A Framework to Develop AJAX Based Web Applications

Sneha Ankurkar1, D. M. Khatwar2

P.G. Student, Dept. of Computer Science & Engineering, Agnihotri College of Engineering, Wardha, India1
Professor, Dept. of Computer Science & Engineering, Agnihotri College of Engineering, Wardha, India2

ABSTRACT: To develop Rich Internet Application, AJAX provides very strong script based approach. But due to various complexity in the development process, it gets less attention from new developers. If there is some simple rule-set, some guidelines defining the use of AJAX in simple manner, then that may help application developers to build Rich Internet Application with less complexity and rapidly. With that aim in mind, this paper proposes some guidelines and best practices regarding the AJAX based Rich Internet Application development. The proposed best practice strategies are to deploy Rapid Application Development methodology through initial planning of application, basic development, and alteration after deployment, maintenance and expansion. Through these best practices, the rapid and less complex realization of AJAX based Rich Internet Applications can be possible. This work proposes a general model of an architecture for the AJAX application, with some best practice rule-set.


1. INTRODUCTION

As the name suggests, Rich Internet Applications (RIAs), provide rich user experience, in terms of enrich Graphical User Interfaces (GUIs) [1] and faster request responses [2] than the traditional web applications with page sequences causing limited interaction and large response time. RIAs have ability to replace the classic desktop applications. With the help of HTML 5 and CSS 3, RIAs become more powerful and portable [3]. Even with all these advantages and features RIAs are facing lack of development strategies and development methodologies [4]. To develop RIAs, there are two approaches: AJAX based approach and proprietary plugin based approach [5].

AJAX is not a technology, it’s a technique which uses various mature technologies like JavaScript, HTTP request-respond model, XML, HTML and CSS [7][8]. The reason AJAX based RIAs are preferred over traditional proprietary plugin based applications is its capability of communicating asynchronously between web GUI and server. This feature gives user rich experience in terms of faster interactive and responsive web applications since AJAX based application does not require to refresh whole page, it can update section of webpage partially [9][10]. AJAX technology is not very difficult to understand, but its realization in RIA is bit complex [9], [11] and difficult to realize [6]. Many solutions have been proposed in the form of various tools and framework by industry and academy. However there is still immaturity in this technology [11] and it absences of models [12] and building formalism [8].

Programming engineering today has a tendency to be bolstered by Rapid Application Development (RAD) techniques, because of the reason that all the partners occupied with the advancement process, as to see a quality yield quickly. RAD is a way to deal with manufacture frameworks which joins Computer Assisted Software Engineering (CASE) devices and systems, client driven prototyping, and stringent undertaking conveyance time limits; into an intense, tried, solid equation for top-indented quality and efficiency [10]. Just saying RAD definitely raises the nature of completed frameworks while lessening the time it takes to manufacture them [10]. RAD is upheld by effective CASE tools which makes it conceivable for designers to make frameworks much speedier than at any other time; also, the achievement of any RAD undertaking is principally subordinate upon the devices utilized [10]. RAD devices areimaginatively moving towards in two territories: dialect based programming options and programming outline support [11]. The models or the outlines which determines the applications from a conceptual perspective have related CASE instruments for mechanization procedures, programmed code era and computer aided planning and analysis; and these tools create executable code [12], [10].

Our research depends on, distinguishing the explanations behind the complexities, and outlining solution(s) to conquer the troubles, in AJAX based RIAs designing; keeping in mind the end goal to expand the backing for RAD.
This paper portrays the technique we used to recognize the complexities; and talks about the examination of the learning we have picked up from reviews; and the solutions we propose. We survey some related work to sum things up and toward the end of the paper we close our discoveries and propose some future work, which can be useful to proceed with the examination moreover.

Methodology

We utilized three unique techniques, to pick up the learning, comprehend the foundation, and experience the environment; required for outlining arrangements and proceed with the research. To assemble the information about the history and the foundation of RIAs and AJAX related looks into – as the principal strategy – we led literature survey to study, what the specialists have said and what are the related scrutinizes have been finished.

As the second technique – to make the knowledge picked up in the literature review more grounded, and to affirm the learning is exceptional and related – a cross-sectional study is conducted, focusing on people occupied with RIA designing; utilizing arbitrary inspecting strategy. Utilizing factual strategies, this paper broke down the knowledge gather; and deduced the confirmation for the discoveries of the literature review, and some other essential actualities identified with the complexities in AJAX RIA advancement.

Parallel to the studies and literature overview – as the third technique – this work directed a progression of experiments constantly from the earliest starting point of the research. These experiments help to experienced, the complexities and challenges what have learnt from the studies, and separate a few certainties which influence those complexities by taking a shot at various sorts of RIA undertakings. These experiments help to recognize the areas to be enhanced, and to experiment with different systems to build the acknowledgment of the AJAX adoption inside of the RIA to make the development process smooth. The prototype model was a model based iterative and incremental, yet for the duration of the life cycle in a few emphases this work changed the task and the knowledge in past undertakings emphases was utilized constantly.

II. RELATED WORK

This section talks about some closely related work. While researching on this topic, several tools, techniques are being studied, some of them which relate to the proposed work are mentioned here.

[17] introduces the simple framework to adopt AJAX by introducing JavaScript based framework. This framework wraps the XMLHttpRequest object creation codes inside a functions, which can effectively reduce the redundant code in AJAX features implementation. This paper provides the framework based solely on the JavaScript, hence it is very handy tool for new developers. This framework uses established technologies such as jQuery, JavaScript which are free which provides very efficient wrappers for AJAX with many more other features. But this framework is not recommended to implement since it is efficient in implementing the basic AJAX features within RIAs. Though the learning curve of this framework is not much too high, in case of complex AJAX feature realization, the framework is failed to provide support in order to implement AJAX features efficiently within RIAs.

The authors of the paper “jQuery-based Ajax General Interactive Architecture” [3] introduces an effective solution, they call it JAGA, which can greatly reduce the amount of code, and as the authors interpret “the reduction of code means the reduction of difficulty to use”. The JAGA principle has its own architecture and the sub modules in the modules of this architecture use specific patterns and data structures. Because of this complex nature, we think this solution has a great learning curve. And this architecture is based on jQuery, which limits the tools selection and usage of the engineer.

The solution introduced in [3] is called “JAGA”. The thinking behind developing this JAGA or jQuery-based Ajax General Interactive Architecture, is less the code less the trouble! The JAGA principle has its own particular design and the sub modules in the modules of this engineering use particular examples and information structures. On account of this perplexing nature, we think this arrangement has great learning curve. Also, this design depends on jQuery, which confines the apparatuses determination and use of the architect.

A good generalized architecture is proposed in [9] for AJAX realization in RIAs, the architectural solution provide is called SPIAR. It acquires a few components from Google's GWT, Backbase and Echo2; yet does not taking into account them. Since this is another engineering style and it has an alternate part structures, the expectation to absorb information will be high; and the reception of the style can present another level of multifaceted nature. SPIAR is intended for single page RIA improvement and we don’t think it gives enough reflection to general AJAX based RIAs.
After analysing the data and records gathered in cross-sectional survey, one major fact comes out. Majority of people don’t have firm opinion on the AJAX general architecture, that whether it is easy to implement or not! Fig.1 shows the general architecture of AJAX.

In the survey of CASE tools implementation, it is found that nearly 80% of individuals use one or more IDEs (Integrated Development Environment), nearly 60% people use code generator, quality assurance, libraries, frameworks and designing tools. In summarized manner, it can be stated that the majority of developers uses the CASE tools and architectures to implement AJAX based RIAs. Even though this is the case, most of them express the difficulty in implementing more than two features a page. Fig. 2 shows the difficulty level of development stage in graphical format.

From above discussion, it is clear that, having a good understanding of AJAX architecture or implementation of CASE tools and architecture, it not sufficient to reduce the complexity and difficulty in realizing the implementation of AJAX based RIAs.

From the results of cross-surveys and literature overview, it can be concluded that the development process of RIAs based on the AJAX technology is suffering from complexities [3], [13], [5]. These complexities are result of difficulties in realizing the adoption of AJAX structure within the framework of RIA [3] and the reason behind this is lack of standard architectural framework for AJAX implementation within RIAs [13]. Many researchers and scholars propose various tools, framework and other solutions to reduce these complexities, which have seen in literature survey section.
These tools and techniques have their own pro and cons, which are subjected – out of the scope of this paper – to be discussed. And from literature survey, some observations have been made, which are noted down here:

- First of all it is very important to realize the system architecture properly [4], [14]. A sound architecture can improve system performance and also reduces system complexities.
- Considering architecture as pillar, complete system design should be based on it.
- And while implementing these designs, utilizing and combining proper CASE tools with the development process can facilitate the RAD adequately.

The solution proposed in this work is a general architecture based solution for realizing the AJAX based RIAs. This architecture was designed and tested throughout the series of experiments, in multiple scenarios. In each scenario, we experiments under different conditions, that way focused on different set of complexities and difficulties related with them, which need to be solved; and also trying to establish designing parameters, that needs to be addressed and to be worked upon; in order to reduce complexity in realizing AJAX based RIAs. From these set of experiments and observation derived from them; this work proposes the general framework for the development of AJAX based RIAs. Since this work is driven by the available architectural patterns, it can be said that the proposed architectural guidelines and models will likely be a general novel hybrid structure for AJAX based RIAs. The features of proposed model is as follows:

1. The mainframe of the architecture should consist of only two primary layers: client-server. And to support these layers further expansion of additional layers should be done as shown in fig. 3. These layers may contain Databases (DB), Web Services (WS) or Enterprise Service Buses (ESB). Main motive to design architectural structure like this is to increase degree of freedom to incorporate with available design patterns without additional learning curve, while realizing RIAs with AJAX.

2. The model-view-controller technique should be adopted in RIA realization using AJAX. Various parts of web application is developed using different programming languages, tools and techniques. If we consider the basic example of traditional web application, the view part or the front end of web application is developed mostly using HTML and CSS. The behaviors or the Controllers are developed using client-side JavaScript or server-side language. The business logics are mostly developed using server-side implementation techniques. MVC provides the freedom of designing and developing the whole system or application in majorly three separate parts. By developing RIA in MVC way, implanting AJAX technology becomes realizable. With MVC models page view, data flow, background controls and http request-respond and event update etc. can be implemented with AJAX efficiently. In fig. 4 MVC modeling is shown.

---


---

Copyright to IJIRCCE

DOI: 10.15680/IJIRCCE.2016.0404156 7137
3. Since the focus of this work is to realize the AJAX in RIAs, the general architecture should be self-explanatory that how AJAX is adopted and implemented in the RIA. The features of RIAs can be categorized into two groups – AJAX based and non-AJAX based. The purpose of this grouping is to identify and understand the techniques implemented to realize these rich features and also how these features are going to integrate in webpage. Fig 5 shows the general adaptation of the AJAX under RIA. The general idea is to realize the asynchronous data flows (AJAX based) and synchronous data flows are integrated in the same structure.

4. The proposed architectural framework is general and it should not base on any particular technique, platform, programming language, software library, so that it can be realizable in any platform with cross-over techniques and technologies. Hence the aim is to minimize the learning curve of this architecture for developers. The architecture should support wide range of platforms, tools and techniques adjusted to the stakeholders imperatives on platforms and devices as well.
5. As stated above and as discussed earlier, general architecture should increase the realization of all parameters of AJAX in RIAs. This generalized architecture should help to improvised and speed-up the implementation. At the point when the unique acknowledgment gave by the architecture is adequate in proficient outlining and improvement, it might encourage expanding the CASE device joining, henceforth the RAD abilities.

IV. SIMULATION RESULTS

The simulation studies involve the deterministic small network topology with 5 nodes as shown in Fig.1. The proposed energy efficient algorithm is implemented with MATLAB. We transmitted same size of data packets through source node 1 to destination node 5. Proposed algorithm is compared between two metrics Total Transmission Energyand Maximum Number of Hops on the basis of total number of packets transmitted, network lifetime and energy consumed by each node. We considered the simulation time as a network lifetime and network lifetime is a time when no route is available to transmit the packet. Simulation time is calculated through the CPUTIME function of MATLAB. Our results shows that the metric total transmission energy performs better than the maximum number of hops in terms of network lifetime, energy consumption and total number of packets transmitted through the network.

The simulation results involve the deterministic comparative test results of a web based application build upon proposed AJAX framework and the other existing web applications which based on the traditional web technology. For the simulation purpose we build a healthcare web application based on the proposed framework and then tested it against the existing medical web applications and healthcare websites. The comparative results are shown in the figure 6.

Fig. 6. Comparison Reports

The figure 6 above shows the comparison results of the AJAX framework based health care app and three other health related web apps. Our web app is at the extreme left in the fig. 6. Now let’s analyze the comparison results. We upload our web app to a dummy web server and then use online tools for checking its comparative performance. First index “PageSpeed Grade”, shows the relative loading speed of the web page. It is calculated by considering the size of the page, internet speed of the user computer and loading time of the page on the user’s computer. Second index is “YSlow Grade”. This index shows that how much faster a webpage respond to the user interaction with it. For example, if user click somewhere on the link on web page, then how much time it takes to complete the action of loading the new page, or if there is a search box on the web page, then is it autocompleting? All these things are measured by this index. “Page load time”. As the name suggest, it shows the amount of time require to load a page. And the last index shows the number of request browser has to make to the server, before complete web page appears. Now as seen from the figure 6, our model web application is excelling in all the above described categories.

Now we want to make sure that our framework is really user friendly and fast enough to load on the slow internet speeds. So we tested it on the Google Developer platform. Figure 7 shows the result from the Google.
The figure above shows the Google Developers’ result of our web app, which is 91 points out of 100. This is very descent score. This score signifies the user experience as well as speed on a web app.

V. CONCLUSION AND FUTURE WORK

RIA building is experiencing complexities and the AJAX reception is viewed as troublesome. A decent comprehension of the AJAX general engineering or use of different CASE apparatuses have not helped decreasing the troubles, so we can’t expect a sufficient backing in RAD of RIAs by them. Since a decent engineering can help in expanding the acknowledgment of the framework and diminish the intricacies, we propose a general cross breed RIA design to build the realization of AJAX based RIAs and decrease the complexities. We have distinguished some vital elements, which ought not out of the ordinary from such a general RIA engineering, and we recommend that these elements will be helpful to outline the proposed general cross breed RIA design to expand the realization of AJAX selection and the other general components in RIAs. This general engineering might help with complete and exact RIAs outlining, thus expand the capacity of joining CASE devices, supporting the RAD methods sufficiently.

In future we hope to proceed with our trials and completion the configuration of general mixture RIA design, which gives the elements examined in this paper. The assessment of the new broad design against the traditional RIA building in future is additionally important. Recognizable proof of the use requirements and impediments of the proposed RIA engineering and raise dialogs on the most proficient method to defeat the limitation(s) if any, is key as well.

REFERENCES

15. R. Shirvani and N. Modiri, “Software architectural considerations for the development of secure software system,” in Proc. the 7th International Conference on Networked Computing (INC), 2011, pp. 84-88.

BIOGRAPHY

Sneha Kailas Ankurkar is a M. Tech scholar in Dept. of Computer Science & Engineering, Agnihotri College of Engineering, Wardha, India. She has received her Bachelor’s degree from Nagpur University in the field of Computer Science. Her research area includes web technologies, data mining, and internet of things (IoT).