computing@computingonline.net
www.computingonline.net

ISSN 1727-6209
International Journal of Computing

METHODS OF CREATING RICH USER INTERFACES BASED ON FLEX FRAMEWORK

Bartosz Sakowicz, Piotr Mazur, Marek Kamiński, Michał Wojtera, Adam Dębiński

Technical University of Lodz, Poland,
al. Politechniki 11, 90-924 Łodź, Poland,
e-mail: sakowicz@dmcs.pl
http://www.dmcs.pl

Abstract: *The article, by describing features of sample application for project management, presents methods of creating graphical interfaces of modern applications that run in web browser environment. Sample program is written with the use of FLEX Framework and some FLEX supporting tools. The FLEX technology is described as well as other RIA techniques.*

Keywords: *FLEX Framework, RIA, GUI, Red5.*

1. INTRODUCTION

World Wide Web since its beginning in 1990 has come a long way marked with constant evolution. Since 1993, when the first widely known, graphical web browser – *Mosaic* – was created [3], web graphics and web GUIs have been the subject of the same evolution [6]. From the simplest, low-resolution 2D images, through Flash based vector graphics, to advanced 3D components and transformations – resources that web designers have at their disposal in conjunction with high-speed Internet connections and high-power computers of the final users gives that slowly the boundaries of imagination becomes the only limitation.

2. RICH INTERNET APPLICATIONS (RIAs)

RIAs join best functionalities of desktop applications and the Internet. (Fig. 1) [11]

The first ones offer rich and intuitive graphical interfaces with great capabilities, which make using them easier and prevent unwanted frustrations. Internet, from the other hand, provides perfect platform for created applications. Thanks to its global nature it allows everyone with active Internet connection to access available RIAs.

RIAs broke the boundaries encountered by the designers of standard HTML-based pages, while trying to give their applications desired design or level of interaction. Considering most significant features of RIAs, they seem to be almost perfect solution. They guarantee:

- stability, scalability and interactivity,
- rich interaction possibilities,
- ease of managing and maintaining,
- expanded communication functionalities.

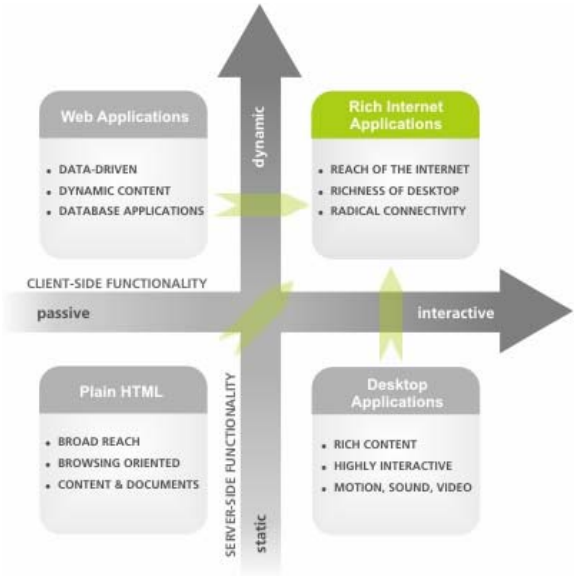


Fig. 1 – Diagram illustrating how RIAs join the best features of desktop applications, Internet and communication technologies [1]

Interactivity and scalability of the application is usually achieved by means of using existing technologies to update only part of the application interface when needed. Standard web applications require the whole user interface to be refreshed when an user performs an action which requires

server interaction. This can introduce significant delays and have a negative impact on user experience.

On the other hand RIAs use specialized techniques to refresh only a portion of the user interface when the need arises. It is usually achieved by using AJAX[1] to communicate with the Web Server and updating the user interface by means of HTML DOM[2] manipulation[9]. AJAX allowed to achieve greater user interface responsibility by issuing only small requests to the Web Server and transforming the result on the client side. Usage of AJAX in a Web Application can therefore have a positive impact on the web server performance by eliminating the need to generate the whole HTML page, when only a small portion of the interface was updated.

By joining all of before mentioned advantages, RIAs appear to be fascinating and impressive, new medium, which by providing exciting and interactive, graphical interfaces are perfectly suited for creating B2B[3] and B2C[4] applications and services [7].

It is also important to mention that although RIAs tend to provide similar user experience as standalone desktop applications there are a few capabilities that are still missing or not have an easy implementation. Network communication with other network services which are not compatible with HTTP protocol are not easily implemented limiting the user interaction to HTTP servers only. In addition some aspect of the HTTP protocol, which was designed as s simple request-response protocol can lead to difficulties with user interface interactivity. A HTTP server cannot maintain the network connection with the client for longer periods of time eliminating the possibility for the server to notify the client of a certain event. Although there are ways to lengthen the duration of the HTTP connection they usually lead to unpredicted behaviors especially when dealing with servers which limit the number of simultaneous connections.

Development of an RIA application with dynamic HTML, AJAX and DOM manipulation can also prove to be a difficult task. In recent years a huge effort was made to facilitate developing such applications by means of using JavaScript language libraries such as Prototype, jQuery or MooTools[5]. These libraries can greatly reduce the time required to develop an Rich Internet Application[10] but still require a significant amount of time to be spent on implementing the client-server connectivity and user interface modification.

Mentioned caveats indicate that while RIAs in conjunction with dynamic HTML and AJAX provide an important step forward in designing interactive web applications they do not address all the concerns, and some aspects of achieving desktop-like functionality in web applications are still missing or are difficult to achieve. Introduction of the Flex Application Framework allowed for these issues to be properly addressed allowing to greatly improve the time required to design an Rich Internet Application.

3. FLEX FRAMEWORK

Flex was published by Macromedia in March 2004 as an expensive server product designed for creation and deployment of applications. Since version 2.0 however, its nature changed drastically, making Flex a client-side technology which, additionally, was much more affordable. Introducing version 3.0 Adobe[6] made Flex SDK[7] open-source and free, charging only for the charts library and development environment – *Flex Builder*.

Flex Framework is the combination of at least a few technologies. It is not a standalone application but rather it consists of following elements [2]:

- **Languages:** MXML and ActionScript 3 (for creating Flex applications a combination of two languages is used. MXML is a markup language similar to HTML, which main purpose is to describe layout of the application. ActionScript 3 is an object, scripting language used to implement business logic and interaction events of the components of the application). Separating the layout of an application from its business logic allows for easy maintenance of the application code. Additionally it allows for one developer to be responsible for the graphical interface layout, while others deal with the business logic.
- **Components library:** Flex SDK (it's the set of graphical user interface components like buttons, lists or containers used for building Flex applications. Beside charts library, all SDK is free and open-source).

[1] AJAX – Asynchronous JavaScript and XML – technique allowing to send a request to the Web application Server and dynamically modify the user interface using the Server response.

[2] HTML DOM – HTML Document Object Model – a model representing the HTML document as a hierarchical structure which can be modified in real-time.

[3] B2B – *business to business*

[4] B2C – *business to consumer*

[5] Prototype, jQuery, MooTools – JavaScript libraries designed to facilitate AJAX and DOM manipulation in a web application

[6] In the meantime Macromedia was bought by creators of Acrobat Reader

[7] *Software Development Kit*

- **Integrated Development Environment:** Flex Builder (it's the environment created on the base of *Eclipse IDE* used for code editing, application compiling, debugging and efficiency testing. Unfortunately it's not a cost free product. Adobe sells it in two versions: costing $249 Flex Builder Standard and costing $699 Flex Builder Pro, which offers additionally tools for memory and efficiency management and charts library [5]).
- **Runtime environment for multiple web browsers:** Flash Player (created applications are run in the web browser environment with the help of Flash Player plug-in. There's also a possibility of running Flex applications in the standalone mode using AIR[8] Environment, provided free of charge by Adobe).

Usage of the aforementioned elements enable developers to create easy to use Rich Internet Application in a much shorter time span when comparing to the standard HTML based web applications. By using a standard Flash Player application users can experience a rich GUI (Graphical User Interface) while using only a browser and an appropriate plug-in.

Usage of the component library enables developers to design application interface in a similar way it is done in standalone desktop application using an WYSIWYG[9] interface. The richness of the Flex component library can also be greatly augmented by extending existing graphical components and improving them and adding new behaviors. Creation of an extended version of a graphical component can be as easy as writing a new ActionScript class which extends a well known graphical interface class with additional methods that alter the component's behavior when a need to repaint the component has arisen.

In addition to the graphical interface components application developers can leverage other technologies available to Flex which can greatly improve the responsiveness of an application. A rich network communication library was introduced allowing Flex application to easily integrate with not only web application servers but also other services which require a different protocol for communication. Flex application can still use network communication to interact with various web servers by means of Web Services or simply by downloading appropriate content from a HTML page. Network communication also introduces the possibility for Flex applications to communicate with each other using a simple interface.

Using a media server such as Adobe Flash Media Server[10] or Red 5 Media Server[11] it is also possible to stream audio and video content to a Flex Application. Streams can also be published from within a Flex application, which in combination with access to a camera and microphone can allow for creation of live video streaming applications in a relatively simple matter. It is also possible to stream static video content directly from a streaming server or a simple HTTP server. Unfortunately not all existing video and audio codecs are currently supported and the choice is often limited to proprietary codecs only.

Additionally the presence of a media server can also provide the possibilities not easily available for standard Web Applications. The connection between the application and the media server can be maintained indefinitely allowing the server to initiate an action on the client when a need arises. This is not easily available in dynamic Web Applications where the connection is severed as soon as the content was received, and usually some kind of pooling mechanism has to be employed to periodically probe the server for new data.

Flex technology also introduces the SharedObject class which represents an arbitrary object that can reside on the server side. An update on the shared object performed by any of the Flex client application is propagated to every other flex application that have subscribed for this object. This allows for the data to be transparently passed between many Flex applications without introduction of an additional delay and with a minimal input required from the application developer. Although similar technologies emerged for use in traditional web applications (for example: COMET[12]) there are not easy to use and burden the developer with additional application code.

When introducing network communication to the existing application connection security becomes a great concern. Flex uses an proprietary RTMP protocol to communicate with media servers with an additional ability to secure all maintained connections by means of SSL encryption. This allows for all media or SharedObject data to be

[8] *Adobe Integrated Runtime*

[9] WYSIWYG – (What You See Is What You Get) – an popular term used for designing an user interface by simply placing the GUI components in some kind of application

[10] Flash Media Server – a server component offered by Adobe allowing for streaming video content in a publish-subscribe fashion

[11] Red 5 Media Server – an Open Source media server which implements some aspects of the Flash Media Server.

[12] COMET – a technique which allows to maintain a connection to the Web Server for an indefinite period of time. This way a server can notify the client application when en event on the server has occurred.

passed between client and server in an encrypted matter.

SharedObject class can also be used to store user data locally on the machine running the Flex application. In a similar fashion as cookies[13] are used in standard web applications the SharedObject class can be used to serialize whole objects and store them locally on the disk.

4. "PROJECT DESIGNER" APPLICATION DESCRIPTION

Project Designer is an application created with Flex Framework, which main purpose is to improve projects planning, creating and maintaining processes [8]. Program tries to achieve this goal by moving most of the functionality that person responsible for project management could possibly need – to graphical environment. Almost every single action can be done with a few mouse clicks and moves. At the same time Project Designer is the tool advanced enough that it could successfully find its place in companies, considering project management an ordeal, because of the complexity of solutions included in competitive applications.

Project Designer consists of three key views responsible respectively for project management, project's stage management and resource management. Switching between them takes place in strictly defined points of application, however they're intuitive and don't present any orientation difficulties for the user. What's more, most of the functionalities of each view is visible directly after loading the view and those that are hidden, can be found in easily predictable places, what further improve user experience.

5. PROJECT MANAGEMENT VIEW

Fig. 2 presents application right after first launch. It can be easily seen that it consists of three main parts: starting from the top there is a time ribbon, workspace and a simple toolbar.

Time ribbon consists of three sets of buttons positioned in three rows. Every row represents successive date field starting from years (top row), through months (middle row), to days (bottom row). After application launches, only years row is visible, which consists of three buttons representing current year and two following ones. When user presses any year, then button representing it takes all available width of the years row and months row shows up. Pressing any of the month buttons causes resizing of pressed button and showing up of the days row. Pressing any button above the lowest row causes all the rows bellow pressed button to hide and the row with button to return to initial state.

Algorithm of creation of successive rows works as follows. Every row and every button placed in it is described by time range attribute (start and end date). Algorithm launches every time this attribute is changed for a row. First step is to create an array of labels that will be displayed on buttons. The mechanism of ActionScript language is used here that allows access to field of any class by providing the name of the field in the form of String literal (Fig. 3).

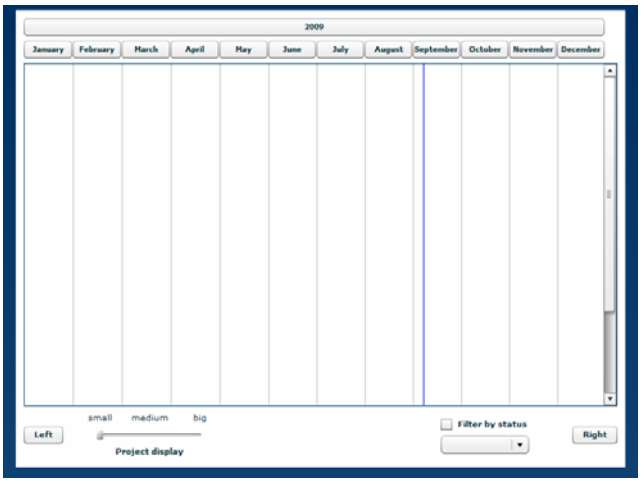


Fig. 2 – Project management view

```
var rok = dateRange.startDate.fullYear;
var rok = dateRange.startDate["fullYear"];
```

Fig. 3 – Example of accessing field of a class by providing name of the field as a String literal [4]

Thanks to this mechanism it was possible to write one all-purpose method, getting field name from the nest level of the time ribbon. Labels are obtained by iterating through all values from start date to end date for appropriate date field and changing numeric values to text in the case of months. Next, buttons are created in the amount equal to the amount of labels. Width of each of them is measured as row width divided by number of buttons. In the next step, for every created button, based on the date field that it's representing, date range field is set.

Workspace, located in the middle of the application is the main area of user's work. It's here where project management is mainly happening. Workspace mirrors time ranges defined by buttons in time ribbon, what is stressed by vertical lines, positioned to mirror positions of spaces between buttons. Every change of the time ribbon structure is immediately reflected in generating of the workspace and every element that it contains. Blue vertical line represents current date.

[13] Cookies – a method allowing for storage of simple name-value pairs on the client side. Commonly used by Web Applications to store user configuration data.

Last element of the described view is toolbar. It consist of components that help decreasing chaos that can appear when to much projects are managed at the same time. First of them is slider for changing projects' height. Typically they are displayed with both name and time range visible. However, when user decides that this view is unreadable for him, he can switch size to one of the two smaller ones.

Next functionality allows filtering of visible projects by their current status, if user doesn't want to see projects that are ex. finished or still planned.

"Left" and "Right" buttons are used to scroll time ribbon one unit of actually active row to the left or right allowing more smooth work with application and improving user experience.

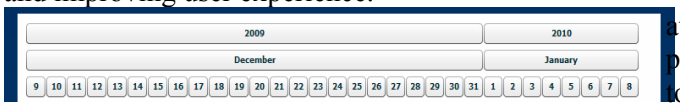


Fig. 4 – Time ribbon scrolled 8 days right from December 2009

Every graphical component that represents project consists of three main elements: icon, reflecting current status of the project, button, showing every change in the time range of the project and a label with project name. (Fig. 5).

Fig. 5 – Sample project component

To create a new project, all that is necessary is to press LMB[14] at some place in the workspace and move the mouse with LMB pressed until desired length of a project is reached. Time range changes are visible on the button in real time so acquiring desired time boundaries of the project is quite easy.

Editing of the project is equally trivial. User can change both start and end dates at the same time by dragging project component around the workspace or he can change only one date at a time by resizing component with handles located at the both sides of it. Every change in the "x" coordinate of the left or the right side of the component results in launching an algorithm estimating current dates.

The algorithm starts with searching for reference button. If the result of dividing ribbon row width by number of buttons in it was an integer the reference button is last button in the row. If it wasn't, the reference button is last button in the row that has width bigger than the rest of buttons. It guarantee that all buttons to the left and to the right of the reference one are of the same width. Next, "x" coordinate of the right side of the button is set as a reference point. After that, algorithm measures the distance between the reference point and the point, for which a date is being searched. The distance is divided by appropriate width of the button and the result represents the offset against the value of the date field visible on the reference button. Rest from this division is additional offset (in days) that allows to precisely estimate quested date.

Dates can be also edited more traditionally by pressing the button on the project component. This way a popup window with two calendars will be opened allowing to freely choose desired dates. The only restriction is that the start date must come before the end date.

Beside dates, user is allowed to modify other attributes as well. After double-clicking on the project component a pop-up with attributes editing tools can be accessed. It allows to modify project component color and transparency as well as project name and status. (Fig. 6)

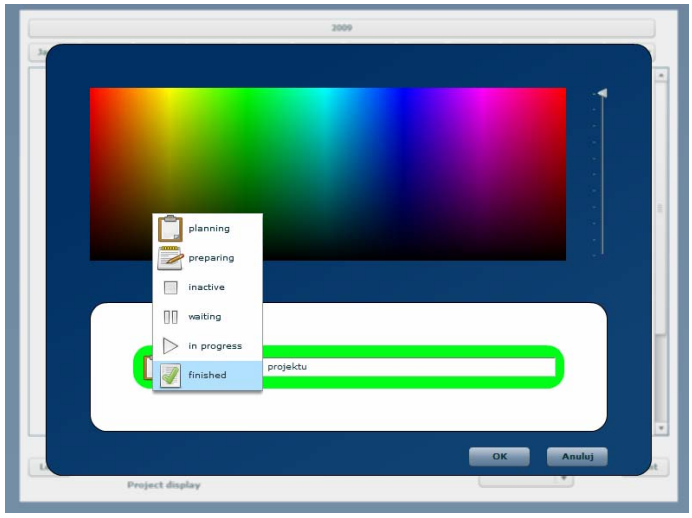


Fig. 6 – Pop-up for project attributes modifying

If user wants to start editing stages of the project, he can do it anytime by double-clicking on a project and choosing "Edit stages" from a menu.

6. STAGE MANAGEMENT VIEW

Fig. 7 presents stage management view right after switching to it. The only difference to the previous view is additional functionality implemented by the ribbon – time boundaries. With projects, user wasn't in any way limited in how long his project was or when it started. Stages from the other hand are not so liberal. Time ranges of the ribbon are set to those of currently edited project. Fig. 7 presents this kind of situation. Time ribbon is shorter because edited project time range is set to 15.05.2009 – 28.10.2009.

[14] *Left Mouse Button*

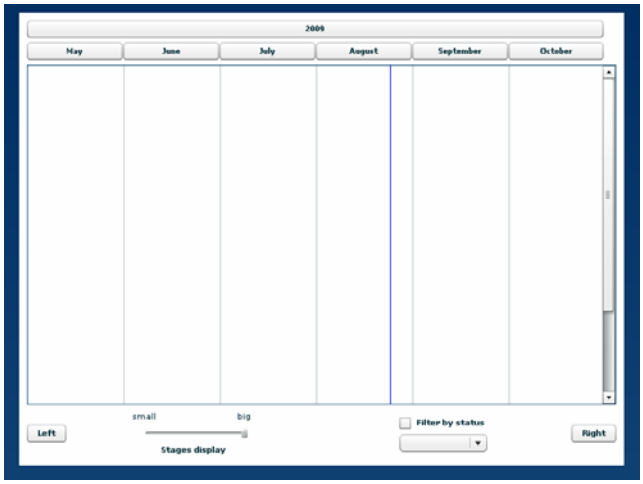


Fig. 7 – Stage management view

Stage component differs quite much from the project one. The biggest difference is introducing of component division. (Fig. 8)

Fig. 8 – Sample stage component

That way much greater flexibility was achieved. Each part has its resources defined separately what allows better time, workspace and resource management.

Most of the functionality related to component editing is implemented in the same way as in the project management view (moving the mouse while holding LMB for creating, dragging and resizing the component for editing), however in the stage view, user additionally gains the ability to change the date separating both parts of the component. Attributes are set with the help of appropriate pop-up as well. Stage component is also "equipped" with two triangles. Their function is to hold handles of the connections used for mirroring real-life dependencies between the stages and representing workflow of the project.

Creation of the connection is accomplished by placing mouse pointer above one of the triangles, moving the pointer above another triangle with LMB pressed and releasing LMB.

Stage management view also allows user to create additional *split*/*join* components. Using them gives the ability to control simultaneous starting and ending of multiple stages.

All before mentioned functionalities can be seen at Fig. 9.

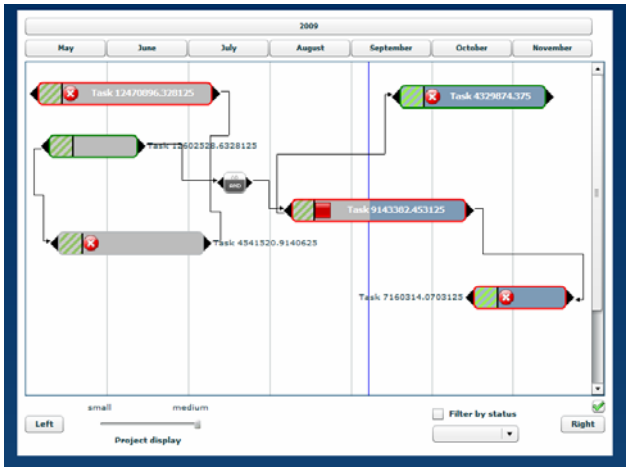


Fig. 9 – Sample arrangement of stages inside of the project

By double clicking the stage and choosing "Edit resources" option access to the resource management view can be gained.

7. RESOURCE MANAGEMENT VIEW

Resource management view is really a pop-up that allows few basic resource management actions. It consists of two graphical lists. Right one contains all resources available for currently logged on user, the left one – resources applied to edited stage. Fig. 10 shows the exact moment of moving the resource from one list to another. Icons that can be seen to the left of the resource item represent type of the resource (human, device). Little globe is the indication of global resource that is available for both parts of the stage.

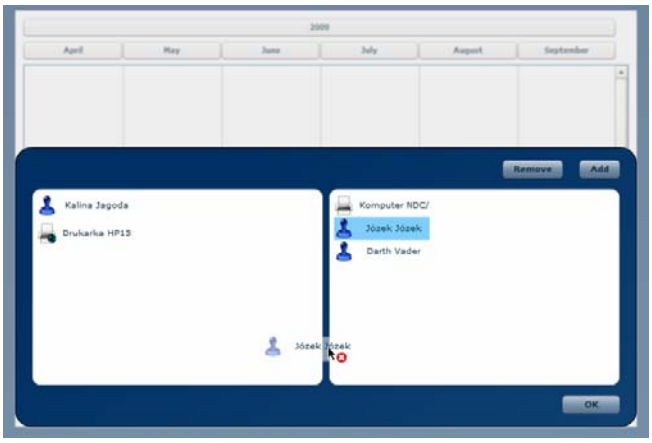


Fig. 10 – Resource management view in the moment of adding resource to the stage

8. CONCLUSIONS

Modern IT projects head for moving most of the "behind the scenes" calculations to final user's devices. By doing that costs of maintenance and support of such applications drop drastically. However to achieve this kind of functionality web

applications have to mirror desktop programs as well as it is possible, including presentation layer. Adobe with its Flex Framework, as well as the rest of RIA frameworks making competition, by providing advanced, mature tools for creating such applications, show that they're the ones that'll be shaping the future of the IT.

9. REFERENCES

[1] Outsmart. Outsmart, Rich Internet Applications: Extraordinary Interactive Experiences. [Online]. http://www.getoutsmart.com/rich-internet-applications.htm

[2] D. McCune and D. Subramaniam, *Adobe Flex 3.0 for Dummies*, K. Darosett, Ed. Indianapolis, USA: Wiley Publishing, Inc., 2008.

[3] G. Wolfe. (October 2004) Wired 2.10, *The (Second Phase of the) Revolution Has Begun*. [Online]. http://www.wired.com/wired/archive/2.10/mosaic.html

[4] Adobe Co. (2009) Adobe Flex 3.3 Language Reference. [Online]. http://livedocs.adobe.com/flex/3/langref/

[5] A. S. Incorporated. (Wrzesień 2009) Adobe Flex. [Online]. http://www.adobe.com/products/flex/

[6] B. Sakowicz, M. Wójtowski, P. Zalewski, A. Napieralski. Problems of standardization in web technologies, *XI Conference "Sieci i Systemy Informatyczne*. Łódź, October 2003, pp. 111-114, ISBN 83-88742-91-4.

[7] M. Ostruszka, B. Sakowicz, A. Napieralski. Universal web-based charts generator based on J2EE platform, *CADSM'2007*; ISBN 978-966-553-587-4.

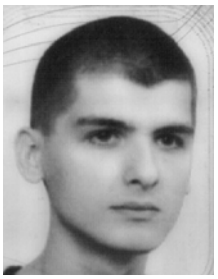
[8] M. Wojtera, B. Sakowicz. Web application for project management based on open source solutions, *MIXDES'2006*, ISBN 83-922632-9-1

[9] Den Odell, *Pro JavaScript RIA Techniques*, Apress Publishing 2009, ISBN-13: 978-1-4302-1934-7.

[10] Jonathan Chaffer, Karl Swedberg. *Learning jQuery 1.3*, Packt Publishing 2009, ISBN: 978-1-847196-70-5.

[11] Adam Debinski, Bartosz Sakowicz, Marek Kaminski. Methods of creating graphical interfaces of web application based on the example of FLEX framework, *TCSET'2010*.

Bartosz Sakowicz *received a Masters Degree and Ph. D. Degree in Computer Science from the Technical University of Lodz in Poland in 2001 and 2007 respectively. During Ph. D. studies he joined the Department of Microelectronics and Computer Science (DMCS TUL). He is now an Assistant Professor. His research activities include internet applications, distributed computing, data warehouses, game theory and stochastic optimization.*



Piotr Mazur *is a PhD candidate with the Department of Microelectronics and Computer Science on Lodz, Poland. His current research focuses on areas concerning relational database management systems, operating systems and voice transmission using VoIP technology.*

Marek Kaminski *received a Masters Degree and Ph. D. Degree in Electronic from the Technical University of Lodz in Poland in 2002 and 2006 respectively. During Ph. D. studies he joined the Department of Microelectronics and Computer Science (DMCS TUL). He is now an Assistant Professor. His research activities include internet applications, distributed computing, data warehouses and stochastic optimization.*

Michal Wojtera *received Masters Degree in Computer Science, in 2006, from Technical University of Lodz, Poland. Currently he is a PhD student with the Department of Microelectronics and Computer Science. His research is focused on neural networks, parallel computing, software engineering and internet applications.*

Adam Dębiński *received a Masters Degree in Computer Science from the Technical University of Lodz in Poland in 2009. His main field of work are RIAs and their integration with existing web applications.*