# COMPARISON MATRIX FOR WEB HCI

Kais Samkari
Msc at AI&NLP Dept. Faculty of Information Technology,
Damascus University
Damascus, Syria
ksamkari@el-ixir.com

Ammar Joukhadar
Associated Professor, Faculty of Information Technology,
Damascus University
Damascus, Syria
ammarj@scs-net.org

**View tier is one of the three basic layers of any web based application designed using the common MVC pattern. From the end user point of view, the whole application is seen from this layer. During the project development process, View seems to be the most time and effort consuming part (almost 80% of the whole process), and that is true because of its fundamental role which covers everything related to the conversation workflow with the end user. Today, there are a lot of available web presentation technologies and frameworks that all aim at facilitating the construction of View tier, such as: Apache Tapestry, Apache Struts, JSF, Microsoft ASP.NET, Ajax and others. The goal of this paper is to determine criteria which enable the comparison of these technologies and frameworks, and applying it on Apache Struts, JSF, Ajax and eliXir presentation tier.**

***MVC, web presentation framework, conversational workflow, presentation tier***

## I. INTRODUCTION

Enterprise applications involve a lot of business entities, each entity needs:

- Pages allowing the creating/editing/viewing of the entity
- Pages allowing the search for the entity, and showing the search result for the entity
- Pages for the business operations and their parameters applied on the entity

This means six GUI per entity. If we have 100 entities in the application, this means we need to design 600 different pages. But, business entities don not live apart, i.e. there are business relations among them, and so, this increases the number to almost 1000 different pages. In addition to this, each entity is associated with several business processes (i.e. as a data object) which demand many other views, one for each task, focusing on different parts from the entity, and which in turn adds a number equal to the number of tasks in each process (such as: create, validate, finish, …). According to this, the web based application's provider needs to build the view layer (presentation tier) that is responsible of managing the conversion workflow with the end user on the web. Fortunately, there are lot of available technologies and frameworks which all aim at facilitating the construction of view tier such as: Apache Tapestry, Apache Struts, JSF, Microsoft ASP.NET, Ajax and many others, but what remains is answering the question: "Which technology or framework to choose?".

This paper answers the question by first discussing several related works in Section 2. Our approach in extracting the criteria and their definitions are listed in Section 3. The comparison is made in Section 4. Section 5 includes the conclusion.

## II. RELATED WORKS

Among the problems identified by Ginige and Murugesan [1] into the difficalties of building web applications were the fact that information contained in the web application can change rapidly, and the structure and functionality of the application can also change over time, making maintenance more difficult and time-consuming. Web presentation technologies and frameworks change very rapidly and according to Amanda Quek and Albert Alderson [2] it is easier for people to accept evolution as compared to revolution which greatly increases the speed of development, and which causes interesting effects in the way tools are chosen for development nowadays.

The choice of framework depends on the project [3]. The following factors need be considered when choosing the frame work: the type of application, the size of the project, scope and requirement for future enhancements, and availability of resources and experts to support the selected framework. Apache Struts, Spring and JSF were compared, and as conclusion, one should use Spring if there is problem with technical skills to learn this new technology.

In reference [4], the article introduced RIAs (Rich Internet Applications), discussed current UI technologies, and evaluated them considering the following factors: richness of the UI, complexity, flexibility and componentization, refreshing the page, security, support for basic Web paradigms, tooling and usability.

Other articles [5] evaluated different frameworks according to: Ajax Support, bookmark-ability, validation, testability, post and redirect, internationalization, page direction, community

and support, tools, marketability of skills, job count. The choice between these applications depends on: what the type of application is, ease of development, project community, project future and roadmap, maintenance and technical features.

Reference [6] discussed factors that need to be considered when choosing an application framework, including but not limited to: suitability for specific business needs, developer productivity, performance, support and community activity, technology maturity, developer prowess and business relationships.

## III. CRITERIA

To extract the criteria, we have to consider the main players which are the end user of the web application, and the development team that provides the web application. The end user is the client of the application and the one who decides if the application is going to live or not. Since the whole application to the end user is represented by the view layer, it is very intuitive to consider his point of view represented as a set of non-functional requirements.

On the other hand, the project development process represented by the team providing the solution is the real user of the technology or framework. The team, which consists of designers and programmers, expect to increase productivity and decrease correlation in work. Taking the team's point of view into account might put the hands on the causes of technology evolution.

*A.* The end user's concerns are:

*1) Usability:* according to ISO, "Usability is the effectiveness, efficiency and satisfaction with which a specified set of users can achieve a specified set of tasks in a particular environment", and is characterized by the followings:

*a) Productivity,* how effectively a user can perform his job using the system.

*b) Learnability,* how fast a user can learn how to use the user interface sufficiently to accomplish basic tasks.

*c) Error frequency,* how often does the user make errors while using the system and how serious these errors are.

*d) Memorability,* can a user, who has used the system before, remember how to use it effectively next time, or does the user have to learn everything from the beginning.

*e) Satisfication,* how much does the user like using the system.

*2) Application's availability,* the effeciancy and ability of the system to respond in a timely manner, and is characterized by the followings:

*a) Load time,* time needed to deliver the main of the application, script or client engine, by the server to the client. While much of this is usually automatically cached it needs to be transferred at least once.

*b) Responsiveness,* delay between the user's request and the received response.

*c) Network efficiency,* size of data needed to be exchanged with the server for each individual request or response.

*3) Portability,* an application is portable across a class of environments to the degree that the effort required to transport and adapt it to a new environment in the class is less than the effort of redevelopment.

*a) Platform Independence,* for example, the independency from extra system requirements like JVM, cookies and javascript.

*4) Security,* securing the access to the end user's local resources.

*5) Multi channel,* using the system from different channels such as mobile SMS, web service, etc…

*B.* The programmers' concerns are:

*1)* Decreasing the time needed to adopt a new technology which depends on the followings:

*a) Maturity,* technology maturity, support and level of community activity.

*b) Scalability/Extensibility,* often desired when an application must be able to support new features, such as networking protocols or file formats, that do not yet exist. This requires the application to supply a framework for the general problem without concern for the specifics of details.

*c) Flexibility,* the ability of software to change easily in response to different user and system requirements.

*d) Abstraction,* characterized by the archeticture and design patterns used.

*2)* Decreasing correlation with the designer by separating between the two aspects: content, business in the page, and view, final format of the page.

*3)* Increasing productivity by separating between the content and workflow such as view navigation rules, which allow the involvement of designers in early stages of application development process.

*4)* Application's performance are characterized by the followings:

*a) Build time at the server,* time needed to start executing the required action on the server.

*b) Server hit's rate,* the ratio between end user's clicks and the server's requests.

*5)* Application's reliability, the duration or probability of failure-free performance under stated conditions, and is characterized by the followings:

*a) Session management,* which is the process of keeping track of a user's activity across sessions of interaction with the computer system, for best utilizations of memory allocations, due to the distributed nature of the application.

*b) Reports management,* techniques used when generating large amount of data.

*6)* Reporting capabilities like the support of charts (pie chart, histogram, bar chart) and read only formats (Excel, pdf, rtf, …).

*7)* Support of multi-channels like wap/wml.

*8)* Security, presentation security.

*C.* The designers' concerns are:

*1)* Decreasing the need to modify the views each time the workflow is changed and that is by separating between the view and workflow, which is always in change.

*2)* Clear separation between view and content.

*3)* Dynamic look & feel, the ability of the application to change its look and feel, specific color scheme for example, without changes required to the application code.

## IV. MATRIX

The previously mentioned criteria are applied to four different web presentation frameworks:

*Apache Struts*, is an open-source web application framework for developing Java EE web applications. It uses and extends the Java Servlet API to encourage developers to adopt a model-view-controller (MVC) architecture. It was originally created by Craig McClanahan and donated to the Apache Foundation in May, 2000. Formerly located under the Apache Jakarta Project and known as Jakarta Struts, it became a top level Apache project in 2005.

*JavaServer Faces (JSF)*, is a Java-based web application framework that simplifies the development of user interfaces for Java EE applications. Unlike other traditional request-driven MVC web frameworks, JSF uses a component-based approach. The state of UI components is saved when the client requests a new page and then is restored when the request is returned.

*AJAX (Asynchronous JavaScript and XML), or Ajax*, is a group of inter-related web development techniques used for creating interactive web applications. A primary characteristic is the increased responsiveness and interactivity of web pages achieved by exchanging small amounts of data with the server "behind the scenes" so that the entire web page does not have to be reloaded each time the user performs an action. This is intended to increase the web page's interactivity, speed, functionality, and usability.

*eliXir presentation tier*, is an MDA conversational workflow which aims at facilitating the construction of view tier in distributed web applications. eliXir framework uses an MDA approach and takes as input BPM diagrams and UML class diagrams. These meta data are used by the presentation tier to automatically generate the system GUIs in different formats (interactive html, readonly html, pdf, excel, xml and graphical charts for example) and manage the conversation with end user through a well defined technical workflow and the business workflow. At the same time, completely controlling the size of the end user's session. The key features of the eliXir presentation tier are:

*1) Increased prductivity achieved by the full separation between the aspects: content, view, workflow and look&feel.*

*2) High usability achieved by a well defined conversation model.*

The following table contains the result of the comparison, where the – means that this point is not clear enough to consider as features. references from [6] ~ [13] were used to fill the table.

| **criteria** | **Comparison Matrix** | | | |
|---|---|---|---|---|
| | ***Apache Struts*** | ***JSF*** | ***Ajax*** | ***eliXir*** |
| ***end user*** | | | | |
| productivity | - | - | - | two main pages: home page for starting a process, and tasks page to execute pending manual tasks |
| learnability | - | - | - | two main page designs: page for creating, editing and viewing one object, and a page for searching for objects |
| error frequency | - | "Core" library which aids in common application development tasks such as validating / converting input data | - | client and server side validation and business integrity |
| memorability | - | - | - | two main page designs |
| satisfaction | - | from simple to complex UI components | desktoplike interface | dynamic look&feel |
| load time | - | - | ajax engine | one javascript file |
| responsiveness | synchronous request and response | synchronous request and response | asynchronous data retrieval using XMLHttpRequest | synchronous request and response |

| | | | | |
|---|---|---|---|---|
| network effeciency | - | - | data interchange and manipulation using XML and XSLT | optimized html page size |
| platform independance | - | - | javascript | javascript and cookies, jvm to generate the graphical charts |
| security | - | - | - | certified applet for charts reports |
| multi-channel support | only html | pluggable rendering capability | - | several output devices like: printer, fax and email are supported by eliXir |
| ***programmer*** | | | | |
| maturity | has the edge | you can rely on different levels of support depending on which implementation you choose | depends on which implementation you choose | - |
| flexibility | - | has 6 objects that implement much of the framework's capabilities and you can easily replace those objects by decorating the default implementations | - | pluggable reports engine which allow the customization of the report business and/or format |
| scalability/extensibility | generate html directly, there're no components | components based, renderers are pluggable | - | pluggable layouts and writers |
| abstraction | front controller pattern, command pattern.<br>actions are tied to the Struts api | page controller pattern, POJO action methods | - | front controller pattern, Layout and Writer patterns.<br>Objects are tied to eliXir api |
| presentation security | - | - | - | security through time: check whether the current user has the rights to execute the requested business action NOW |
| correlation | - | allowing the developer to construct web user interfaces using pre-built ui components | - | business content of the page is derived from the model, so the programmers can implement the business without the interfering with the designers |
| productivity | action definition in XML files, developers can then programmatically choose which forward to return | static navigation rules | - | business content is derived from the manul tasks contained within the business process which represents the navigation rules.<br>designers can start design the manual tasks while the bp is developed |
| build time | - | - | - | data is fetched from the request, business is executed on the server, response page is build and sent back to the client |
| server hit's rate | user's clicks are server requests | user's clicks are server requests | depends on which implementation you choose | behavioural and business actions are requests to the server |
| session management | - | - | - | the size of the user's session is preserved according to the user's |

| | | | | |
|---|---|---|---|---|
| | | | | actions |
| reports management | - | - | - | reports are generated using an incremental technique |
| readonly formats | only html | pluggable rendering capability | - | Excel, pdf, xml, rtf and graphical charts using JFree charts |
| multi-channel support | only html | pluggable rendering capability | - | pluggable layouts and writers |
| ***designer*** | | | | |
| view and workflow sep. | - | dynamic navigation rules contained within the view | - | workflow is dynamically derived from the model, so the designed page doesn't need changes when the business workflow changes |
| view and content sep. | - | - | - | business content of the page is derived from the model, so the designers can start working on the design without the interfering with the programmers |
| dynamic look&feel. | - | - | - | permit the customization of a stylesheet file |

## V. CONCLOSION

There are lot of presentation frameworks for web based and rich client applications. Current presentation frameworks focus on increasing the richness of UI components, increasing the responsive time, and trying to keep the architecture scalable and flexible. Each has their own advantages and disadvantages. It is also possible to mix frameworks to exploit the best of each one, and JSF + Ajax is a good example for this.

The paper outlined factors which are not addressed by current presentation frameworks, and which tend to increase usability, productivity and reliability. Considering eliXir presentation tier, this paper suggests building an MDA framework on top of one of the current presentation frameworks.

## REFERENCES


[1] Frank Nimphius and Duncan Mills. Swing or JavaServer Faces: Which to Choose?, Published March 2006, ORACLE – TECHNOLOGY NETWORK.

[2] Malcolm Davis, 01 Feb 2001. Struts, an open-source MVC implementation. IBM.

[3] Jesse James Garrett, February 18, 2005. Ajax: A New Approach to Web Applications. Adaptive Path.

[4] Wikipedia, the free encyclopedia. Rich Internet application.

[5] 2004-2005. Introducing JavaServer Faces Technology. Sun Microsystems, Inc

[6] Joseph Ottinger, April 2007. Rethinking JSF-The real Problem. TheServerSide.COM

[7] David Geary, 8/16/2005. Top 10 Reasons to Prefer JSF over Struts.

[8] Ginige, A and Murugesan, S. 2001' The Essence of Web Engineering: Managing the diversity and complexity of Web Application Development, IEEE.

[9] An Activity Theory Approach to Understanding the Difficulty of Applying Web Engineering In Practice. Amanda Quek and Albert Alderson.

[10] Comparison of Frameworks, R. Arunachalam Program Manager, Cerulean

[11] Technology options for Rich Internet Applications, Mr. Vaibhav V. Gadge, IBM, 25 Jul, 2006

[12] Comparing Java Web Frameworks, Matt Raible, Nov 26, 2007

[13] Web frameworks, Steven Haines, informIT, Aug 18, 2006