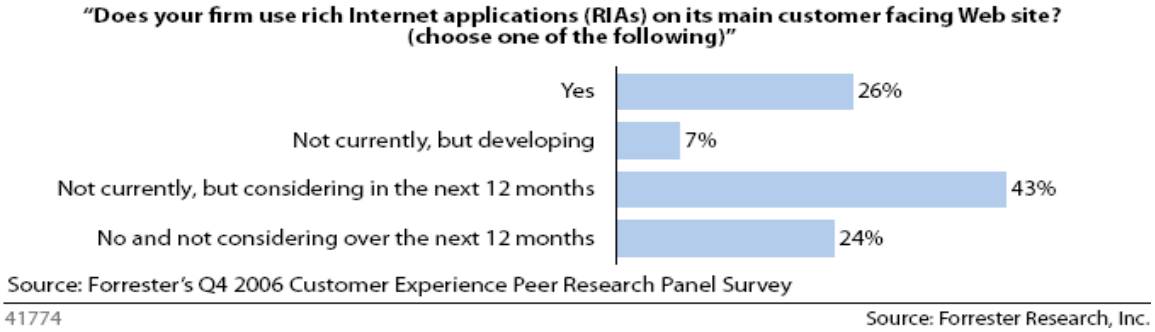
RIAs integrate the best of both Desktop & Web *technologies (Extraordinary interactive experiences) [10]*

Feature	Desktop	Web	RIA
Steady and responsive	✓	✗	✓
Rich and capable	✓	✗	✓
Easy to get hold of and low maintenance	✗	✓	✓
Great communicator	✗	✓	✓

The table above compares the existing applications which can be approached in different ways. RIAs are the fastest growing type within all the approaches, the reason behind this is that they can provide the best user experiences.

Rich Internet Applications are the next stage in the evolution of software and web technology. They offer a range of benefits to users, software developers and businesses alike:

- Improved and enriched user experiences
- New opportunities for enhanced products and services
- Reduced cost of deployment and support



The figure above is taken from *[13]* and shows results of a survey performed by *[14]*. It shows the rapid growth in the adaptation of RIA technology by the companies for their web services.

6.0 Prototype

The prototype phase could not begin until the author had thorough knowledge in the various web technologies existing today. Ideal systems gratuitously payed for this education, and followed it up by making their in-house software available to the author. With these conditions met, it was now possible to re-implement a small part of their software, using RIA technologies.
The possible limitations and the problems which could be faced during the implementation are discussed in chapter 4. Even after taking such measures, the problems yet faced during implementation are discussed in chapter 7. This chapter describes the purpose of the software which was re-implemented, the underlying reason for doing such an experiment from a technical and business point of view, what was done, how it was done and what the results were.

6.1 Genesys Web Toolbar background

GWT (Genesys Web Toolbar) is a unique framework for its customers: integrated voice, fax and email handling, all fully web-enabled. It was written from an integration perspective: integration to any type of back-end system is straightforward via a Universal Connector, allowing integration to web-based, fat client-based or even mainframe applications. All interactions and attached call interaction data are passed to the back-end solution, allowing customers to define their business processes any way they want, without being dependent of the GWT *[15]*.

6.2 Assessment of current GWT and what RIA can do

Currently GWT is being used in many companies as a contact center application. More than 10,000 agents are using GWT in different countries. Since GWT is being selected by different countries across Europe for the contact centers, the developers are facing massive problems due to its inflexible nature, and the reason is its current implementation restrictions. In other words, it have become very hard for the developers to make changes in the appearance of different windows according to the needs of the different companies, adding new features, and translating the language of the application according to the country its being used in.

Another big issue with GWT is its performance. GWT works on the AIL layer of Genesys, which have been improved and gone through lots of changes in the last year, but GWT have not yet adopted those changes, which means that GWT is not capable of using the new features available.

The factors mentioned above are affecting GWT not only technically, but are also affecting the reputation of the application in the market, and the reputation of Ideal Systems Nordics AB which owns GWT and have developed it from scratch. By having competitors like Telia-Sonera and Cisco Telecoms, it has become a “do or die” situation

for Ideal Systems Nordics to improve GWT and its framework to keep itself alive in the market.

The reason for implementing GWT with RIA is to allow the users to experience the consistency of the application regardless of what operating system and environment is being used by the client. Ideal systems offer their clients the ability to request customizations, implementing new features for them if required. The usage of RIA technology could greatly simply this type of additions. The RIA architecture is well structured, ensuring that Ideal systems, and other companies like it, can confidently begin implementing any kind of customization which the customer request. Stability of the application will not be an issue. In using RIA, the developers would be able to implement user-interface behaviors which are not obtainable using only HTML widgets, which are only available to standard browser-based web applications. This richer functionality may include anything which can be implemented using the technology available on the client side, includingdrag and drop, using a slider to change data, etc. One of the biggest advantages of using RIAs is that some of the calculations can be outsourced to the client, making the application more responsive compared to traditional web applications which must constantly interact with the server. Having the client perform some of the calculations also improves server performance by simply distributing the processing load.

6.3 Try out improvement

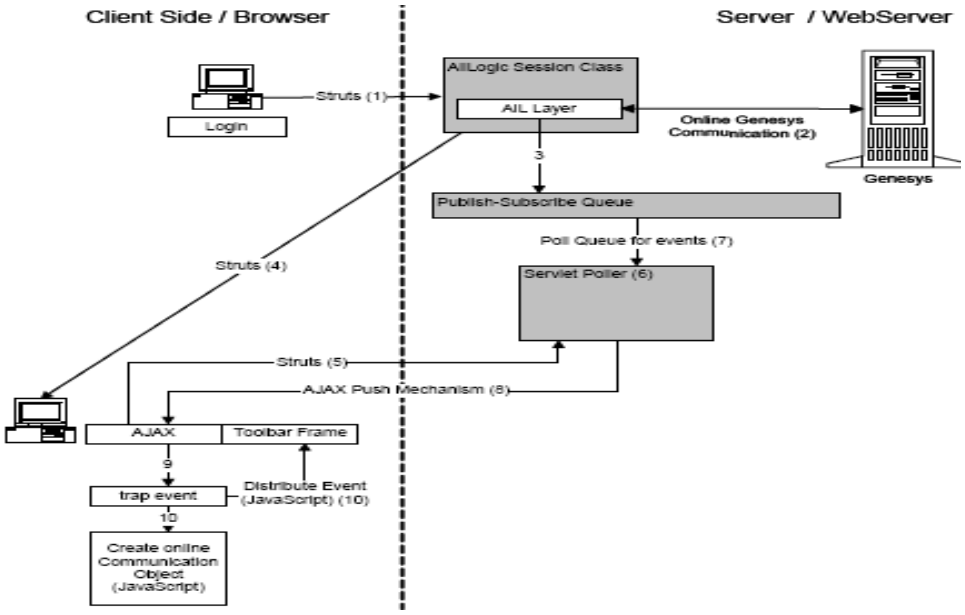
GWT is a huge system with more than 800 classes, 150 java server pages and more than 100,000 lines of code. In order to observe any discernible results using RIA, a small part of GWT was selected, and re-implemented. The objective was twofold, first of all, to get a feel for the architecture and observe the availability of new functionality, and secondly, to measure any improvement in performance over the previous implementation.

Work begain with the login of GWT and it was re-implemented by using the same basic principle and functionality which was present in the previous implementation. The new login page was implemented using AJAX.

AJAX (Asynchronous Javascript And XML), is a group of interrelated web development techniques used for creating interactive web applications. With AJAX, web applications can retrieve data from the server asynchronously in the background without interfering with the display and behavior of the existing page. Data is retrieved using the XMLHttpRequest object or through the use of Remote Scripting in browsers which does not support it. Despite the name, the use of Javascript, XML, and asynchronicity is not required.(XMLHttpRequest Usability Guidelines)(Beginning Ajax. wrox)*[19]*

The basic idea was to send an AJAX request to the server requesting data from the Genesys AIL layer, use html, HTML and CSS to design the login page, use a Java Server Page to display the HTML contents, use tomcat 6.0.14 as server to execute the Java Server Page, and use struct MVC (model view control) architecture of action classes and mapping them to correct Java Server Pages on success, or returning back to the previous page from where the request was sent in case of an error.

If the above text is put in a diagram, then it would look something like this. Ignore the right hand part which is the plan for further implementation using a push mechanism.



6.4 Source code explanation

The source code can be found under Appendix 1. AJAX Client Engine (ACE) is a Javascript component which makes it easy to develop AJAX-style web applications. A developer only need be familiar with three Javascript classes; Engine, Request, and Response, to access all the functionality of the XMLHTTPRequest object as well as additional framework services.

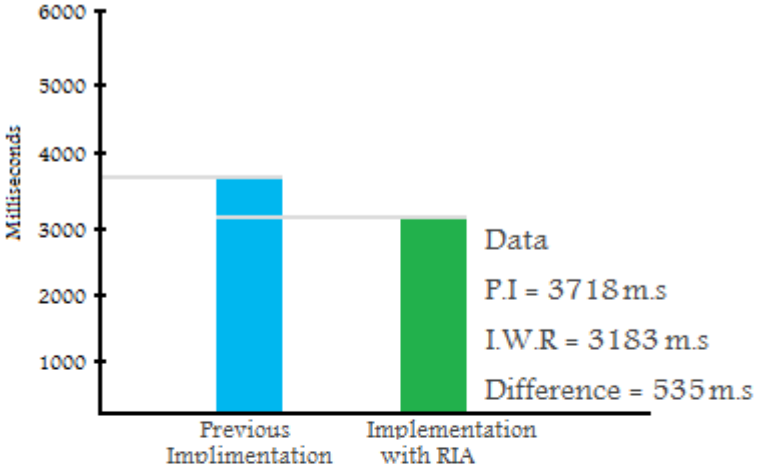
Basically when the page is executed, user data is retrieved by ActiveX, by making a request to XMLHTTPRequest. In case of a success, value "200" *[20]* would be returned. If the return value is "200", then the actual HTML content of the page will be loaded which is implemented as a part of a CSS (Cascading Style Sheet) div element. the code above shows the div element identified as "loginDetailsArea" being loaded. Once the div is loaded successfully, the login page will appear in the browser ready for a login.

This whole process and the code might be a bit much to understand and grasp at once, but as a developer it is pretty easy and a very neat concept.

6.5 Study the results and standardize improvement

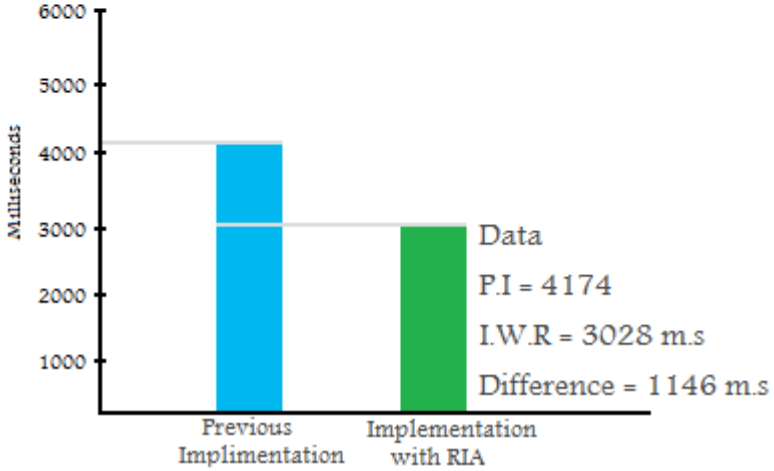
A custom testbench had to be crafted to measure the time between the call and the page having been fully loaded. Being a custom built testbench also means that the results may not be 100% accurate.

The test was conducted by noting the time from the point of running the application by pressing “Enter”, to execute the URL supplied to the browser, until the login page appeared and had been fully loaded on the screen. The same procedure was executed using the old login page and the new login page, using RIA, three times each.

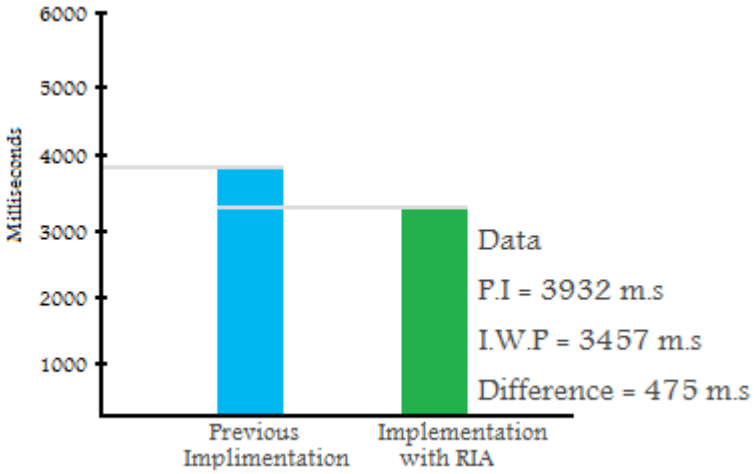


6.6 Results

The results of all three tests show an impressive improvement. The timing of the page being loaded with RIA technology has shown improvement on execution, which clearly shows that the use of RIA technology is one of the best ways of implementing an agent desktop systems.



The execution times varied with each test run, but the login page implemented using RIA technology was ahead in all attempts. In the first and third test the difference was just 535 and 475 milliseconds respectively, which might not seem like a significant difference, but considering that there can be about 500 agents in a contact center using GWT and performance is the primary concern today, a cut of just 475 milliseconds, when multiplied by the number of clients active at any given time would quickly have a noticeable positive impact in GWTs performance.



In the second test the difference of page being loaded was 1146 ms, a quite distinct performance improvement.

7.0 Post prototype discussion

7.1 Difficulties during the prototype

Although no obvious problems were discovered or experienced during the implementation of the prototype, and the technologies which were being used seemed to be sufficient for the implementation, it is still difficult to guarantee that no problems will be encountered while implementing some of the more complex parts. This is due to the fact that there are several aspects of a web application which may not work well with RIA specific technologies. One of the possible reasons why no limitation of the web-based application was found in could be because of the fact that the implemented prototype was very small in size, and was only intended to measure the performance of the small prototype.

7.2 Point of Focus

Since the core purpose of this thesis was to observe the performance difference of RIAs and traditional web-applications, and the thesis focused on answering the core question by formulating a couple of supporting questions, it is apparent that some other factors which could be taken into account were left out. This might not limit or affect the discussion of the paper, but overall it would be a good idea to take other factors into account as well, such as the limitations which could have affected the user interface level. This, since one of the biggest reasons for adopting RIAs is the usability factor; it would be good to study how RIAs can win the race on the grounds of usability as compared to traditional web development methods, and what limitations can be faced.

7.3 Justification of why RIAs should also be studied on the grounds of strong usability

Supposing that two different implementations of the same application, using different technology stacks solve the same problem in almost the same way (not taking performance into account), providing a better user experience on the basis of usability can be a very important success factor.

On the other hand, another factor justifying the comparison of technologies, based on usability and user interaction, is the new era of web applications, where already a lot of discussions on creating better and richer web applications are in progress. By making comparisons between the usability factor, and taking in account the user experience quality, the sole purpose of this discussion is to shed some light on the term “richer”, and what it actually means in the context of user interface development.

One could argue that it is not at all fair to make comparisons on the basis of usability and user experience qualities, which depend heavily on the type of the application under discussion, as well as the intended audience of the application.

7.4 View technologies

Several technologies exist for enhancing the user experience and developing richer applications, though user interfaces with RIA were not the focus of this thesis, Java was used for prototype implementation. The related technologies listed below are interesting alternatives, which could be used to implement richer user interfaces instead of Java, but further investigation is however outside the scope of this thesis.

- Flex
- OpenLaszlo
- Silverlight
- XUL – XML User Interface Language

7.5 Limitations of web browsers and future enhancements

A working Internet connection is needed for a web application to run, which is one of the biggest drawbacks of a standard web application, and in order to work partly without a connection or offline, applications need to be capable of storing resources locally. Local storage is one of the features which is currently being discussed by some of the biggest Internet browser companies, Google have already taken a step forward by releasing a beta version of the Google Gears framework, which is its web browser extension. Apart from the module which allows local storage for web applications, Google Gears have also included a database module and a WorkPool module as well. The database module uses the SQLite database and exposes a Javascript API for storing and retrieving data by executing SQL statements.

The above described browser improvements are not yet standard. The beta version of Google Gears is available for download; It would not be a far leap of the imagination to believe that these improvements will reduce the gap between the web and desktop applications even more.

8.0 Conclusion

New web application technologies and implementation methods are being launched each year focusing on improvement, reduced complexity and maintenance methods. RIA is one of the technologies which have been in this race for a while.

RIA is still young, and has not been completely accepted as an approved method for being used in any kind of web application. Many tests are being made using RIA within different types of web applications. In this paper the author has contributed to this research by measuring the performance of GWT implemented through the RIA specification. Further implementation of the login page in GWT using the RIA specification and AJAX has shown impressive results performance wise.

From the kind of test which is performed, one can easily raise the question regarding the nature of the test, and whether it's enough to prove the point that GWT login page implemented with RIA is more efficient than the one implemented with the traditional web technologies. Well the answer could be yes and no, it's a Yes if we are just talking about the technologies them self and not the things around them (for example the current load on the server), but it might be a No if we start considering to many other things, which would always make the claim difficult to prove, and which is one of the reasons why the author haven't gone into too many technical details and focus have been kept on explaining the terminology, prototype is just to support the argument.

Testing is always a challenge, and how much testing is enough totally depends on the context or the problem, but by observing the results of a specific part, conclusions or hypothesises can be formed that if the same technology is used in the same manner in other parts, there should be a performance improvement there as well. In this case, after implementing the GWT login page with RIA, and noting the performance improvement, it is not unlikely that by implementing other parts of GWT using the RIA specification, significant performance boosts would follow.

It should also be noted that efficiency of a product is not just limited to its performance time wise, such as in case of RIAs user interfaces and experience is one of the biggest efficiencies which could be achieved, the paper have addressed this a little bit in chapter 7, which could be turned into a whole new thesis of it self just focusing on the user interfaces with RIA.

I would conclude this paper by saying that my contribution in this paper has been to introduce the readers to a new technology stack of RIA, and how effective it could be to develop web applications with RIA, I have kept my focus on Contact center soft phones. I have made an attempt to design and implement a small part of traditional web based soft phone with RIA technology, tested it, which produced impressive results. While doing so, in the "Literature Review" section I have introduced the readers with some of the facts and success of RIA in the market, which has been adopted by many companies for their web based application. With each and every technology around, with benefits there are always some limitations, I have introduced the readers with some of the limitations of RIA's and how can they be handled. After watching the results of the tests I performed, I

can say that RIA technology stack is a good way of implementing web-based agent desktop applications to gain efficiency in different areas, but a good effort needs to be put in to think the best way to using RIA for such applications.

List of acronyms

- AJAX (Asynchronous Javascript And XML)
- API (Application Programming Interface)
- CSS (Cascading Style Sheets)
- DHTML (Dynamic HTML)
- GUI (Graphical User Interface)
- GWT (Genesys Web Toolbar)
- HTML (HyperText Markup Language)
- IIS (Internet Information Services)
- JRE (Java Runtime Environment)
- MVC (Model View Controller)
- RIA (Rich Internet Application)
- SQL (Structured Query Language)
- UML (Unified Modelling Language)
- XML (eXtensible Markup Language)

References

Internet

[1]. http://www.xulplanet.com/tutorials/whyxul.html

[2]. (mathias craig) Open access barriers: An Action Research

[3]. Caro-Bruce, C. The Action Research Facilitator's Handbook. National Staff Development Council

[4]. Calhoun,E.F (1994). How to use action research in the self-renewing school. Alexandria,VA: ASCD.

[5]. Action research by Eileen Ferrance (Northeast and islands Regional Educational Laboratory At Brown University)

[6]. What is "Push Technology"? Prepared By Kenneth W. Umbach, Ph.D. 1997

[7]. O'Rourke, Cameron (2004). A Look at Rich Internet Applications.

[8]. Rich Internet Applications: IDC White pape http://www.macromedia.com/platform/whitepapers/idc_impact_of_rias.pdf

[9]. Rich Internet Applications: Design, Measurement, and Management Challenges, Keynote Systems, 2006

[10]. Rich Internet Applications: Extraordinary interactive experiences by outsmart, (http://www.getoutsmart.com/rich-internet-applications.pdf)

[11]. Financial Institutions Need Rich Internet Apps Rich Internet Applications Better Enable Complex Financial Interactions by Ron Rogowski with Bruce D. Temkin and Steven Geller, 26th September 2007.

[12]. Rich Internet Applications 101: A Primer for Marketing Agencies & Multimedia Developers Integration New Media, Inc. By Andrea Simmons, a White Paper from Integration New Media

[13]. The Business Case For Rich Internet Applications by Ron Rogowski, March 12, 2007

[14]. Forrester's Customer Experience Peer Research Panel Survey in 2006

[15]. Ideal systems Genesys Web Toolbar Functional Description Version 7.5.000, 2007

[16]. IT Design, Architecting and Development, Skitsanos.com Monday, October 01, 2007

[17]. Simon Whatley, Rich Internet Applications - A Background (http://www.simonwhatley.co.uk/rich-internet-applications-a-background)

[18]. Source of information was many random websites, main one was (http://kyayagroup.googlepages.com/comet)

[19]. What is AJAX (http://en.wikipedia.org/wiki/AJAX)

[20]. HTTP Status codes (http://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html)

Literature

- Loosley, Chris. Rich Internet Applications: Design, Measurements and Management Challenges. Keynote Whitepaper (2006)

- Marinilli, Mauro. Professional Java User Interfaces. Wiley (2006)

- Schwerin, Rich. Getting Rich with AJAX. Oracle Magazine (2006)

- Zeldman, Jeffrey. Designing with Web Standards. New Riders (2003)

- Weiss, Aaron. WebOS: Say Goodbye to Desktop Applications (2005)

- Keith, Jeremy. Bulletproof AJAX. New Riders (2007)

- Handy, Alex. Where Are the Rich Internet Applications Written in Java? Software Development Times (2005)

- Cederholm, Dan. Web Standards Solutions, The Markup and Style Handbook (2004)

Appendix 1

```
function initRequestObject() {
    var xmlHttp;
    if (window.XMLHttpRequest) {
        xmlHttp = new XMLHttpRequest();
    } else if (window.ActiveXObject) {
        xmlHttp = new ActiveXObject("Microsoft.XMLHTTP");
    } else {
        alert('Problem creating the XMLHttpRequest object');
    }
    return xmlHttp;
}
```

```
</form>

<table align="center" border="0" >
    <tr>
        <td><div id="loginDetailsArea"></div></td>
    </tr>

    <tr>
        <td class="errorLogin" align="center"><html:errors locale="locale"/></td>
    </tr>
     <tr>
        <td class="errorLogin" align="center" id="tdJavascriptError"></td>
    </tr>
    <tr>
        <td align="left">
            <script type="text/javascript">
                var bar1 = createBar(190, 10, 'backgroundBlue', 0, 'black', '#4A5563', 100, 7, 3, "");
                bar1.togglePause();
                bar1.hideBar();
            </script>
        </td>
    </tr>
</table>
```

```
function processStateLoginDetails(xmlHttp,status,num){
// alert(xmlHttp.readyState + ", " + xmlHttp.status);
    if (xmlHttp.readyState == 4 && xmlHttp.status == 200){ //Complete OK response
        try{
            //alert(xmlHttp.responseText);
      document.getElementById('loginDetailsArea').innerHTML = xmlHttp.responseText;
      var agentLogin = document.forms["inputForm"].elements["agentLoginTextField"].value;
      if(agentLogin == null || agentLogin == ""){
          status = true;
          num = 1;
      }
            <%if((languageForm == null) && (loginForm == null)) { %>
            changeLang();
            <%}%>
            if(num == "" || num === null){
                num = 3;
            }
            builtInputForm(status, num);


        } catch (e){
            alert(e.message);
        }
    }else{
        document.getElementById('loginDetailsArea').innerHTML = " error retrieving logindetails";
    }
}
```