COLLABORATIVE SYSTEMS TESTING[1]

Paul POCATILU[2]
PhD, Department of Computer Science in Economics,
University of Economics, Bucharest, Romania

E-mail: ppaul@ase.ro

Cristian CIUREA[3]
PhD Candidate, Department of Computer Science in Economics,
University of Economics, Bucharest, Romania

E-mail: cristian.ciurea@ie.ase.ro

Abstract: *Collaborative systems are widely used today in various activity fields. Their complexity is high and the development involves numerous resources and costs. Testing collaborative systems has a very important role for the systems' success. In this paper we present taxonomy of collaborative systems. The collaborative systems are classified in many categories and there are a lot of criteria for collaborative systems classification. It is presented the importance of testing process in collaborative systems. The paper presents methods, techniques and builds metrics for collaborative systems testing, focusing on collaborative software.*

Key words: *Collaborative Systems; Software Testing; Metrics; Indicators*

1. Introduction

Collaborative systems are an important research field of knowledge-based society and many human activities are involved in this area. Science has great impact on the development of different types of collaborative systems from various activity fields. It is very important to achieve the testing process in every collaborative system in order to assure the good functionality and to eliminate any bug or possible error.

Collaborative systems should work better than other types of systems. This is achieved by:

- *reducing the time waiting in line*, to settle and resolve a specific problem: this proposal applies to collaborative systems involving a large interaction with customers such as banks or stores. Parameters that characterize the process of waiting are: the average number of people waiting in line, the average holding time for a person, the average number of people served and the average serving time for a person. In the case of the collaborative banking system is added also the volume of money traded per unit time;

- *increasing the volume of operations* performed on the mobile, internet banking and other alternative channels of communication, in the case of the banking system. These operations involve a lesser interaction with bank staff, leading to a higher speed of performing transactions. Thus increase the number of people served by the collaborative system, and the number of people waiting in line tends to zero. The implementation of alternative channels of communication is costly for a bank, but the growth rate of profits obtained by the use of these services is upward;
- *reducing employment fluctuations*, in order to increase the stability of collaborative system. A person in a certain position in an institution has accumulated experience that gives a certain stability and safety in the workplace. The staff fluctuation in a bank leads to a decrease in system efficiency and reducing productivity. To reduce this fluctuation, the department of human resources within the bank has a very important role, it can contribute by stimulating and motivating employees;
- *increasing adaptability of staff to the work environment*, in order to increase efficiency and reliability of the collaborative system.

The collaborative systems are developed based on a set of specifications that were defined in the analysis stage in order to establish the goals for the development process. The system must behave and must offer the results that the agents want and that they have established at the start.

Collaborative software architecture is based on distributed systems. That includes a server application, client application and a database server. The client application could be Web based or a rich client, a mobile or desktop client. Figure 1 depicts a general architecture of a collaborative system. It includes all types of clients and servers. The network include wired and wireless transmission medium.

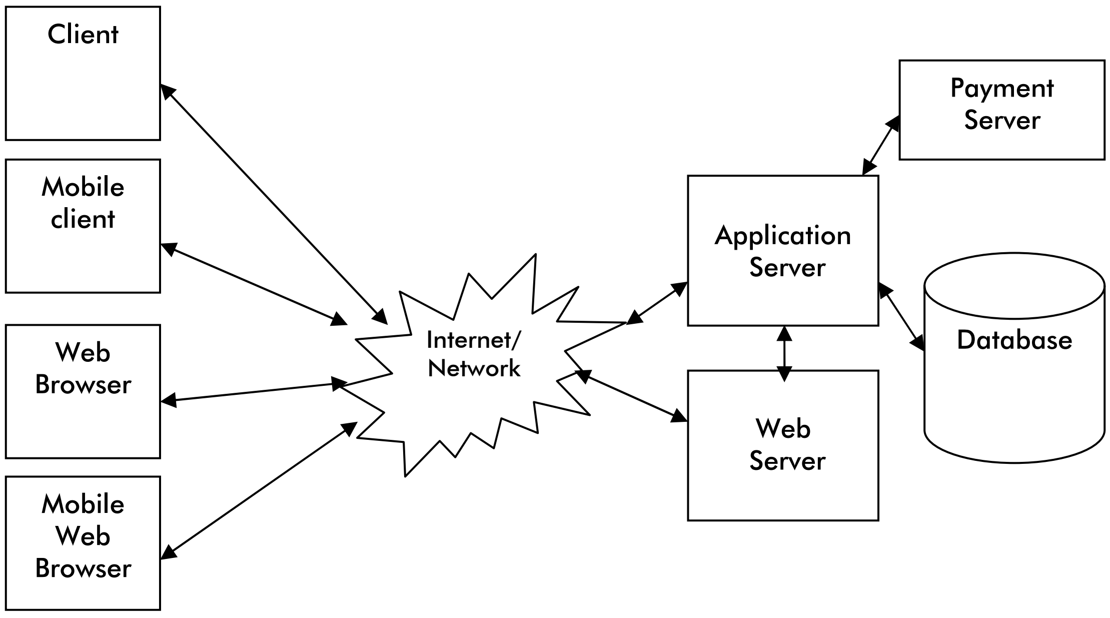

Figure 1. Architecture of a collaborative system

Collaborative systems comprise collaborative software, hardware and all the required personnel. So, testing is done for these dimensions: software, hardware and people. Collaborative system testing uses black-box testing strategy without any knowledge of the system design or code logic. In this paper we will focus mostly on collaborative software testing.

The evaluation of *personnel* involved in collaborative systems could take into account the following aspects:

- education level;
- certification level;
- social abilities;
- experience;
- team homogeneity degree;
- work productivity.

Hardware issues can be the result of incompatibilities between different hardware components of the computer on which the application is used or could be due to hardware faults.

During collaborative software development the testing stage is very important and requires many resources. The following sections in this paper present taxonomy of collaborative systems, methods and techniques for collaborative software testing and several metrics related to collaborative software testing.

2. Collaborative Systems Classification

After the criterion field of application, collaborative systems are classified in: collaborative functional systems, collaborative micropayment systems, collaborative planning systems, collaborative tagging systems, collaborative writing systems, and collaborative medical systems.

Collaborative functional systems include the *collaborative banking systems* and cross all the activities taking place in the economy, providing necessary information and overall coordination for production and finance management.

A banking information system is thus designed to automate a higher set of current bank operations and provide strategic, tactical and operational information necessary in the decision-making process.

The main feature of a modern banking information system is the connectivity level between the factors involved in the banking activity. From this point of view, the banking information systems development supposes the successive or directly implementation of the following types of information systems:

- *banking information systems without connectivity*, which are characterized by the existence of independent computers that run applications specific to certain departments: accounting, credit, etc.; the data transfer between computers is provided, usually through external supports; such information systems are encountered, especially in smaller banking units like branches and subsidiaries;
- *banking information systems with local connectivity*, which are information systems based on local computers networks;
- *banking information systems with global connectivity*, which are information systems based on wide area networks, which connects local networks of the

banking units. [8][4]

All the information systems from a bank are collaborative systems, because they require the cooperation, communication and coordination of many software applications in order to achieve a common goal. This common objective can be represented by the successful processing of a payment order or by the interest calculation of a term deposit.

The banks information system is very complex and very clever, because it must manage client accounts. The bank existence to the market is limited in functionality and maintainability of its information system.

The collaborative banking system is a system with high complexity, with a large number of components and a large variety of links between them. The complexity of the banking system is given by the operations they carry out, but also by the collaboration between different banks from different countries and by the alignment to standards imposed by the regulations in this worldwide field. The collaborative banking system has components that can be represented by using a graph, the nodes being represented by these components, and the arcs by the links between components.

Collaborative micropayment systems guarantees inter-operability, but still allow customers and content providers to use their payment system of choice. In figure 2 is presented a simplified architecture of a collaborative micropayment system.

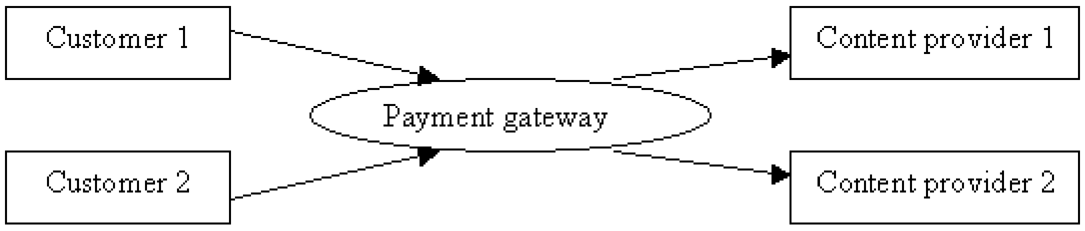

Figure 2. Architecture of a collaborative micropayment system [1]

Collaborative micropayment systems have the potential to provide non-intrusive, high-volume and low-cost pay-as-you-use services for a wide variety of web-based applications. [7]

The high quality services offered by the banks have developed the evolution of collaborative micropayment systems. The banks offer many internet banking solutions and software applications for achieving the development of electronic payments. These services are much secured, including benefits like electronic signature and token authentication.

Collaborative planning systems present the most appropriate way to tackle certain kind of planning problems, especially those where a centralized solving is unfeasible. The main goal is to efficiently obtain a good collective plan. In the figure 3 is shown an example of collaborative planning system. In the proposed example, the agent *w2* has to stack package *d* on top of *a* and package *a* on top of *c*, but packages *a* and *c* are not in the warehouse. Therefore, the only way to achieve its goals is to get packages *a* and *c* in its loading area [3].

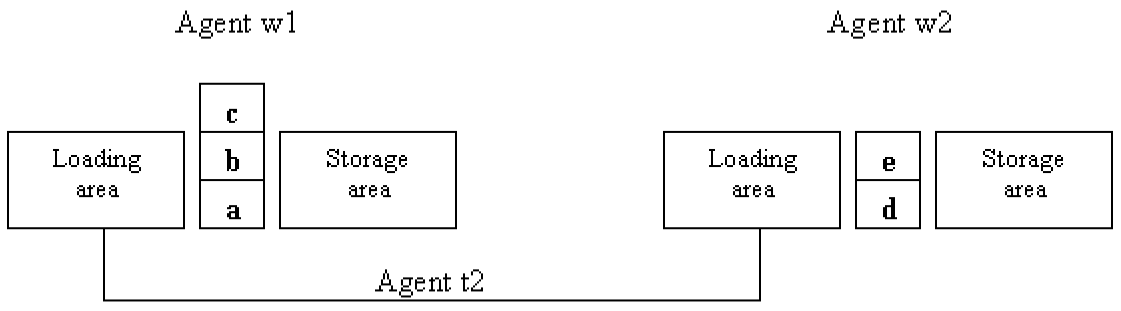

Figure 3. Example of collaborative planning system [3]

Collaborative planning requires capabilities often not found in traditional planning systems. Most importantly, the development of plans must be incremental allowing people to develop plans by focusing on a small part of the plan, exploring options, and making a few decisions before considering the rest of the problem [6].

Collaborative tagging systems provide a new means of organizing and sharing resources. A collaborative tagging system allows arbitrary users to assign tags freely to any documents available on the web [2].

Collaborative tagging systems are nowadays popular tools for organizing and sharing information on the web. While collaborative tagging offers many advantages over the use of controlled vocabularies, they also suffer from problems such as the existence of polysemantic tags [4].

In general, collaborative tagging refers to a system in which users associate keywords, known as *tags*, with various objects or references to objects, e.g., data. Each tag can be user-defined and is usually descriptive of some aspect of the objects to which the tag is associated. A tag can be viewed as a form of metadata in that each tag provides information about the data to which the tag is associated [5].

Currently exist many collaborative tagging systems, but there is the need for a service to integrate the data from the multiple systems to form a large and unified set of collaborative data from which users can have more accurate and richer information than from a single system [15].

Collaborative writing systems, their major benefits include reducing task completion time, reducing errors, getting different viewpoints and skills, and obtaining an accurate text. On the other side, many challenges are raising, ranging from the technical challenges of maintaining consistency and awareness to the social challenges of supporting group activities and conventions across many different communities. For collaboratively writing a document various strategies exist: users can jointly write a document by working closely together or they can work separately, their work being subject to review by other group members.

A collaborative writing system is modeled as follows: it considers *n* sites with each site owning a copy of shared data. When a site performs an update, it generates a corresponding operation. A collaborative writing system consists of a set of participant systems connected by a communication network [10].

Collaborative medical systems, in which modern communication technologies allow doctors from around the world to work on the same patient. In a chirurgical operation each person from the group of doctors has distinct roles.

In [9] it is analyzed a collaborative system model representing a training on different chirurgical activities that is done in a virtual medium. The training is based on the scenario in which the instructor and the trainee are on different locations. The instructor and the trainee share a common virtual space that contains various three-dimensional anatomical models. Each person interacts with the other one through the virtual space and a medical simulation engine describes the physical and logical behavior of objects present on the virtual scene. The interaction is maintained by a multi-modal interface that uses visual 2D and 3D data, voices and audio simulation. Each person is in front of a working table that has a monitor and stereo active pair of glasses. All of these generate a three-dimensional desktop. For collaborative use, it has been implemented a mini broadband system that allows creating a videoconference between persons. The interaction between the instructor and the trainee is based on voice, gestures, chirurgical demonstrative actions, step by step tutorial and simultaneous actions.

Table 1. The main characteristics of collaborative systems

Collaborative System Type	**System Complexity**	**Required Reliability**	**Security requirements**	**Number of concurrent users**
Functional	High	High	High	High
Micropayment	High	High	High	High
Planning	Medium	Medium	Medium	Medium
Tagging	Low	Low	Medium	Medium
Writing	Low	Low	Medium	Medium
Medical	High	High	High	Low

Table 1 summarizes several characteristics of collaborative systems, characteristics that influence the testing process. Each cell with value 'High' needs special attention on testing on that direction.

3. Methods and Techniques for Collaborative Software Testing

Software testing is the process of finding errors in software. There are two main strategies for software testing: white box testing (structural testing) and black box testing (functional testing). For each strategy, many testing techniques strategy were developed. Software testing is a time consuming process and usually complete testing of the applications is impossible.

Collaborative software testing involves two aspects: common testing activities to all collaborative software and specific testing activities depending on the collaborative system type.

Testing collaborative software, as Internet application, requires the following type of testing: functional testing, compatibility testing, content testing, performance testing, load testing, security testing, Web server testing, application server testing and database testing. [11]. Unit testing, integration testing and regression testing need to take place during system development in order to assure high quality software.

Functional testing is needed in order to check if the behavior of the collaborative system acts as specified. The details regarding this kind of testing depend on the nature of the collaborative systems. The main activities involved are:

- link checking;

- forms testing;
- embedded objects functional testing (Flash applications, Java applets, video players etc);
- database transactions testing.

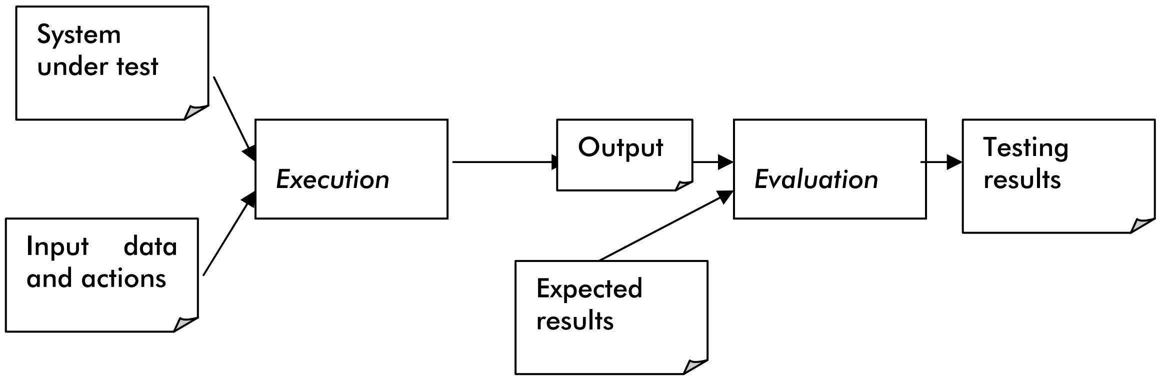

Figure 4. Functional system testing

Functional testing is mainly done automatically using specialized software. Figure 4 depicts the architecture of a functional testing system. The system under test is tested against test data and user actions. The expected results are based on specifications. The evaluation compares the expected result with the actual output and the testing results are displayed and logged.

For each collaborative system type, functional testing is made based on the system's specifications.

Trough *compatibility testing* the Web-based client of the collaborative software layout and behavior are checked on different operating systems and Internet browsers. This kind of testing reveals the problems with HTML and CSS content, ActiveX controls, Java applets, JavaScript and VBScript functions and forms. At this moment there are over 100 combinations between operating systems and Internet browsers. The testing team will focus on the most frequent combination or will test only the recommended combination.

Mobile clients will also be tested for compatibility if the collaborative system has support for these devices. In this field there are also various combinations between operating systems and browsers.

Content testing focuses on the correctness and the layout of texts, images, forms, UI controls and video, animation and sound files in page. This type of testing is made mostly manually and for some components is automated.

The content displayed on mobile devices is also tested, having in mind that there are numerous operating systems (Windows Mobile, Symbian, iPhone OS, webOS, proprietary operating systems, etc.) with specific GUI. Also, mobile web browsers exist for these platforms and their behavior is not the same from device to device or from platform to platform.

It is very important in a collaborative system the way in which the content is displayed, especially where the number of users is very high with a large diversity.

Performance testing is used to measure the behavior of the collaborative systems in various traffic conditions. Depending on the required bandwidth for audio, video, and data transfers, tests can be done using less or required bandwidth. Data is collected and the results are analyzed in order to deal with performance issues.

Mobile devices have less computational power and memory than desktop computers so testing the performance of collaborative systems clients running on these devices is very important. There are tools used for performance testing for mobile environments.

Testing the transactions security is very important, especially for collaborative banking and micropayments systems, keeping in mind that confidential data are used, and if someone has access to these data, it could lead to important financial loses.

Application server testing focuses on testing its functional and structural characteristics. Application server components are tested using classical techniques and testing techniques that take into account the transactions and the asynchronous communications between them.

Every collaborative system uses at least a database, so database testing is required. *Database testing* deals with the checking if the queries and the update operations are executed correctly and if the connection between the application server and the database is reliable. The database integrity within the database server needs to be checked.

Aspects of the certification of the databases used in Internet applications are described in [12].

Web server and application testing focuses on:

- the interaction between the Web server and the application server;
- the interaction between the application server and the database server;
- security issues;
- scalability issues;
- the correctness of the server side scripts execution (PHP, ASP.NET, JSP, etc.).

The application server components will be tested independently and will be integrated within the application and tested accordingly.

Load testing is necessary to check if the collaborative system can manage a large number of users that access it simultaneously within acceptable limits, considering the response time. This will be done using automated tools that creates and simulates multiple clients simultaneously, accessing the systems' resources and measuring the response time and system lag. Collaborative functional and micropayment systems require load testing as the number of expected users is very high.

Also, load testing is made using large amount of data, for example on collaborative writing systems.

Security testing is done in order to be more confident that the system is secure. For that, the systems have to be tested using specific methods and techniques in order to assure confidentiality, integrity, authentication, authorization, availability and non-repudiation [14]. Secure software is good quality software. Not all collaborative software requires the same level of security (for example a micropayment system needs more tests related to security than tagging system). Security testing includes source code analysis, penetration testing, passwords checking [16].

4. Collaborative Systems Testing Metrics

The metrics helps to make a quantitative analysis of the collaborative systems testing. The testing process evolution is represented by the continuous dynamic models: by differential equations or by systems of differential equations, as outlined by a single main indicator or a set of indicators related with the model equations, both among themselves and with the factorial variables which makes the process. Continuous linear dynamic models of collaborative systems are frequently encountered in researching the dynamics of testing processes and are represented by linear differential equations.

In order to evaluate the collaborative systems testing, we have defined several metrics, mainly focused on testing costs.

The number of errors per size of collaborative system (ESCS) shows the efficiency of the testing team:

$$ESCS = \frac{EDS}{CSS},$$

where:

EDS – number of errors detected in the system;

CSS – the size of collaborative system, usually expressed in lines of code (LOC or KLOC) of function points (FP).

The result of a software execution (success or failure) mainly depends on the number of errors that still exist in the software and on the user actions and inputs that are given.

The cost of testing a collaborative system (CT) is composed by costs of testing for each component of the collaborative system added to cost of integration testing:

$$CT = \sum_{i=1}^{NC} CCTC_i + CCITC,$$

where:

NC – number of collaborative system components;

$CCTC_i$ – cost of testing ith component of the collaborative system;

CCITC – cost of integration testing of collaborative system components.

The cost of testing includes regression testing and can be decomposed on testing stages or the resources involved in testing. Integration testing costs are also higher than the integration costs for the classical applications.

For the collaborative systems there are many combinations of components that have to be integrated and tested. The total cost of testing collaborative systems is given by the sum of all testing activities. There are also some overhead costs. The main cost category is the personnel's salaries. Other costs include the costs of the tools and hardware used in testing. The main costs of software testing are described in [13].

The *cost of resources involved in testing (C)* takes into account the category of resources and the cost per unit for each category:

$$C = \sum_{i=1}^{w} NR_i \, d_i \, p_i$$

where:

NR$_i$ – number of resource from the category *i;*
pi – price per unit for the resource category *i;*
di – units of usage for the resource category *i*.

The efficiency of testing method (ET$_i$) is related to the number of errors found:

$$ET_i = \frac{NEi}{NTE} \times k$$

where:

NEi – number of errors found using method *i;*
NTE – number total of errors found*;*
k – coefficient depending on the collaborative system type; it has values from 0 to 1 and it is calculated based on empirical data.

To define operational metrics for collaborative systems testing, it is necessary to accomplish a series of stages:

- validation of measured values for determining if they are correct ;
- definition of exact rules for building test examples;
- guarantee of the comparability of results by using same measuring procedures on predefined factors.

If are considered the collaborative systems S_1, S_2, …, S_n, we can also build other indicators for the implementation of testing metrics for collaborative systems. For each system S_i are collected the data d_{i1}, d_{i2}, …, d_{im} regarding it behavior. Through the intersection of d_{i1}, d_{i2}, …, d_{im} values are obtained some data, which is common to all collaborative systems. These information are necessary to create new indicators I_1, I_2, …, I_h. It selects from these indicators some of them which must be sensitive, stable and representative.

It must be reached equilibrium between the model dimension and its capability to give significant results. The metrics must be not too complicated because it will use lots of resources when implemented and also it must be not too simple because the measured levels will lose relevance.

5. Conclusions and Future Work

The development of collaborative systems conduct to increase their complexity and the global character of the economy is designed to determine a global character for many of the collaborative systems. From the information point of view, to these global collaborative systems must correspond global performance indicators, procurement systems scratchy and data conversion procedures, to transform heterogeneous information into homogeneous entries for aggregate indicators, defined in the metrics of collaborative systems.

Testing collaborative software has many in common with Internet application testing. Collaborative software testing is more complex than testing classical software. When a transaction has failed, there could be many causes for this:

- there are network problems;
- the application has errors; that could be on the server application or on the client application;
- the Web server or application server is poorly configured;
- the database management system is not working properly;
- the database scripts contains errors.

Collaborative software trend to be very complex and the testing effort increases. Also, the testing effort increases by combining many technologies and programming languages in developing collaborative systems.

Today, the application development needs to be made rapidly, so the time allocated for the testing process be shortened. This could lead to poor quality of the applications, but combining automated testing tools and manual testing with other verification activities, and having very good testing plans, the testing process will succeed.

References

1. Allen, J. and Ferguson, G. **Human-Machine Collaborative Planning**, NASA Workshop on Planning and Scheduling for Space, 2002
2. Au Yeung, C., Gibbins, N. and Shadbolt, N. **Contextualising tags in collaborative tagging systems**, "Proceedings of the 20th ACM Conference on Hypertext and Hypermedia", Torino, Italy, June 29 - July 01, 2009, ACM, New York, pp. 251-260
3. Au Yeung, C., Noll, M. G., Gibbins, N., Meinel, C. and Shadbolt, N. **On Measuring Expertise in Collaborative Tagging Systems**, "Proceedings of the WebSci'09: Society On-Line", 18-20 March 2009, Athens, Greece
4. Choi, J., Rosen, J., Maini, S., Pierce, M. and Fox, G. **Collective Collaborative Tagging System**, "IEEE Grid Computing Environments Workshop, GCE '08", November 12-16, 2008, Austin
5. Ciurea, C. **A Metrics Approach for Collaborative Systems**, Informatica Economica Journal, Vol. 13, No. 2, 2009
6. Dai, X. and Grundy, J. **Architecture for a Component-Based, Plug-In Micro-payment System**, Springer Berlin / Heidelberg, Volume 2642/2003, p. 599
7. Ivan, I., Pocatilu, P. and Cazan, D., **Certificarea bazelor de date utilizate in aplicatii Internet**, Informatica Economica, vol. V, no. 2(18), 2001, pp. 71-74
8. Párhonyi, R., Pras, A. and Quartel, D. **Collaborative Micropayment Systems**, Proceedings of World Telecommunications Congress 2004, Seoul, Korea, 2004
9. Pocatilu, P. and Popa, M., **Internet Applications Testing**, "Proceedings of 6th International Conference on Economic Informatics, IE'2003, Digital Economy", Bucharest, May 8-11, 2003, pp. 1028-1032
10. Pocatilu, P. **Costurile testarii software**, ASE Publishing House, Bucharest, 2004, pp. 70-90
11. Pocatilu, P., **Software Security Testing**, Informatica Economica Journal, vol. IX, nr. 4(36), 2005, pp. 78-82
12. Sapena, O., Onaindia, E., Garrido, A. and Arangu, M. **A distributed CSP approach for collaborative planning systems**, Engineering Applications of Artificial Intelligence, 2008
13. Skaf-Molli, H., Ignat, C. L., Rahhal, C. and Molli, P. **New Work Modes for Collaborative Writing**, International Conference on Enterprise Information Systems and Web Technologies-EISWT 2007, Orlando, Florida, 2007

14. Stevenson, D., Hutchins, M., Gunn, C., Adcock, M. and Krumpholz, A. **Multiple approaches to evaluating multi-modal collaborative systems**, CSIRO ICT Centre, Australia, 2005
15. * * * **Employing Organizational Context within a Collaborative Tagging System**, http://www.freepatentsonline.com/y2009/0164267.html, retrieved on June 15th, 2009
16. * * * **Security Testing**, http://en.wikipedia.org/wiki/Security_testing, retrieved on June 15th, 2009

---

[1] **Acknowledgements**
Some parts of this article are results of the project „Doctoral Program and PhD Students in the education research and innovation triangle". This project is co funded by European Social Fund through The Sectorial Operational Programme for Human Resources Development 2007-2013, coordinated by The Bucharest Academy of Economic Studies (Project no. 7832, "Doctoral Program and PhD Students in the education research and innovation triangle, DOC-ECI").

[2] **Paul POCATILU** graduated the Faculty of Cybernetics, Statistics and Economic Informatics in 1998. He achieved the PhD in Economics in 2003 with thesis on Software Testing Cost Assessment Models. He has published as author and co-author over 45 articles in journals and over 40 articles on national and international conferences. He is author and co-author of 10 books, (Software Testing Costs, and Object Oriented Software Testing are two of them). He is associate professor in the Department of Economic Informatics of the Academy of Economic Studies, Bucharest. He teaches courses, seminars and laboratories on Mobile Devices Programming, Economic Informatics, Computer Programming and Project Management to graduate and postgraduate students. His current research areas are software testing, software quality, project management, and mobile application development.

[3] **Cristian CIUREA** has a background in computer science and is interested in collaborative systems related issues. He has graduated the Faculty of Economic Cybernetics, Statistics and Informatics from the Bucharest Academy of Economic Studies in 2007. He is currently conducting doctoral research in Economic Informatics at the Academy of Economic Studies. Other fields of interest include software metrics, data structures, object oriented programming in C++ and windows applications programming in C#.

[4] Codification of references:

[1]	Párhonyi, R., Pras, A. and Quartel, D. **Collaborative Micropayment Systems**, Proceedings of World Telecommunications Congress 2004, Seoul, Korea, 2004
[2]	Au Yeung, C., Noll, M. G., Gibbins, N., Meinel, C. and Shadbolt, N. **On Measuring Expertise in Collaborative Tagging Systems**, "Proceedings of the WebSci'09: Society On-Line", 18-20 March 2009, Athens, Greece
[3]	Sapena, O., Onaindia, E., Garrido, A. and Arangu, M. **A distributed CSP approach for collaborative planning systems**, Engineering Applications of Artificial Intelligence, 2008
[4]	Au Yeung, C., Gibbins, N. and Shadbolt, N. **Contextualising tags in collaborative tagging systems**, "Proceedings of the 20th ACM Conference on Hypertext and Hypermedia", Torino, Italy, June 29 - July 01, 2009, ACM, New York, pp. 251-260
[5]	* * * **Employing Organizational Context within a Collaborative Tagging System**, http://www.freepatentsonline.com/y2009/0164267.html, retrieved on June 15th, 2009
[6]	Allen, J. and Ferguson, G. **Human-Machine Collaborative Planning**, NASA Workshop on Planning and Scheduling for Space, 2002
[7]	Dai, X. and Grundy, J. **Architecture for a Component-Based, Plug-In Micro-payment System**, Springer Berlin / Heidelberg, Volume 2642/2003, p. 599
[8]	Ciurea, C. **A Metrics Approach for Collaborative Systems**, Informatica Economica Journal, Vol. 13, No. 2, 2009
[9]	Stevenson, D., Hutchins, M., Gunn, C., Adcock, M. and Krumpholz, A. **Multiple approaches to evaluating multi-modal collaborative systems**, CSIRO ICT Centre, Australia, 2005
[10]	Skaf-Molli, H., Ignat, C. L., Rahhal, C. and Molli, P. **New Work Modes for Collaborative Writing**, International Conference on Enterprise Information Systems and Web Technologies- EISWT 2007, Orlando, Florida, 2007
[11]	Pocatilu, P. and Popa, M., **Internet Applications Testing**, "Proceedings of 6th International Conference on Economic Informatics, IE'2003, Digital Economy", Bucharest, May 8-11, 2003, pp. 1028-1032
[12]	Ivan, I., Pocatilu, P. and Cazan, D., **Certificarea bazelor de date utilizate in aplicatii Internet**, Informatica Economica, vol. V, no. 2(18), 2001, pp. 71-74
[13]	Pocatilu, P. **Costurile testarii software**, ASE Publishing House, Bucharest, 2004, pp. 70-90
[14]	* * * **Security Testing**, http://en.wikipedia.org/wiki/Security_testing, retrieved on June 15th, 2009
[15]	Choi, J., Rosen, J., Maini, S., Pierce, M. and Fox, G. **Collective Collaborative Tagging System**, "IEEE Grid Computing Environments Workshop, GCE '08", November 12-16, 2008, Austin
[16]	Pocatilu, P., **Software Security Testing**, Informatica Economica Journal, vol. IX, nr. 4(36), 2005, pp. 78-82