

Web applications development in collaborative frameworks

Sorin Lucian PAVEL

Academy of Economic Studies, Bucharest, Romania

pavelsorin@gmail.com

Abstract: *The paper presents the development requirements of web applications. It also settles functional structures and identifies typologies of suitable web application interfaces. The stages of web applications development in collaborative frameworks are presented along with particularization on matrix blocks operating application.*

Keywords: *web application, collaborative framework, matrix blocks*

1. Web Applications

In software engineering [1], a web application is an application destined to be accessed via a web browser over a certain network – Internet or intranet. A web application may also mean a computer software application that is coded in a language supported by the browser – HTML, JavaScript, Java – which render the application executable.

The convenience of using a web browser as client along with the ability to update and maintain web applications without distributing and installing software on potentially thousands of client computers are the main reasons for their popularity. Common web applications include e-mail, online retail sales, online auctions and many other functions. In addition, the high level of accessibility and usability along with a great deal of resources is another argument for employing web applications.

Because of their diversity, many classification criteria have been pointed out. Taking user interaction, Hagan Rivers [2], the principle designer for Netscape browser, describes three different structures of web applications:

- i. *Interview-based Applications* – illustrated in **Fig. 1** – that ask users for information in a linear manner, often using “continue” and “back” buttons. These applications are recommended for gathering large amounts of information at once. Progress meters in interview-based applications are critical, giving users a clear picture of where they are and how far they have to go. Interview-based apps are excellent for helping users make complex decisions.

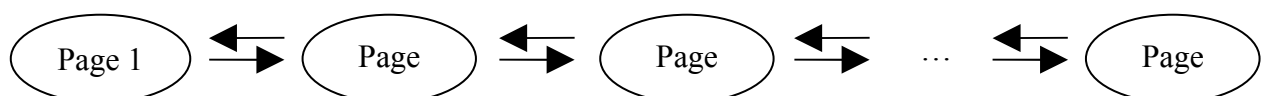


Fig. 1. Interview-based Applications

- ii. *Hub and Spoke Applications* – illustrated in **Fig. 2** – that show an overview of information (hub) and a bunch of options to choose from (spokes). The similarity comes from a wheel with many spokes. Hub application developed out of desktop

applications. The calendar application is a classic example. Users need to change the information on one of the dates and return to the main view when finished. Many hub applications can be combined to create more sophisticated applications.

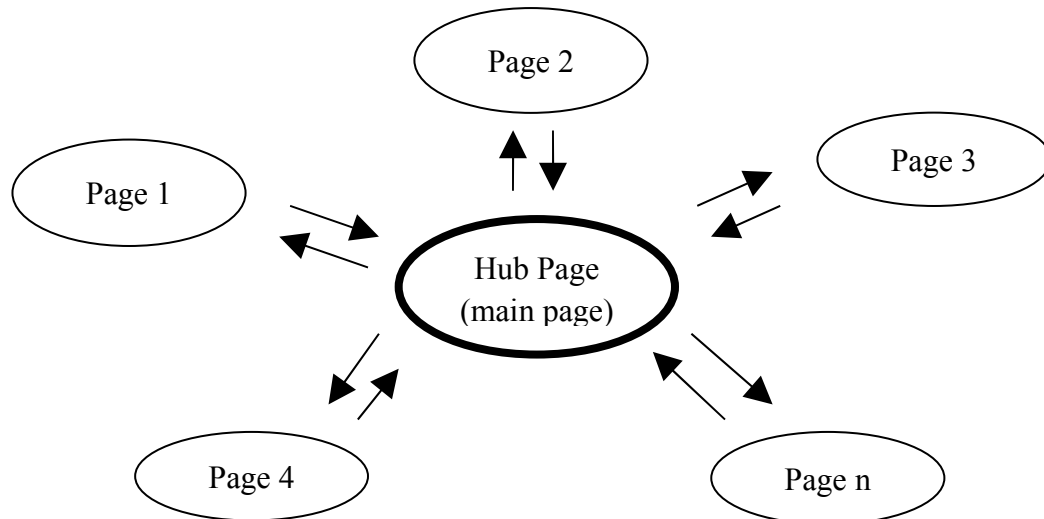


Fig. 2. Hub and Spoke Applications

- iii. *Hub/Interview Hybrid Applications* – illustrated in **Fig. 3** – that have characteristics of both hub and interview applications. Amazon checkout is the classic example. Entering addresses and credit card information are the interview part, while the review page and modifying pages are the hub/spoke part of it. These applications are the most difficult to get right, but one of the best when done correctly. For example, when users complete the interview section of one of the applications and go back to change a bit of information they previously entered, they can be taken back through the rest of the interview, or they can be taken directly back to the review page.

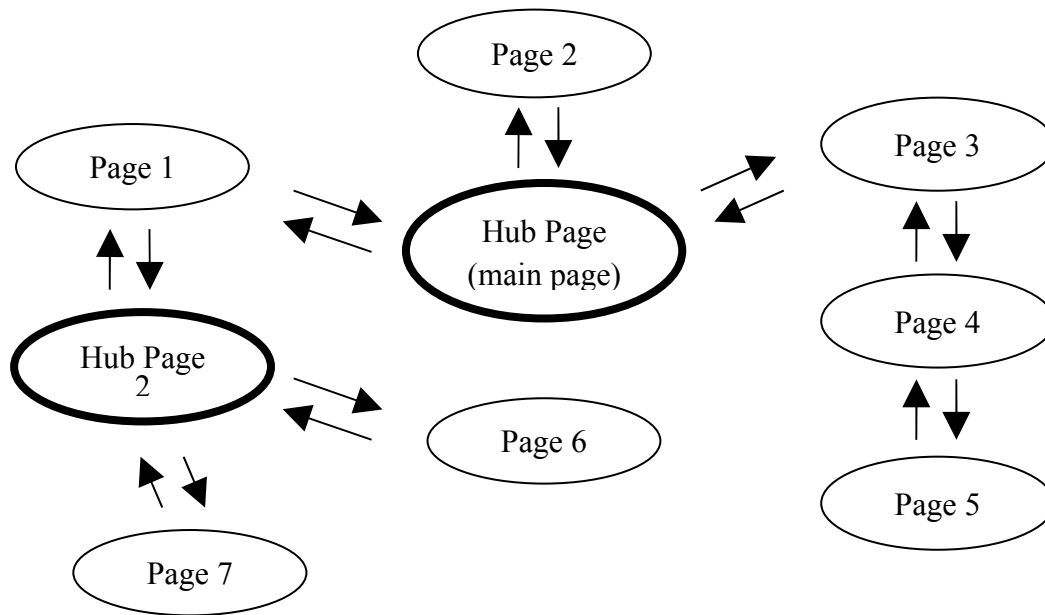


Fig. 3. Hub/Interview Hybrid Applications

All three approaches of web applications are based on dynamic shifts between pages and have their own way of managing the data transit.

2. Development cycle of web applications

Just as with a traditional desktop application, web applications have their own process and practice of development. The stages or steps of the cycle may vary, depending on:

- the software technology chosen for building the application [3] – ASP.NET, Java, ColdFusion, PHP, Perl, Ruby on Rails etc;
- the lifecycle model, including agile models like: Extreme Programming, Scrum, Timebox Development, Feature Driven Development etc;
- level of risk given by the amount and importance of data and data flows which affect the dimensions of certain development phases like quality planning or testing;
- software tools which support the chosen software technology: Visual Studio for ASP.Net, the CFML engines for ColdFusion, Notepad ++ for almost anything etc;
- adopted frameworks by providing libraries for database access, templating frameworks, session management, and often possibility of code reuse.

The general model for web application development includes [4] defining the target group, needs analysis, technical design, code development and production, integration, testing, quality assurance, production support and maintenance.

- *Defining the target group and the browser*

A primary consideration in planning is identifying who will use the Web Application or who will be the audience. The target group can be defined by identifying those who have access to the Web Application and the type of Web browser they use. Access can come through an intranet, the Internet, or an extranet. The capabilities of the Web browser shape

the plans for using client and server script in the Web application. Using server script and Active Server Pages (ASP) can generate browser-independent pages easily. In contrast, if it is known the type of browser the user will have, the application can use client script to generate the page and minimize the load on the server. The audience also determines what type of run-time security your Web application needs.

- *Needs Analysis*

This phase provides a solid understanding of what the business problem is, and what goals and objectives are set for commissioning a custom application. It consists in a number of meetings with the owner, the clients or the target audience, and the domain experts to thoroughly examine all aspects of the needs and preferences. The deliverable at this stage is a functional specification document that details all project requirements, the target audience and user environment, and any functional or organizational constraints to consider.

- *Technical Design*

A robust, effective, and reliable custom application should be designed from the beginning with consistent, well-documented technology and software engineering standards and practices. The goal is to produce high-quality code and database designs that are easy for both the current and future programmers to understand, build, test, and maintain throughout the life of the application. The deliverable at this stage is a technical specification document.

- *Code Development and Production*

This is where the software code and any required databases are actually created. The web application developers use both the needs analysis and technical specifications as guides while building the application. The deliverable at this stage is completed versions of the software code and databases, ready for testing and final quality assurance trials.

- *Integration, Testing and Quality Assurance*

Integration and testing of the application provides opportunity to confirm that the application meets the functional requirements and the needs of all users, as outlined in the needs analysis and functional specifications. The testing process is crucial giving the diverse threats which exist over the internet.

- *Production Support and Maintenance*

The final step consists in providing a comprehensive set of support, database and system maintenance services, and working with clients to identify and add minor features or user interface enhancements or to correct any unforeseen problems that might arise. This might mean supporting the application over the complete life cycle, from initial launch to re-design or decommissioning, or arranging a transition to newer code technologies or hardware support.

The development process of web applications may critically be affected when a collaborative framework is present. The lifecycle needs to be adapted to the unique specifications of the collaborative development.

3. The collaborative framework

A collaborative system describes a situation where multiple users or agents are engaged in a shared activity, usually from remote locations, while working together towards a common goal and have important interaction with each other [5]. As IT communication is continuously developing, spontaneous collaborative systems are forming throughout all fields of activity. The uniqueness of the collaborative systems is given by their capacity to optimize and reduce time for certain complex activities by assigning different tasks to different agents and resolving them simultaneously.

The structure and the constituent parts of the systems may vary leading to different types of collaborative assemblies. Giving the level of complexity, are identified [6]:

- low-complexity collaborative systems, which have few components and/or number of relationships;
- medium-complexity collaborative systems, with many components, but do not have large number of flows or relationships;
- high-complexity collaborative, which have many components and many flows.

The specificity of the common goal identifies:

- collaborative systems in economics;
- collaborative systems in IT, etc.

Given the method of organization within the system, there are:

- linear systems, where components interact and communicate from the same level;
- tree systems, with levels of importance or responsibilities and restrictions of communication;
- network systems, with levels but no restriction of communication.

Other criteria may sight the nature of components or the lifetime of the systems.

The software development process suffers several modifications when it unfolds in a collaborative framework [7]. Every phase with its own activities and tasks needs to be divided and assigned to different groups for independent work. This could lead to confusing and misleading actions or results if not properly managed. For example, in the needs analysis step, each component may gather information about certain required functionalities for a given area or module. But the way this area affects other modules is unclear and needs further details. There are situations when the information gathered within the collaborative system is not matching. Different components described differently the same functionality. As the development evolves, the process unifies and the fuzziness is resolved. The last stages – application testing or maintenance – is more effectively completed in collaborative systems: each component deals with specific functionality and assures its validity and proper run.

Collaborative systems offer the same stimulating framework for web applications as well. The communication is easier as the application is built on “neutral ground” and all system components have access to the same version. The development process is similar to the open source communities, but collaborative systems imply more structured and effective organization.

4. Matrix blocks computation

High-technology development in current society, citizen oriented software proliferation, along with new IT laws issuance, lead to production of software that works with very large scale datasets:

- telecommunication operators records data about each call or message sent or received through the network, and keeps the information for a six-months period;
- internet and e-mail providers record data for each IP address in their own administration, about visited web pages, along with exact time of access, and also about every electronic message sent;
- public administration keeps payment history for millions of citizens;
- national providers of gas and electricity operate millions of bills annually;
- on-line search engines integrate content management of billions of websites.

These cases presume lots of users that wield very large numbers of data – $10^7 \sim 10^{10}$ datasets – along with software applications that structure, control and operate the data.

An example of large datasets usage is the large matrix computation web application. By “large matrix” is referred matrix with hundreds of lines and/or columns. The recommended way of dealing with this kind of dimensions, is splitting the matrix in smaller matrices or matrix blocks. The content of the matrix is described by real or integer numbers, while the matrix type can be: unity matrix, rare matrix, triangle matrix, symmetric matrix, diagonal matrix etc.

The web application for large matrix computation (WALMC) in collaborative systems must allow definition and manipulation of matrix for algebraic calculus:

- defining matrix dimensions;
- entering values for each matrix element;
- uploading existent matrices from files;
- visualizing existing matrices;
- updating/modifying matrices;
- uni-matrix calculus: inverse matrix, pseudoinverse matrix;
- multi-matrix calculus: matrix addition, subtraction, multiplication;
- visualizing results.

The user access to the WALMC is granted by choice:

- free access with declarative and functional restrictions;
- registered access with no functional or physical restrictions.

The WALMC architecture implements a Hub/Interview Hybrid approach. The main/default page is the starting point where the user choose whether to register or not, and starts using the application. The logic structure is described in **Fig. 4**.

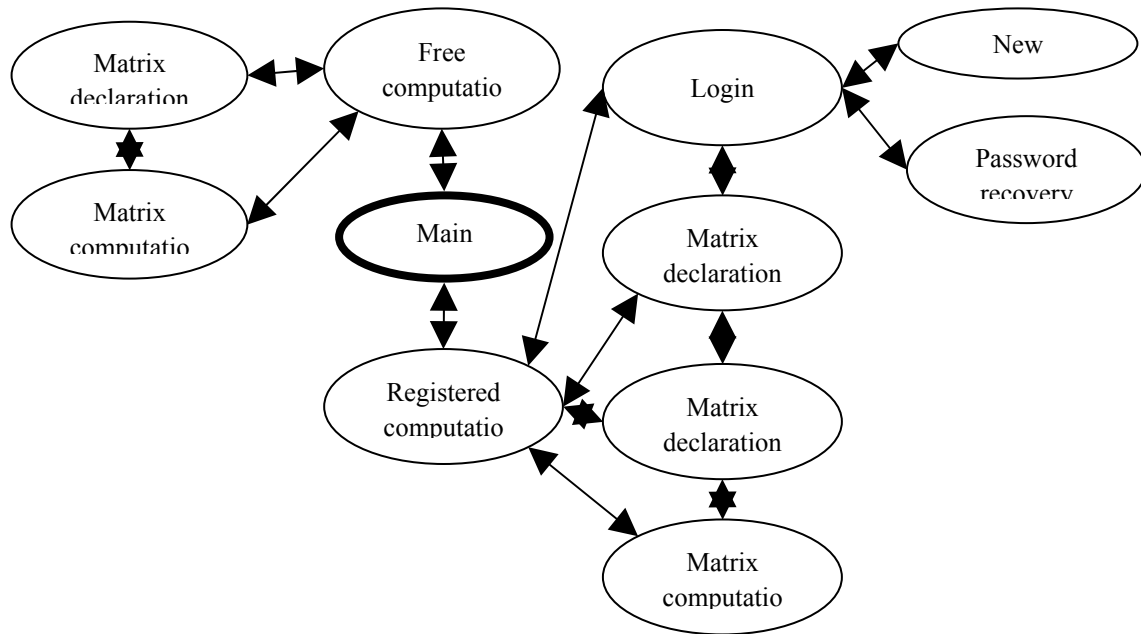


Fig. 4. The Hub/Interview structure of WALMC

The user interface of WALMC includes pages for guidance and direction, pages for registering/log-in and pages for matrix definition and computation.

The collaborative way of computing matrices is based on splitting large matrices in smaller blocks. Let A be a matrix having m lines and n columns:

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}$$

then A is decomposed in containing blocks $A_{11}, A_{12}, A_{21}, A_{22}$ as:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

where:

$$A_{11} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1k} \\ a_{21} & a_{22} & \cdots & a_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ a_{h1} & a_{h2} & \cdots & a_{hk} \end{bmatrix}$$

$$A_{12} = \begin{bmatrix} a_{1k+1} & a_{1k+2} & \cdots & a_{1n} \\ a_{2k+1} & a_{2k+2} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{hk+1} & a_{hk+2} & \cdots & a_{hn} \end{bmatrix}$$

$$A_{21} = \begin{bmatrix} a_{h+11} & a_{h+12} & \cdots & a_{h+1k} \\ a_{h+21} & a_{h+22} & \cdots & a_{h+2k} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mk} \end{bmatrix}$$

$$A_{22} = \begin{bmatrix} a_{h+1k+1} & a_{h+1k+2} & \cdots & a_{h+1n} \\ a_{h+2k+1} & a_{h+2k+2} & \cdots & a_{h+2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{mk+1} & a_{mk+2} & \cdots & a_{mn} \end{bmatrix}$$

and h, k are specified given the following conditions:

$$1 \leq h < m$$

$$1 \leq k < n$$

Given the collaborative framework, the initial matrix A is read and stored in different system components that might be geographically or logically sparse. The advantages of collaborative systems are outlined in computing matrix addition, subtraction or multiplication.

For large matrix addition, let A, B be in their decomposed form:

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}$$

$$B = \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix}$$

The addition can take place only if the dimensions of A_{11} and B_{11} , A_{12} and B_{12} , A_{21} and B_{21} , A_{22} and B_{22} are equal in pairs. When computing the matrix addition, the different components will operate with different blocks of matrix, so the processes are able to take place simultaneously giving the following relations:

$$C_{11} = A_{11} + B_{11}$$

$$C_{12} = A_{12} + B_{12}$$

$$C_{21} = A_{21} + B_{21}$$

$$C_{22} = A_{22} + B_{22}$$

For matrix subtraction, the same relations are used, but firstly the opposite blocks are created:

$$C_{11} = A_{11} + (-B_{11})$$

$$C_{12} = A_{12} + (-B_{12})$$

$$C_{21} = A_{21} + (-B_{21})$$

$$C_{22} = A_{22} + (-B_{22})$$

In case of matrix multiplication $C = A * B$, the restriction specifies that the number of lines in the B matrix needs to be equal with the number of columns in A matrix. The result:

$$AB = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix} \begin{bmatrix} B_{11} & B_{12} \\ B_{21} & B_{22} \end{bmatrix} = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

is obtained by collaborative computing:

$$C_{11} = A_{11}B_{11} + A_{12}B_{21}$$

$$C_{12} = A_{11}B_{12} + A_{12}B_{22}$$

$$C_{21} = A_{21}B_{11} + A_{22}B_{21}$$

$$C_{22} = A_{21}B_{12} + A_{22}B_{22}$$

In case of inverse matrix computing, the matrix blocks need to be square. The inverse matrix blocks are computed simultaneously. If

$$A^{-1} = X = \begin{bmatrix} X_{11} & X_{12} \\ X_{21} & X_{22} \end{bmatrix}$$

then:

$$X_{11} = (A_{11} - A_{12}A_{22}^{-1}A_{21})^{-1}$$

$$X_{12} = -(A_{11} - A_{12}A_{22}^{-1}A_{21})^{-1}A_{12}A_{22}^{-1} = -X_{11}A_{12}A_{22}^{-1}$$

$$X_{21} = -A_{22}^{-1}A_{21}(A_{11} - A_{12}A_{22}^{-1}A_{21})^{-1} = -A_{22}^{-1}A_{21}X_{11}$$

$$X_{22} = (A_{22} - A_{21}A_{11}^{-1}A_{12})^{-1}$$

The matrix computation time is in each case dramatically reduced when using a collaborative environment, the total saved time rising up to 70%, but more study is needed to be completed in this area.

5. Conclusions

Web Applications are more and more preferred to traditional desktop applications because of their increased accessibility and effectiveness. The web application development cycle includes defining the target group, needs analysis, technical design, code development and production, integration, testing, quality assurance, production support and maintenance. All of these steps are affected when working in collaborative frameworks. The matrix computation web application implements collaborative operation by dividing large matrices in smaller blocks and manipulating them simultaneously. Further studies need to be completed to show how much time is saved in collaborative systems.

References

- [1] Fowler Susan, Stanwick Victor “Web Application Design Handbook: Best Practices for Web-Based Software”, Morgan Kaufmann, 2004 ISBN: 1558607528
- [2] Hagan Rivers, “Deconstructing Web Applications”, *User Interface 10 Conference*, 10-13 October 2005, MA, USA
- [3] Marc Wandschneider, “Core Web Application Development with PHP and MySQL”, Prentice Hall, Sep. 2005
- [4] Wei Huang, Ru Li, Maple C., Hongji Yang, Foskett D., Cleaver V., “Web Application Development Lifecycle for Small Medium-Sized Enterprises (SMEs)”, *The Eighth International Conference on Quality Software*, 12-13 Aug. 2008

- [5] O. Dobrican, "An Example of Collaborative System", *International Workshop Collaborative Support Systems in Business and Education*, Risoprint, Cluj-Napoca, October 2005, pp. 48.
- [6] I. Ivan, C. Boja, and C. Ciurea, "Collaborative systems metrics", ASE Publishing House, Bucharest, 2007.
- [7] I. Ivan, C. Ciurea, "Collaborative Systems Assessment", *Workshop ISOM - Information Systems and Operation Management*, March 2007, pp. 310-324.

Author



Sorin Lucian PAVEL has graduated the Faculty of Economic Cybernetics, Statistics and Informatics from the Bucharest Academy of Economic Studies in 2008. He is currently following Master's in Software Project Management and the Doctoral School in Economic Informatics, both at the Academy of Economic Studies.