An affordable Web-based Grant Management Software Designed to Support Romanian Scholarly Publications

Ionuţ-Eugen SANDU[1,2], Dragoş-Marian SMADA[1], Mihail DUMITRACHE[1,3]*

[1] National Institute for Research and Development in Informatics, 8-10 Maresal Averescu Avenue, Bucharest, 01145, Romania

[2] "Lucian Blaga" University of Sibiu, 10 Victoriei Blvd., Sibiu, 550024 Romania

[3] University of Bucharest - Faculty of Letters, 5-7 Edgar Quinet Street, Bucharest, 010017, Romania

ionut.sandu@ici.ro, dragos.smada@ici.ro, mihail.dumitrache@ici.ro (**Corresponding author*)

Abstract: The rise of Web-based publishing and the increase of electronic usage has triggered profound changes in academic publishing and has determined national governments to take some forms of action designed to actively promote and support the publication of high-quality scientific literature produced in their own country. Considering the spread of ICT technologies nowadays, an online platform was developed in order to encourage participation in the grant competition for Romanian scholarly publications and establish a digital connection between the main actors involved in the process. In this context, the article presents the system architecture, development and outcomes. Building such a platform has required gathering and studying accurate information for data aggregation and user experience, having in mind a scalable design for requirements update or further developments. The system revolves around digitally connecting three major actors – publisher, evaluator, competition organizer. For ensuring the platform's reliability and preventing failures, user-centric tests were developed and set to run automatically at specific triggers, such as code submission. The platform contributes to the transparency of the entire grant management process, substantial reduction of the time for submission, and streamlining of the evaluation procedures. The web platform has been effectively developed, implemented and regularly used by both publishers and reviewers in the latest Romanian grant competitions.

Keywords: Software development, On-line evaluation, Scientific publications, User-centric tests.

1. Introduction

During the last few decades it has been realized that the Information Society must use and serve knowledge, which led to a new term for describing modern society – Knowledge Society. This concept focuses on production, distribution, reuse and the evolution of knowledge. Governments are investing heavily both politically and financially in the knowledge society as a route to economic growth and international competitiveness. Web technology provides an efficient, cost-effective platform for national knowledge management, by supporting the means for knowledge creation and dissemination. In order to encourage the development of high-quality scientific publishing, the governments support knowledge dissemination through grants.

The information society must be supported by the administration, which currently undergoes essential changes. Hence, a central place in the context of government policies and strategies is represented by the adaptation of public administration to the information society which is conducive to the provision of other computerized public services. This implies the expansion and modernization of the national information infrastructure, the development of applications and services based on the convergence of information, communication and media technologies (Banciu, 2005).

With few exceptions represented by international publishing giants such as Elsevier, Springer, MacMillan or Wiley, scientific publishing is not a profitable business especially for small scale publishers. Smaller journals have few (if any) subscribers and therefore higher costs per issue. The rise of Web-based publishing and the increase of electronic usage has triggered profound changes in academic publishing over the past several years. Usually, small-scale academic journal publishers, can hardly cover the costs of publishing scientific, technical and medical (STM) research articles as well as the increased cost of maintaining the electronic infrastructure as most of their publications are open access journals.

That is why in most countries, the central government takes action by actively promoting and supporting the publication of scientific literature produced in their own country. Most often than not each country achieves this goal through its specialized organizations such as Ministry of Education, Ministry of Research, R&D institutes, Research Councils and Academies, Universities, other Sectoral Ministries.

In Romania, one of the main goals of the Ministry of Research & Innovation is to financially

https://doi.org/10.24846/v28i1y201910

support the Romanian academic journals that can meet certain scientific quality criteria in the area of Technical-Scientific Literature. Each year the above-mentioned Ministry organizes a competition intended for receiving grant funding proposals from Romanian publishers who publish technical and scientific books and journals. This financial support is granted and allocated according to certain book and journal rankings and to the overall score they receive after the evaluation process has ended.

Initially, the editors were required to send all the documents by post at the ministry headquarters. After receiving the documents, each proposal had to be evaluated by two specialists, again at the headquarters. The entire process was time-consuming, costly and it discouraged Romanian editors from participating in the competition – a fact that led to the loss of quality publications that could not reach the target readership. In view of the knowledge which can become available to the public, a grant management platform designed to support scholarly publications can be considered an opportunity to contribute to the Knowledge Society by providing the technological means for supporting the grant-funding competitions.

Considering all of the above, we developed a platform for the automatization of the entire procedure with the goal to increase the publications poll and time required to finish the evaluation of the submitted publications. A complex platform was set up to fully meet the requirements of the organizer, but also those of the users of the platform - publishers, evaluators, administrators. By implementing an online solution to manage the information, the users have instant access to their account from anywhere in the world by means of a common internet connection. This provides a high degree of flexibility for the information management (Banciu et al., 2011).

The article presents the development of an integrated solution to support Romanian technical-scientific publications, featuring models, service and integration in Cloud Computing Centre. Section 2 describes the system analysis and development methodology, section 3 presents the system architecture, section 4 describes the technological tools used for the development of the platform and section 5 sets forth an automated testing system focused on user functionalities.

The paper ends with statistical data about the registration and evaluation processes, main benefits and conclusions.

2. System Analysis

In order to develop the technical and functional design of the on-line evaluation platform, the analysis of the Guide for the Evaluation of Technical and Scientific Literature was carried out. This analysis was performed both on the registration and evaluation components, so that business logic into system architecture may be translated.

The main objectives pursued by the development team were the following:

- Defining and designing applications that manage data repositories;
- Designing tools for registering information about the scientific publications which are to participate in the competition;
- Designing a service with facilities to monitor the process of registering and evaluating publications.

System design is based on user needs, and the quality of interaction between users and the website where the service is provided depends on the quality of the system and the quality of the information. The design of the evaluation module was based on data gathered by the members of the Evaluation Committee. A questionnaire was elaborated and distributed to the evaluators for gathering relevant data needed in establishing the development requirements.

To develop the grant management system designed for supporting Romanian scholarly publications, the Agile software development approach was used. Agile represents several iterative software development methodologies where the most important factor is the close collaboration between the members of the functional teams.

The main reason was the urgent need for the platform release, as this methodology appeared as a response to the increased pace of application release imposed by the business environment. Over

the past years, software development has shifted from traditional development models of single major delivery towards models of continuous delivery based on Agile development methods.

The vast majority of organizations that adopted Agile reported that their organization realized success from agile projects (VersionOne, 2017). Through this approach, the software is developed in multiple versions, which ensures its releases on time established. The adoption of Agile methodology led to the building, testing and releasing of software in a fast way. It also reduces costs, time and the risk of making critical production changes, allowing for incremental upgrades of the production system. Using Agile, you can easily develop, create and resolve any problems or errors occurring in uncertain environments (Beck et al., 2001).

But Agile has some disadvantages such as the following:

- The planning may not be accurate;
- Team members must have solid knowledge in more than one area of the project;
- Active involvement and collaboration of developers is required throughout the project;
- The final product can differ from the initial planning - new iterations can be added during its development based on user feedback.

The development technique used was Scrum, which is a subset of Agile and one of the most popular processing frameworks for Agile implementation. It is an iterative software development model used to manage complex software and product development (Smartsheet, 2018).

Many of Scrum's development processes cannot be predicted and therefore, several teams can participate in the development of the final product in a series of defined sections during the planning called *sprints* (Vlaanderen, 2011). After the end of each sprint, the development team members begin planning the steps to launch a future software version. In its original form, Scrum is designed for small interdisciplinary teams of about six to nine developers. An important property of any Scrum team is self-organization: i.e., the team itself has the authority to decide on strategies to achieve the objectives of the sprint (Hron & Obwegeser, 2018). A team is managed by a Scrum master that coordinates the meetings and task allocation.

According to yearly report conducted "State of Agile Report" (VersionOne, 2017), Scrum and combinations of Scrum with other techniques are by far the most common agile techniques used for software development, totaling more than all other techniques combined. Among the major benefits offered by Scrum the following can be mentioned (Smartsheet, 2018):

- A higher transparency and visibility of the project: daily meetings allow the team members to know who is doing what;
- Increased team responsibility: there is no classic project manager but only a Scrum master who has no any authority over the team, instead he must trust the team that he is managing and never impose them any methods for doing their work. Team members collectively set the number of tasks that can be achieved in each sprint, leading to a close collaboration;
- Increased adaptability to changes: having periodic feedback after the ending of each sprint, make it easier to cope with changes;
- Increased cost savings: by releasing features in smaller chunks the bug fixes can be performed in the next sprint without the accumulation of major and costly problems.

On the other hand, the Scrum also has some associated risks, such as delivery time failures or increased costs that may be by poor planning of tasks or unclear specification of project goals.

3. Architecture

The process of designing the architecture was done iteratively in three steps:

- **capturing functional requirements** by using the competition guidelines and the results from questionnaires as sources
- **designing the software architecture** by defining the structure, responsibilities and interactions of the software components;
- **validating the architecture** by going through the current and subsequent list of requirements

and by checking that the architecture designed in the previous stage(s) allows the implementation of the requirements.

The platform operates in a changing environment due to evolving requirements which are generated by various new needs and by varying requirements imposed by the competition's organizer. Both applications and middleware were designed for coping with changes. In order to allow scalability, the internal structure of the system must be made accessible.

The provisions of this competition revolve around digitally connecting three major actors – *the organizer*, *the publisher* and *the evaluator*, as presented in Figure 1.

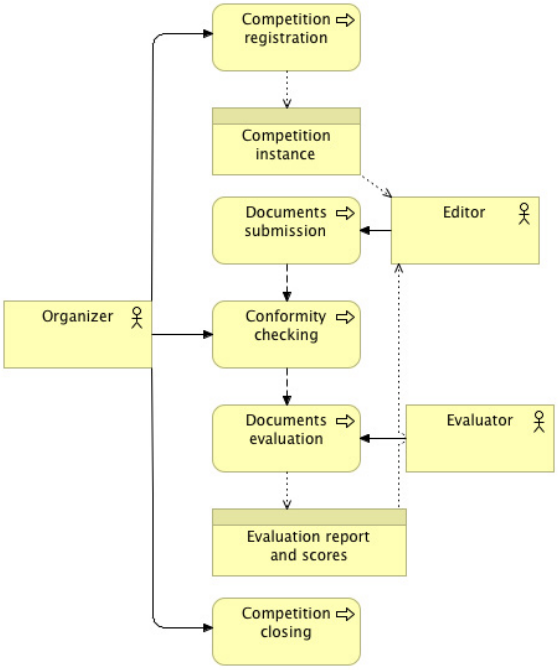

Figure 1. Logical workflow

After the organizer launches the competition, any interested editor has the option to create an account and register scientific publications. The registration requires the submission of documents which vary depending on the publication type – book or journal. After the submission period ends, the publications are distributed to the evaluators. Each publication must be reviewed by two editors. The evaluators must perform a conformity checking, in order to assess if all submitted documents are compliant with the grant competitions rules. In case of failing the conformity checking the publications are rejected. The evaluators fill in the evaluating report and give scores based on the criteria specified by the organizer. After all the publications are checked for conformity and evaluated, the results are reported to the organizer which will decide upon the ending of the competition and publish the final results.

The architecture for the platform (Figure 2) relies on the **Middleware** paradigm, which means that the software uses a middleware component to link other components. The middleware is the software layer between the operating system and other applications, in a distributed computing system, which help to develop modular applications that can communicate with each other through middleware (Krakowiak, 2007).

From the application's user perspective, the middleware is completely hidden. Users interact with the application and are not aware of how information is exchanged internally. As long as it works flawlessly, the middleware must remain a hidden structure.

The most important features of the middleware are:

- Setting up connections with web servers and web services.
- Downloading and extracting the required information from documents.
- Validating the integrity of the received information.
- Presenting all the information in a uniform way so that it could be reused by the local applications or integrated with them.

The benefits which result from using the middleware for application development are the following:

- hiding the distributed nature of the application. The platform can be viewed as a collection of interconnected components that are operational and can be distributed in different locations;
- masking the platform heterogeneity, including the hardware used, server operating systems, or communication protocols;
- ensuring a uniform, standardized, high-level interface for application developers so that the platform can be easily re-used and interoperable;

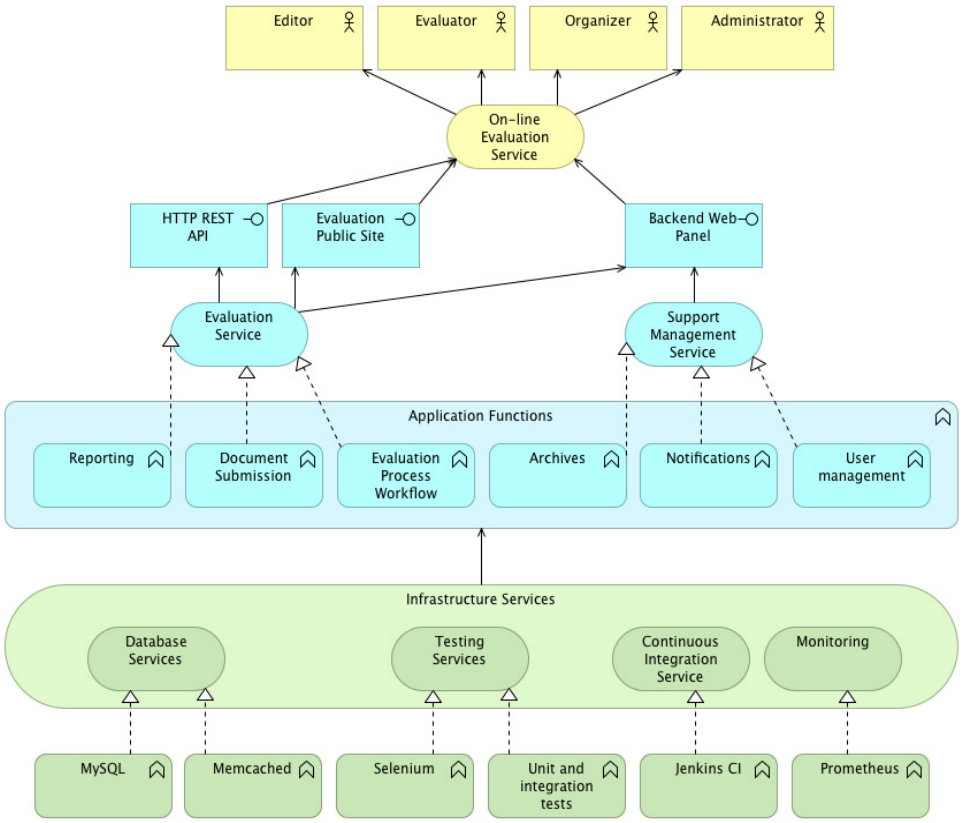

Figure 2. The platform's architecture

- providing a set of common services used in order to avoid increased development efforts and to facilitate a possible collaboration with other applications or web services.

Application services are accessible via a REST interface - Representational State Transfer – which is an architectural style that sets the restrictions, such as a uniform interface, which, if applied to a web service, renders the properties desired by the developer, such as performance, scalability and changeability, which allow the REST service to work best on the Web.

In REST architectural style, data and information are accessed as resources by using the Uniform Resource Identifier (URIs), which are web-specific links. Resources are driven by the use of a set of simple and well-defined operations. REST has a client / server architecture which uses a stateless communication protocol, usually HTTP. In REST, clients and servers exchange resource representations using a standard interface and protocol (Fielding, 2000). The data that the client instructs the server to process resides in the URI and the operation performed by the server on the data is described directly the means of the HTTP method (see Table 1).

Table 1. HTTP methods in REST

Method	Description
GET	retrieves all the data that is identified by the Request-URI.
POST	creates a new resource on the existing URI.
PUT	replaces all current representations of the target resource with the request payload
DELETE	deletes the specified resource
CONNECT	establishes a tunnel to the server identified by the target resource
OPTIONS	requests information on communication options available on the request/response chain
PATCH	applies partial modifications to a resource
HEAD	asks for a response identical to that of a GET request, but without the response body

In recent years, the number of organizations that have started to use the services offered by Cloud Computing has increased. The evaluation platform is included in the ICIPRO Cloud Computing

Centre; therefore, it benefits from the major advantages provided by a Cloud infrastructure.

In order to provide electronic quality services to citizens, ICIPRO allows public institutions to host computer systems used for their activities.

4. Technological Tools Stack

For the development of middleware, we used *the Django framework* which consists mainly in a collection of libraries written in Python language. Python is a dynamic multi-paradigm and multi-functional programming language used for application development. Python emphasizes the cleanliness and simplicity of the code, and its syntax allows developers to express some programmatic ideas in a clearer and more concise manner.

Django is an open-source development framework designed for web-based applications. It uses the Model-View-Controller (MVC) architectural model and supports four main databases (MySQL, PostgreSQL, SQLite and Oracle). Apart from it, there are also other software libraries that offer support for other SQL and NOSQL databases.

Django can be used to build almost any type of website and can integrate with any client-side framework and deliver content in almost any format (such as HTML, RSS feeds, JSON, XML, etc.).

As a database management system, we used *MYSQL*. It is a relational database management system, which means that a database stores data in several separate tables, rather than storing all data in the same place.

CodeIgniter and *Bootstrap* were used for the development of user interface. CodeIgniter is a PHP framework built containing a set of tools to create full-featured web applications. It contains many libraries, helpers, plug-ins and other resources that handle complex PHP procedures and functions; if simplifies PHP syntax and rationalizes code underlying webpages. CodeIgniter is a MVC based system, with complete database classes integrating support for multiple platforms, active database support, a custom routing system, validation of forms and data and XSS security and filtering.

Bootstrap is a very popular HTML, CSS and JS framework for developing responsive web projects. Bootstrap has a large collection of tools which contain HTML and CSS templates for forms, buttons, printing, navigation, and other interface-related components. For monitoring purposes, Prometheus, an open-source time series database and alerting toolkit is used. It is a data model which uses time series data which can be identified by metric name and key/value pairs. It collects the time series data via a HTTP pull model and has several graphics modes and support for dashboards.

5. Automation of User-centric Tests

Agile project development promotes the idea of self-organizing and self-managing teams, requiring software engineers to embrace behaviors of flexibility, empowerment, trust, and collaboration (Calefato & Ebert, 2019). The adoption of Agile methodology in the development process involved short cycles of software development. This approach leads to frequent releases of the source code. From the software developers' perspective, the quality of an application which is meant to complete the development cycle in a time-saving and cost-effective manner, is ensured by the ease with which the application can be modified, by minimizing the number of situations where maintenance is required for the application, and the level at which the application attracts new users (Zamfiroiu, Boncea & Petre, 2018), (Zamfiroiu, 2018)**.**

For an online application, client experience represents the most important factor, as it is decisive for its success. During the software development process, the team benefited from one of the most important features of the Agile methodology namely: - accepting changes to requirements during any phase of the SDLC (Software Development Life Cycle), which makes it more flexible and highly adaptable to dynamic environments where requirements change frequently (Al-Zewairi et al., 2017).

In the continuous development of the software platform, the testing process is an important condition for propagating changes in the production environment. Test automation is

required to eliminate major manual test problems such as: time loss, error reproduction, share repetition of actions (Smada et al, 2018).

Frequent software launches can cause many errors in the application's source code and may affect the user-side performance; automated testing systems are the solution that can prevent defects in user behavior.

An adequate test automation solution reduces the time and the amount of resources needed to ensure the reliability of the software application behavior while increasing the quality of the tested application. Test automation also aims to recognize and address weaknesses, otherwise the systems will become vulnerable to cyber-attacks (CÎRNU et al., 2018). An automated testing system has certain major advantages, such as:

- ensuring a better experience for users, due to the increased reliability;
- reducing the workload for the development team. Code reviewing is done faster since reviewers only check the code that is validated by automated testing;
- in the event of creating a code that affects customer behavior, the submitted code will not pass the test system as it will be automatically rejected so that code failures do not reach the production environment;
- ensuring the correct operation for all user levels in the evaluation platform.

In most cases, new technologies and software products, will not provide just benefits but they will also involve a number of risks such as failures or increasing security breaches, so when releases are performed complex tests are imposed. The classic test mode assumes that tests are often repeated each time the source code changes and other situations such as changing the operating environments or hardware settings. An automated testing tool can play pre-recorded and predefined actions, compare results with expected behavior, and report the success or failure of these manual tests to a test engineer. Once automated tests have been created, they can be easily repeated and extended in order to carry out tasks that are impossible to perform by manual testing.

The technological stack for automating tests is the following:

Selenium - is an open source software solution used with other tools to automate web browsers on different platforms. It supports several programming languages such as Python, Ruby or Java, etc. (Selenium, 2019).

Selenium IDE – provides the interface for recording the user actions while they are performed. The user actions are then translated into Selenium specific commands. The tests can be exported in the form of reusable script files that can be called from a server.

Jenkins - an open source solution written in Java, used for running the tests without the need of user assistance. Jenkins can monitor any job defined as a cron, SVN or GIT (Version Control Systems). A continuous integration server is designed to automatically or manually trigger complex workflows in order to build, test, and deploy software components.

For developing the automated test system, a document containing all the operations that users can perform on the platform was first elaborated. The test assertions in the document were derived from the OASIS Test Assertions Guidelines Version 1.0. The document was updated whenever a change in the web-side platform is required.

The user actions were recorded, based on the key elements included in the system pages and forms, such as authentication, menus, web links, submit buttons, or confirmation/warning text boxes. The recordings were done in Mozilla Firefox browser and exported as Python2 script files. These ran heedlessly on a continuous integration server, under Linux Centos v6. The tests simulated the browser behavior without the need of displaying it.

Atlassian Stash, a Git repository management solution for enterprise teams, was selected in order to create, organize and discuss competition-related issues. Jenkins was configurated in order to build the test suite automatically when a pull request is performed on the Stash Server. Stash Webhook was configured on the Stash server to trigger Jenkins automatically. In the workflow, a developer creates a branch on his local repository, completes its task, then commits and pushes the changes to Stash and creates a

pull request in order to propagate the code into production. When the pull request is merged, Jenkins is triggered automatically in order to run the Selenium tests. The code is pushed in a development environment, which is different from the production environment, that Jenkins uses for running the tests and in the event of failure it will reports reports the errors to the specified reviewers and developers.

6. Result Evaluation and Interpretation

The platform was developed, implemented and tested at National Institute for R&D in Informatics. The Romanian Ministry of Research & Innovation has adopted the system and thus it has been operating online for four years during grant competitions. The figures related to the registration of publications on the platform are presented in Table 2.

Table 2. Publication submission per year

	Editors	Books	Journals	Evaluation period (days)
Y 1	80	286	48	21
Y 2	51	482	58	21
Y 3	75	641	67	28
Y 4	77	433	63	28

Only slight differences are noticed for the journals registered in the competition each year, as these are periodic publications and are applying for subvention each time.

The decrease in book submissions starting in the fourth year of the above-mentioned competition was due to the fact that subvention was granted to publications in previous years (Table 3).

All the participating academic publications were registered according to the following research fields:

D1 - Mathematics and the natural sciences

D2 - Engineering Sciences

D3 - Socio-human sciences

D4 - The science of life and of the Earth

D5 - Agricultural sciences

D6 - Medical sciences

Table 3. Distribution of publications by field

Domain/year	Y1	Y2	Y3	Y4
D1	8	29	41	37
D2	78	111	125	96
D3	8	22	29	34
D4	219	338	464	279
D5	6	14	26	28
D6	15	26	23	22

The high number of publications submitted in the field of Social and human sciences determined the organizer to adapt the Evaluation Committee by including new evaluators with social expertise in order to enable it to cope with the vast amount of publications in this field.

7. Conclusion

The activities carried out in the present research led to the development of functionalities dedicated to an on-line platform dedicated to the evaluation of the Technical and Scientific Literature. The goal of this research was to facilitate the process of recording and evaluating scientific publications.

The main components of the platform are: authentication, publishing, book registration, magazine registration, publication eligibility, and evaluation, monitoring and reporting.

User-centric tests were developed with a view to ensuring platform reliability and preventing behavioral bugs.

The main benefits of the on-line platform for evaluating the Technical and Scientific Literature are: the transparency of the entire evaluation process, substantial time reduction in submission procedures, and the streamlining of the evaluation procedure by abandoning the classical methods that involved an additional effort for publishing houses, for the evaluation committee, but also for the organizer of the competition that was approached in this paper.

The above-mentioned platform contributes to the increase of the researchers' expertise in the technical-scientific field by providing access to scientific and technical publications which

have been evaluated by experts involved in the Scientific and Technical Literature Review Committee of the Consultative College for Research, Development and Innovation on Scientific and Technical Literature.

Acknowledgements

The authors wish to express their gratitude to the members of the *Evaluation Committee for the Romanian Technical-Scientific publications* and to the main promoter of the platform, Professor Doina Banciu, PhD. We are also grateful to Dr. Niculescu Andrei, the Managing Editor of *SIC* for his valuable advice and guidance on bringing both content and style of our manuscript up to the standards established by this journal.

REFERENCES

1. Al-Zewairi, M., Biltawi, M., Etaiwi, W. & Shaout, A. (2017). Agile Software Development Methodologies: Survey of Surveys, *Journal of Computer and Communications*, *05*, 74.

2. Banciu, D. (2005). Vectorul informațional în societatea cunoaşterii, *Revista Română de Biblioteconomie* şi Ştiinţa *Informării*, *Anul 1*(nr. 2), 14-16.

3. Banciu, D., Petre, I., Smada, D. & Anghel, M. (2011). Developing an Interactive System to Provide Management Support for Transportation Research Organizations, *Studies in Informatics and Control*, *20*(4), 421-428, ISSN 1220-1766, DOI: 10.24846/v20i4y201111

4. Beck, K. et al. (2001). Manifesto for Agile Software Development, *The Agile Alliance*. Available at *<http://agilemanifesto.org/>*, Accessed in December 2018.

5. Calefato, F. & Ebert, C. (Jan.-Feb. 2019). Agile Collaboration for Distributed Teams [Software Technology], *IEEE Software*, *36*(1), 72-78.

6. Cîrnu, C., Rotună, C., Vevera, A. V. & Boncea, R. (2018). Measures to Mitigate Cybersecurity Risks and Vulnerabilities in Service-Oriented Architecture, *Studies in Informatics and Control*, *27*(3), 359-368, ISSN 1220-1766, DOI: 10.24846/v27i3y201811

7. Fielding, R. T. (2000). *Architectural styles and the design of network-based software architectures*, PhD Dissertation. University of California.

8. Hron, M. & Obwegeser, N. (2018). Scrum in Practice: an Overview of Scrum Adaptations. In *51st Hawaii International Conference on System Sciences (HICSS 2018)* 10.24251 / HICSS.2018.679 (10 pg).

9. Krakowiak, S. (2007). *Middleware Architecture with Patterns and Frameworks*. Electronic book, available at < http://lig-membres.imag.fr/krakowia/Files/MW-Book/index.html >, accessed in December 2018.

10. Selenium, *Web Browser Automation*. <http://www.seleniumhq.org/>, accessed in January 2019.

11. Smada, D., Rotună C. I., Boncea R. & Petre I. (2018). Automated Code Testing System for Bug Prevention in Web-based User Interfaces, *Informatica Economică*, *22*(3), 23-32, DOI: 10.12948/issn14531305/22.3.2018.03

12. VersionOne (2017). 12th State of Agile Report, *State of Agile*. Available at <https://explore.versionone.com/state-of-agile/versionone-12th-annual-state-of-agile-report>, accessed in January 2019.

13. Vlaanderen, K., Jansen, S., Brinkkemper, S. & Jaspers, E. (2011). The agile requirements refinery: Applying SCRUM principles to software product management, *Information and Software Technology*, *53*(1), 58-70.

14. *What's the Difference? Agile vs Scrum vs Waterfall vs Kanban*, <https://www.smartsheet.com/agile-vs-scrum-vs-waterfall-vs-kanban>, accessed in January 2019.

15. Zamfiroiu, A. (2018). Security Management for Mobile Learning Systems. In *International Scientific Conference eLearning and Software for Education*, *vol. 1* (pp. 42-48). National Defence University "Carol I".

16. Zamfiroiu, A., Boncea, R. & Petre, I. (2018). Determinarea calității aplicațiilor mobile pe baza modului de dezvoltare, *Romanian Journal of Information Technology and Automatic Control-Revista Romana de Informatica si Automatica*, *28*(1), 35-46, ISSN 1220-1758.