Essential Features a General AJAX Rich Internet Application Architecture Should Have in Order to Support Rapid Application Development

Nalaka R. Dissanayake and G. K. A. Dias

***Abstract*—Rich Internet Applications have gained a good demand in web engineering, and AJAX plays a major role as a script based technique to develop Rich Internet Applications, even though its adoption is considered difficult due to various complexities. If the root cause for these complexities can be identified and the difficulties can be overcome, we can simplify the AJAX based Rich Internet Applications engineering and support the Rapid Application Development methodology to produce Rich Web Apps faster, while maintaining the quality. We have identified the importance of a general architecture for Rich Internet Applications – which is capable of addressing the difficulties in AJAX adoption – and we propose some essential features which need to be expected from the general architecture; based on the knowledge gained from a survey and a series of experiments. These features are supposed to be used, to design a general hybrid Rich Internet Application architecture – which increases the support in Rapid Application Development – in our ongoing research.**

***Index Terms*—AJAX, rich internet applications, complexities, architecture, rapid application development.**

I. Introduction

Rich Internet Applications (RIAs) were introduced in the era of Web 2, to facilitate the users with faster web applications, which have rich features and responsive interfaces, over the slow traditional web applications with limited interactivity [1]. Users experience the richness of the RIAs via popular applications like Facebook, Flicker or Google apps, so that they like to feel the same in other Web Apps they consume too, which creates high demand for RIAs. RIAs can be organized into three distinct types [2]. First type is the proprietary plug-in based (Sandbox) approach like in Flash/Flex/Java applets/Microsoft Silverlight. The second type is the well-known Asynchronous Java Script and XML (AJAX) techniques, referred to as script based RIAs. And the third type is the least publicized browser based approaches, like XUL from Mozilla Foundation.

AJAX has already become the most popular and well used technology [3] for developing RIAs; and AJAX is far more than just a trend, it's a powerful approach to build RIAs [4]. The combination of established technologies like HTML, CSS, JavaScript, XML and http request-response model, used to develop AJAX makes it unique and strong [13]; and it is a major breakthrough in the web development area even though it is rather complex to set up [5]. But if its complexities can be realized and overcome, AJAX will be a better technology in RIAs development [6]. There are several perspectives for these complexities and Difficulties related to the AJAX based RIAs engineering. Engineers have to think in a different way to design and develop AJAX applications [7] and the realization is difficult [3]. The Model-View-Controller (MVC) is an essential architectural pattern and in AJAX development the MVC module is not clearly separated, instead they are mixed in the web page [8]. The lack of architectural formalism in AJAX based RIAs is related to all these complexity and difficulties [9] and understanding proper tools and methodologies is recommended. However, immense application of AJAX can introduce significant code redundancy, hence unexpected errors [3].

Software Engineering process today tends to be supported by Rapid Application Development (RAD) methodologies, due to the reason that all the stake holders engaged in the development process, like to see a quality output rapidly. RAD is an approach to build systems which combines Computer-Assisted Software Engineering (CASE) tools and techniques, user-driven prototyping, and stringent project delivery time limits; into a potent, tested, reliable formula for top-notched quality and productivity [10]. Simply saying RAD drastically raises the quality of finished systems while reducing the time it takes to build them [10]. RAD is supported by powerful CASE tools which makes it possible for developers to create systems much faster than ever before; and the success of any RAD project is primarily dependent upon the tools used [10]. RAD tools are innovatively moving towards in two areas: language-based programming alternatives and software design support [11]. The models or the designs which specifies the applications from an abstract viewpoint have associated CASE tools for automation techniques, automatic code generation and computer aided planning and analysis; and these tools generate executable code [12], [10].

Our research is based on, identifying the reasons for the complexities, and designing solution(s) to overcome the difficulties, in AJAX based RIAs engineering; in order to increase the support for RAD. This paper describes the methodology we used to identify the complexities; and discusses the analysis of the knowledge we have gained from surveys; and the solutions we suggest. We review some related work in brief and at the end of the paper we conclude our findings and propose some future work, which can be helpful to continue the research furthermore.

Manuscript received March 28, 2014; revised June 20, 2014.

Nalaka R. Dissanayake is with University of Colombo School of Computing, Sri Lanka (e-mail: nalakadmnr@gmail.com).

G. K. A. Dias is with University of Colombo School of Computing, Sri Lanka (e-mail: gkad@ucsc.lk).

II. Methodology

We used three distinct techniques, to gain the knowledge, understand the background, and experience the environment; needed for designing solutions and continue the research. To gather the knowledge about the history and the background of RIAs and AJAX related researches – as the first technique – we conducted a literature survey to study, what the experts have said and what are the related researches have been done.

As the second technique – to make the knowledge gained in the literature survey stronger, and to confirm the knowledge is up to date and related – we conducted a cross-sectional survey, targeting individuals engaged in RIA engineering; using random sampling method. Using statistical techniques, we analysed the data gathers; and inferred the evidence for the findings of the literature survey, and some other important facts related to the complexities in AJAX RIA development.

Parallel to the surveys – as the third technique – we conducted a series of experiments continuously from the beginning of the research. These experiments helped to experience ourselves, the complexities and difficulties what we have learnt from the surveys, and isolate some facts which affect those complexities by working on different types of RIA projects. These experiments helped to identify the areas to be improved, and to try out various techniques to increase the realization of the AJAX adoption within the RIA to make the development process smooth. The process model was a prototype based iterative and incremental, but throughout the life cycle in some iterations we did change the project and the knowledge in previous projects iterations was used continuously.

III. Discussion

When analyzing the data gathered in the cross-sectional survey, we identified that just 21% of individuals find it is difficult to realize the AJAX general architecture, where the majority 79% find it is not difficult or easy. See the Fig. 1 for AJAX general architecture.

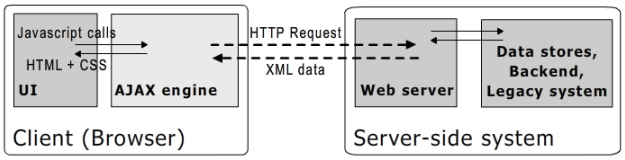

Fig. 1. AJAX general architecture.

Then we took a look at the CASE tools usage, and we found that 79% of individuals use one or more Integrated Development Environment (IDE) tool(s); 58% use some other tool(s) like Code generation tools, Quality assurance tools, Libraries/Frameworks, and Design tools; and 61% of them use a JavaScript framework too. That interprets, the majority uses the CASE tools in the engineering of AJAX based RIAs. However, even though the majority finds it is not difficult to realize the AJAX general architecture and they use CASE tools too, yet 55% of individuals experience that implementing multiple AJAX features (more than 2) in a page is difficult, and 30% says it is very difficult, where just 15% of the individuals express it is not difficult. This condition explains clearly, that a good understanding of the AJAX general architecture or usage of CASE tools/ libraries/ frame works has not been able to reduce the difficulty level in AJAX based RIAs engineering. See the Fig. 2 for the difficulty level of implementing multiple AJAX features (More than 2) in a page.

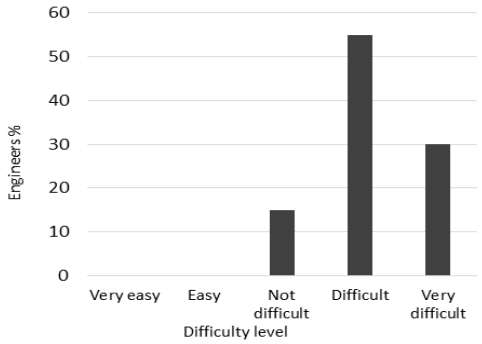

Fig. 2. Difficulty level of implementing multiple AJAX features (more than 2) in a page.

The results inferred from the analysis of the cross-sectional survey evident that the engineering process of AJAX based RIAs are suffering from some difficulties [3], [13], [5]. Actually these difficulties are related to the complexities in realization the RIA structure and adoption of AJAX techniques within the RIA [3], and we think that the main reason is there is a lack of architectural formalism for AJAX based RIAs [13]. To reduce the difficulties and support the development, we have come across – throughout the literature survey – various tools available; and some frameworks and specific architectural based solutions introduced by researchers; where these solutions have their own pros and cons, which are subjected – out of the scope of this paper – to be discussed. And in the literature survey, we noted and filtered some important factors as below, which need to be highlighted here.

- The importance of having a sound architecture to realize the full potential of an application [14]-[16], hence reduce the development confusions.
- The importance of having a good complete system design based on a strong architecture.
- The importance of having a good design combined with the proper CASE tools and best practices to facilitate the RAD adequately.

The proposed solution of our ongoing research, which is a general architecture for AJAX based RIAs, was designed and tested throughout a series of experiments, in multiple versions. In each and every version we concentrated on identifying a different set of complexities and the difficulties related to them, which needs to be addressed; and worked on recognizing and designing the features, need to be provided by the architecture, to offer support for the identified set of difficulties. By analysing the results of these experiments, we constructed a set of features, which are supposed to be expected from a general RIA architecture. Since some of the features are derived from available architectural patterns, we would say that our solution will likely be a general hybrid architecture for AJAX based RIAs, which offers the features specify below.

1) The general architecture should be based on the simple two-tier client-server architecture and support expansion of the layers by adding more tiers or components like files, Databases (DB), Web Services (WS) or Enterprise Service Buses (ESB) easily. See the Fig. 3 for architectural support for layer expansion. It will provide the architect and the system designer the freedom to incorporate available design patterns without an additional learning curve, when designing the RIA.

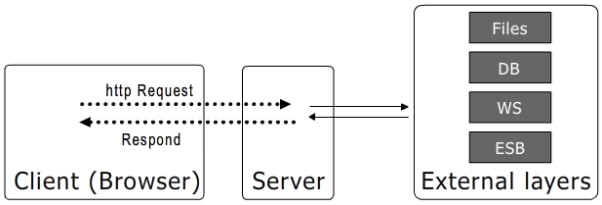

Fig. 3. Support expansion of the layers.

2) We think that the MVC pattern is really useful in Web Development since the various parts of the web application are developed using different languages and techniques. The presentation of the content in web pages or Views and developed 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. Even though the code for MVC is there mostly they are mixed in the web page [8]. We think that the understanding of MVC incorporation and the clear separation of the modules will increase the realization of the RIA structure too. Therefore we propose that the general RIA architecture should incorporate the MVC pattern; provide the realization of how the MVC modules are adopted in RIA; and the data flows like http requests-responds or/and events-updates between the modules should be defined clearly too. See the Fig. 4 for incorporating MVC pattern.

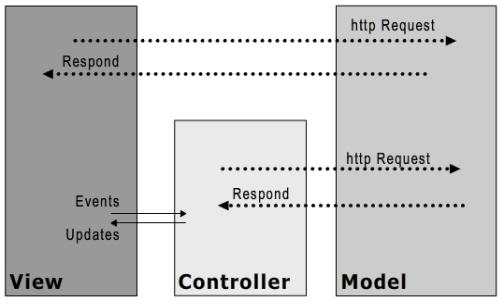

Fig. 4. Incorporate MVC pattern.

3) Since we are focusing on the AJAX based RIAs, the general architecture should explains clearly how the AJAX is adopted and embedded into the RIA. We would like to categorize the rich features into 2 main groups – AJAX-based and non-AJAX-based. In the designing and development of RIAs, it is important to identify and understand the techniques used to implement the rich features, and how these features are integrated into the same application or the same page. The general architecture should increase the realization of how both the AJAX asynchronous data flows and other http request/respond data flows are integrated by providing an abstract overall structural understanding of the RIA. See the Fig. 5 for clear AJAX adoption within RIA.

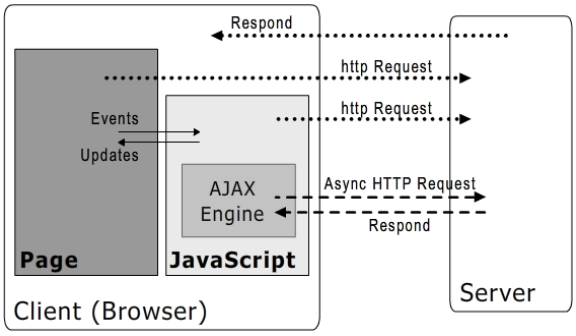

Fig. 5. AJAX adoption within the RIA

4) By all the means, the proposed architectural solution is supposed to be general. It should not be based on any platforms, languages, frameworks or libraries, so that the engineers can continue using their usual tools and knowledge; therefore, when the proposed general architecture is used, the learning curves – caused by the usage of the general architecture – are minimum. This will ensure a support for wide range of platforms, tools and techniques, and will help to be aligned to the stakeholders constraints on platforms and tools too.
5) The main requirement expected by the general architecture is increasing the realization of the all the aspects of AJAX based RIAs as discussed in the above features. Such abstract general architecture should provide a sufficient help in clear and complete system designing and development too. When the abstract realization provided by the architecture is sufficient in efficient designing and development, it may facilitate increasing the CASE tool incorporation, hence the RAD capabilities.

IV. Related Work

While continuing the research, we came across many tools, frameworks, techniques, and some other researches. In this section we discuss in brief about 3 researches directly related to our research.

The paper "AJAX-based Applicable Framework Research and Design" [17] introduces a pure JavaScript based simple 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 a good start for the beginners and this framework can be a handy tool for the developers who use pure JavaScript. Since there are well established JavaScript frameworks like jQuery are available free, which provides very efficient wrappers for AJAX with many more other features, it is not recommended to use a pure JavaScript framework. We think that the learning curve of this framework is not much high, but this framework will not increase the realization of the AJAX based RIAs so that the support provided in a complex project will be less.

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.

SPIAR is the solution provided by the authors of the paper "An architectural style of Ajax" [9], introduces a good general solution for AJAX based RIA complexities. It inherits some features from Google's GWT, Backbase and Echo2; but does not based on them. Since this is a new architectural style and it has a different component structures, the learning curve will be high; and the adoption of the style can introduce a new level of complexity. SPIAR is designed for single page RIA development and we do not think it provides enough abstraction for general AJAX based RIAs.

V. Conclusion and Future Work

RIA engineering is suffering from complexities and the AJAX adoption is considered difficult. A good understanding of the AJAX general architecture or usage of various CASE tools have not helped reducing the difficulties, so we cannot expect an adequate support in RAD of RIAs by them. Since a good architecture can help in increasing the realization of the system and reduce the complications, we propose a general hybrid RIA architecture to increase the realization of AJAX based RIAs and reduce the complexities. We have identified some essential features, which should be expected from such a general RIA architecture, and we suggest that these features will be useful to design the proposed general hybrid RIA architecture to increase the realization of AJAX adoption and the other general features in RIAs. This general architecture may assist in complete and precise RIAs designing, hence increase the ability of incorporating CASE tools, supporting the RAD techniques adequately.

In future we expect to continue our experiments and finish the design of general hybrid RIA architecture, which provides the features discussed in this paper. The evaluation of the new general architecture against the classical RIA engineering in future is also necessary. Identification of the usage constraints and limitations of the proposed RIA architecture and raise discussions on how to overcome the limitation(s) if any, is essential too.

References

[1] F. Piero, R. Gustavo, and S. F. Fernando, "Rich Internet applications," *Internet Computing, IEEE,* vol. 14, no. 3, pp. 9-12, 2010.

[2] J. Farrell and G. S. Nezlek, "Rich internet applications the next stage of application development," in *Proc. the ITI 2007 29th Int. Conf. on Information Technology Interfaces*, Cavtat, Croatia, pp. 413-418, 2007.

[3] J. Li and C. Peng, "jQuery-based Ajax General Interactive Architecture," in *Proc. 2012 IEEE 3rd International Conference on Software Engineering and Service Science (ICSESS)*, 2012, pp. 304-306.

[4] Z. J. Lin, J. Y. Wu, Q. F. Zhang, and H. Zhou, "Research on web applications using Ajax new technologies," in *Proc. International Conference on MultiMedia and information Technology*, 2008, pp. 139-142.

[5] S. Salva and P. Laurencot, "Automatic Ajax application testing," in *Proc. Fourth International Conference on Internet and Web Application and Services*, 2009, pp. 229-234.

[6] N. Dissanayake, G. Dias, and M. Jayawardena, "An analysis of rapid application development of AJAX based rich Internet applications," in *Proc. International Conference on Advances in ICT for Emergin Regions (ICTer)*, 2013, p. 284.

[7] J. S. Zepeda and S. V. Chapa, "From desktop application towards Ajax web applications," in *Proc.4th International Conference on Electrical and Electronoc Engineering (ICEEE 2007)*, Mexico City, Maxico, pp. 193-196, 2007.

[8] D. W. Cheung, T. Y. Lee, and P. K. Yee, "Webformer a rapid application development toolkit for writing ajax web form applications," in *Proc. Distributed Computing and Internet Technology. 4th International Conference.*, 2007, pp. 17-20..

[9] A. Mesbah and A. V. Deursen, "An architectural style for ajax," in *Proc. the Working IEEE/IFIP Conference Software Architecture*, 2007, pp. 9.

[10] C. Inc, "What is Rapid Application Development?" 2002.

[11] J. C. Zubeck, "Implementing reuse with RAD tools' native objects," *Computer*, vol. 30, no. 10, pp. 60-65, October 1997.

[12] M. Linaje, J. C. Preciado, and F. S. Figueroa, "Engineering rich inernet application user interfaces over legacy web models," *Internet Computing*, vol. 11, no. 6, pp. 53-59, November-December 2007.

[13] L. D. Paulson, "Building rich web applications with ajax," *Computer*, vol. 38, no. 10, pp. 14 - 17, Oct 2005.

[14] D. Hough, "Rapid delivery: an eveolutionary approach for application development," *Ibm System Journal*, vol. 32, no. 3, pp. 397-419, 1993.

[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.

[16] M. Lindgren, C. Norstrom, A. Wall, and R. Land, "Importance of software architecture during release planing," in *Proc. Seventh Working IEEE/IFIP Conference on Software Architecture*, 2008, pp. 253-256.

[17] B. K. Ming and C. G. Guo, "AJAX-based applicable framework research and design," in *Information Science and Engineering (ICISE), 2010 2nd International Conference*, Hangzhou, China, 2010.

Nalaka Ruwan Dissanayake was born in Anuradhapura Sri Lanka in 1982. He is working as an assistant lecturer in software engineering at Informatics Institute of Technology (IIT), Colombo 6, Sri Lanka; who is also reading for an M.Phil program as a student of University of Colombo School of Computing, Colombo 7, Sri Lanka. He graduated as a bachelor of science with a special honors degree in software engineering from Sri Lanka Institute of Information Technology (SLIIT), Sri Lanka on 2007. Nalaka started his academic career as a student instructor in SLIIT when he was studying the final year as an undergraduate. After completing the degree he continued as an Instructor in SLIIT. Since then up to date Nalaka has worked in both state and private universities and institutes SLIIT, Rajarata University of Sri Lanka (RUSL), IIT, SLIATE, ESoft as an academic personal over six years. With the experience he has gained over a year in industry as a Software Designer, he works on Research and Development parallel to his studies and academic career. With a keen interest in researching areas such as Software Engineering, Architectures, Rapid Application Development and Rich Internet Applications, Nalaka spends his free time experimenting for new techniques and strategies in Rich Internet Application design and development.

G. K. A. Dias received his BSc in physical science (1982) from the University of Colombo, Sri Lanka, postgraduate diploma in computer studies from University of Essex, UK (1986), and MPhil in computer science from University of Wales, UK (1995). He is currently a senior lecturer with the Department of Communication and Media Technologies of the University Colombo School Computing (UCSC). He is also the coordinator of the M.Phil program at the UCSC. He is a member of the computer society of Sri Lanka, Sri Lanka association for Advancement of Science (SLASS) and a member of the ACM the world's largest educational and scientific computing society. His research interests are computer aided software engineering, multimedia for education, modeling and simulation.