Modern Single Page Application Architecture: A Case Study

Veronica GAVRILĂ*, Lidia BĂJENARU, Ciprian DOBRE
National Institute for Research and Development in Informatics,
8-10 Maresal Averescu Avenue, Bucharest, 01145, Romania
veronica.gavrila@ici.ro (**Corresponding author*), lidia.bajenaru@ici.ro, ciprian.dobre@ici.ro

Abstract: This paper presents the latest technologies and methodologies available today with a view to creating a high performance, unique and user-experience centric platform. The Client application is a Single Page Application (SPA) and is the most complex component due to its extremely modular architecture. This is not a simple webpage but a stand-alone application, divided into components according to their role. The modular and complex component-based architecture proposed at all levels of the application, ranging from the client all the way up to the storage level, underlies the INTELLIT platform. The INTELLIT platform aims to provide an easy access to information about the life and work of Romanian authors, the most important moments in our culture, the complete calendar of events from 1994 to 2000, the canonical work of different national authors. The new approach also addresses the identification of new ways to modulate and structure such a platform, both on the client side and on the server side, so that individual testing of the various components and any subsequent changes can be achieved as easily as possible. The main reasons these technologies and models were chosen are performance, low code duplication, modularity and reuse of components.

Keywords: Literary patrimony, Digital platform, SPA, PWA, MVVM, VueJS, Modular architecture.

1. Introduction

In the context of the digital era, literary heritage is diminishing more and more and, in this situation, efforts must be made to highlight and preserve the Romanian literary heritage. Cultural heritage is important because it strongly influences the sense of identity, loyalty and the behavior of every people in the world. Preoccupation for the preservation of cultural heritage through archives, libraries, museums leads to a higher awareness of the complexity of culture (Banciu,2015).

Under these circumstances, the INTELLIT platform is proposed as a solution that is both a portal open to the digital era and a possible answer to the problems of preservation and conservation. INTELLIT aims to facilitate quick access to literary works, manuscripts, bibliographies, dictionaries of Romanian literature and many other documents.

As a result of the continuous development and ascension of technology, a wide selection of devices that are able to access the Internet has been made available nowadays. These devices will continue to increase in both number and functions, as the world of technology is now focusing on IoT and other interconnected devices that will also have various forms and functions.

One of the most important problems is represented by the lack of accessible ways to display and interact with information using devices that have various screen sizes and resolutions.

A well-thought-out solution for the above-mentioned problem should take into consideration not only the scaling of the presentation layer and various controls that are on-screen but also the design of the application or web site with the responsiveness in mind.

There are two different roads to take when starting to develop a responsive application from the ground up. One can either implement a minimalist layout and later adjust it for various screen sizes or take the path recommended by the industry and create a mobile first design. In the latter path, it is much easier and logical to adapt something that is designed for a small responsive screen to a bigger screen.

The INTELLIT platform is a novelty in the field not only in terms of functionality offered but also in terms of design and user interface. The platform is built around a highly responsive and modular architecture using a MVVM (*Model – View – ViewModel*) frontside component-based framework called *VueJS*. This framework allows the creation of certain reusable components, ranging from layouts and data containers to various user facing controls and data presentation components.

The main advantages of using the frameworks mentioned in this article are the performance bonuses (one can avoid the code duplication and send to the client only the data that has been changed or the data that is needed to generate the requested view) and the modular and dynamic architecture (pages / components are dynamically created from other components, without duplicates).

Further on, the components can be designed to be responsive based on each other and the layouts to change based on the parameters taken into account (screen-size, screen resolution, etc.). By using such architecture, adding responsive components and various other layouts is easily done. The testing possibilities for the platform modules represents an important advantage of using this modular structure. Automated tests can be written for each component, together with integration tests for wider scoped structures.

All of the things mentioned are achieved by using the global layout objects (described in the following chapters) on which the dynamic grid system of the application is based on as well as the client application namely SPA (*Single Page Application*) / PWA (*Progressive Web Application*).

The opportunity to study representative cultural works in a digital format will support the access of future generations to the existing creations of these people and the continuity of Romanian language. Moreover, the traditions will be a true pillar of support for education in Romania.

This paper is structured as follows. Section 2 describes several national and international similar literature platforms and the encountered problems. Section 3 is an overview of the present modular platform architecture with all its components and a description of all the layers presented. Section 4 describes the technologies and methodologies used in the development process highlighting all the features that these methods and technologies offer and the main reason why they have been chosen in this platform. These technologies are presented based on the platform layers. In Section 5 the overall features that the platform offers by solving the common and uncommon problems encountered today in most platforms are described. Section 6 draws a conclusion and summarizes the basis for a future work.

2. Related Work

The National Strategy for the Digital Agenda in our country provides clear objectives in order to support the preservation of national cultural heritage and to provide digitized cultural information to Europeana (Europeana Collections, 2019) by 2020. There are several ongoing initiatives, including programs supporting the development of digital libraries, access to past or current cultural information.

The developed various platforms give users access to Romanian culture, but they have incomplete functionalities, are difficult to access, and less friendly. Following the analysis of the existing platforms, a number of functional requirements have been extracted that will serve as a basis for the new development platform. In addition, the user's requirements that would facilitate access to information and provide a more enjoyable experience, thus involving people's interest in literature, have been identified (Ciurea, 2019).

Among the similar Romanian platforms, the OPAC online catalogue (Catalogul online OPAC, 2019) of the Romanian Academy Library (Romanian Academy Library, 2019) can be mentioned. It contains bibliographic information about the documents, starting with 1998 and, now, it includes about 450,000 records.

Also, another important platform is the online catalogue of the National Library of Romania (Biblioteca Naţională a României, 2019) which contains bibliographic information about books, multimedia documents, articles from periodical literature. The National Digital Library is made up of digital collections created by digitizing documents from special collections of the National Library, organized on themes or events.

There are platforms on the territory of Romania that provide access to information but do not have effective search mechanisms or do not find documents in text or pdf format; the information is not sufficient for a particular topic (Bajenaru et al., 2017).

A great benefit from the digitization of Romania's cultural heritage is the preservation of its cultural creations.

At international level, there are also many platforms for accessing the cultural information. In this respect, some of the most representative platforms, which represent the reference model proposed in this paper, are mentioned below.

The European Digital Library (Europeana collections, 2019) is a source of information provided by the European cultural and scientific institutions, promoting the preservation of cultural heritage through the transfer of knowledge, innovation and technology. It has just over 30 million digitized objects available and it represents a unique access point to millions of books, paintings, films and museum objects, archival documents that have been digitized in the whole Europe.

The British Encyclopedia (Encyclopedia Britannica, 2019) is now only digital, and the new opportunities created by the online environment have enabled the improvement of the quality and quantity of information, thereby enhancing visibility and popularity. Currently, the British Encyclopedia includes several online platforms with content that is geared towards the geographic region, as, for example, the British Encyclopedia of Australia (Encyclopedia Britannica Australia, 2019) or towards the areas of interest (and the age groups) such as Britannica for Children (Encyclopedia Britannica for children, 2018).

The proposed INTELLIT platform is developed on the basis of the functional requirements extracted after analysing the existing platforms in the field and responding to the user's requirements.

The goal of the present work is to highlight everything that a culture can define by providing access to information while using an efficient and friendly digital solution. The new solution is based on the latest technology and standards, offers a wider range of tools in order to meet the needs of users, and can also provide an affordable environment that will last over the years.

The original aspects of this work are represented by the ingenious combination of the innovative technologies used and by the highly modular architecture of the platform. As a result, new ways to modulate and structure such a platform have been discovered, both on the client side and on the server side, so that individual testing of the various components and any subsequent modifications can be accomplished as easily as possible. With regard to the processing side advanced methods of processing texts extracted from the provided documents have been implemented and discovered.

3. Platform Architecture

The INTELLIT platform is developed based on the desire to preserve and highlight the Romanian literary heritage due to several problems that inevitably lead to the definitive loss of the elements that define Romanian culture. The general concept of this ambitious project consists in briefing the public and leaving a legacy of Romanian literature in time (Simion, 2018a).

The information available in the platform comes from the data sources provided by the Romanian Academy Institute and contains 4 type of documents:

- *General Dictionary of Romanian Literature* (A-Z): an academic work for the public, from Romania and from abroad, which is interested in the phenomenon of literature (Simion, 2018b).
- *Timeline of the Romanian Literary Life* (1944-2000): a complex research work, unique due to its magnitude of information, and to the quality of its interpretative approach. The main feature of this research work is the inventory and exhaustive presentation of all Romanian periodicals reflecting actions and tendencies that have a direct or indirect impact on the political ideology of the aesthetic domain and the establishment of political control over literary creation. The aim of this study is to provide a more detailed picture of the events, mentalities, options from the periods 1944-1957, 1965-1967, 1990-1996 regarding the reconfiguration of the Romanian landscape, as it results from the respective publications (Simion, 2018a).
- *Canonical Works of Romanian writers* will be shared by authors, and grouped in order to offer a more relevant sorting of literary genres. The main literary genres (lyrical, epic and the dramatic) will form the basis of the search in this context for the platform. (Diaconu, 2013).
- other structured and unstructured documents and works of our canonical authors that are no longer bound to the copyright laws.

The new INTELLIT Platform is built on 3 levels, each level with its own structure and ramifications and consists of 3 different applications specific to each level (Figure 1):

- *view level*: represented by the client application (MVVM Client Architecture).
- *application level*: represented by the server application which consists of the *Representational State Transfer* (*REST*) application server and its associated middlewares and integrated systems (ORM, Data Processing System, Authentication Systems, Roles and Permissions System, etc.) required for the platform to run according to the related specifications.

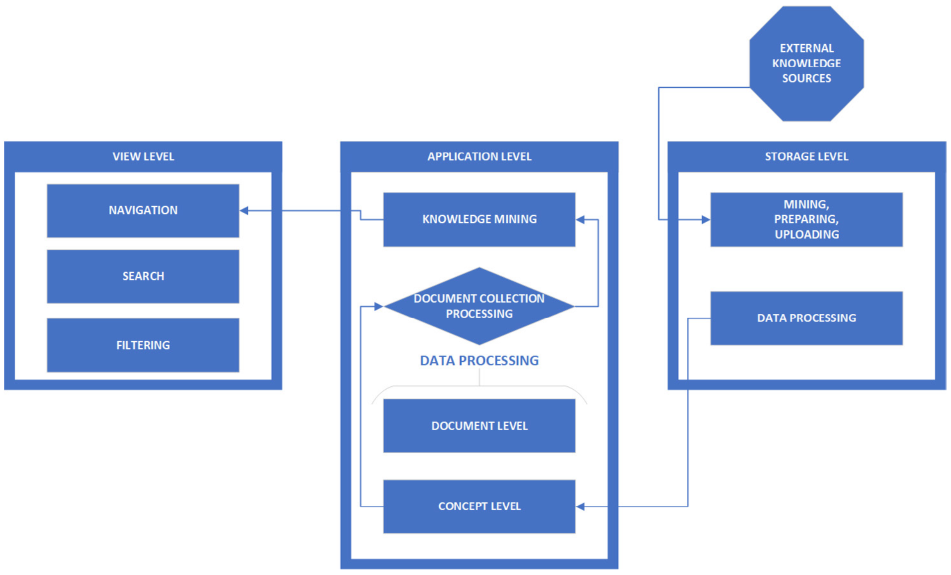


Figure 1. INTELLIT Architecture

- *storage level*: represented by the storage warehouse (*ElasticSearch*) where all the documents provided are stored and eventually modified by the authorized users and by the other database (*MongoDB*), used for static document processing, static data storage, users and permissions systems.

On the server side, this approach provides a higher level of security because the server application and the storage level are not accessible and communicate with the client application only through the *REST Application Programming Interface* (*API*) and various security and authorization middlewares, thus reducing the risk of malicious attacks. This approach also ensures high modularity and it could as well reduce the time required to maintain or modify the platform by adding new services and features in the future.

On the client side, the platform is built around the modular architecture known as MVVM, a frontside component-based framework called *VueJS*.

4. Technologies Used and Features

In this chapter the main technologies used are presented. These are grouped according to the application layers that they represent and to some of the advantages provided by using these specific frameworks starting from performance bonuses (reducing the pages and bundles size, code duplication, component reusability, low code duplication) and leading up to the modular architecture and dynamic components.

4.1 Client Application

The Client application is a *Single Page Application* (SPA) and is the most complex component due to its extremely modular architecture and it's not a simple webpage but a stand-alone application, divided into components according to their role.

Highlighted features:

- a view is dynamically created from the components according to the user's actions

- a router through which specific components / views are provided thus all pages are based on dynamic components that change and are updated automatically without the need to reload (refresh) the page

- all elements on the page, transitions and animations are changed dynamically and individually depending on which component changes without affecting the rest of the elements.

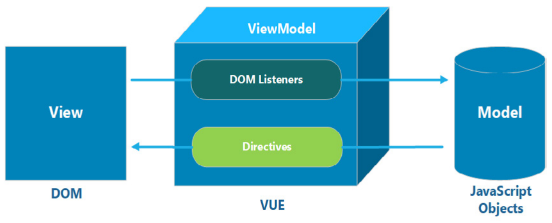


Figure 2. Client Application Architecture

The client can be considered as having its own structure, a MVVM (*Model-View-ViewModel*) model as it is exemplified in Figure 2.

The view in this case is represented by the DOM (Document Object Model) itself, the link between it and the Model (represented by the JavaScript objects in the application) being intermediated by the *VueJS* (The Progressive JavaScript Framework, 2019) instance of the application / component itself, with all the related methods. In the above example, two-way data binding is also exemplified. Changes to the DOM are observed by *VueJS* listeners who update the Model. Modifications to the model are reflected in the DOM through the *VueJS* Guidelines.

In addition to being a SPA application, the INTELLIT platform is also a *Progressive Web Application* (PWA) (Ater, 2017). Under these circumstances the developed application will benefit from all the features of this type of structure that is focused on the gap between the native apps and web apps and tries to reduce it by enabling some of the major functionalities on both versions:

- native behavior (it can be installed as a regular application on the operating system where the platform's native API's and behavior can be accessed and used on any device / Android / Windows operating system / iOS.
- offline caching
- offline function based on cache (mainly to avoid critical errors when there is no internet connection and use the caching system in such a case)
- application shell that dynamically generates the page / component layout and structure, even if the data isn't cached or loaded (application shell is the first component that is loaded)
- adapting design
- service workers (cache the content in the background and provide also other secondary functionalities)
- application manifest which is read by the browser and indicates that it is a PWA application (contains everything ranging from the way the browser interprets the application to the individual icons and colours that the application will use when installed)

The main reasons these technologies and models have been chosen are their performance, low code duplication, modularity and component reusability.

This approach provides modern utilities for various issues that can be encountered in the development process (filters for data in the presentation level or two-way bindings between data and presentation level - such as filters for data in the presentation level or two-way bindings between data and presentation level (consequently, in situations where this functionality is used, the data changes will be followed by changes in the presentation level associated with the changed data, and vice versa). It also provides one of the most innovative and easy to use reactive systems.

4.2 The Server Application

The server application is *RESTful*, stateless and modular with its own router and middlewares. The main technology used is *NodeJS* (NodeJS, 2019) and the framework used is *KoaJS*, the successor of *ExpressJS*, alongside many libraries / packages that are going to be used to ensure a modular, complex architecture.

On this application server, all the mechanisms and methods necessary for optimal operation will be implemented with a focus on modularity, as well as advanced security, synchronization, and real-time data visualization methods (Simsons, 2015a).

Highlighted features:

- modular and complex architecture
- real-time functionality
- asynchronous calls
- high scalability (each service can be automatically divided into several nodes to handle a large number of requests without blocking or slowing down the application)
- REST endpoints and specific logic
- data synchronization
- optimized and minimized

4.3 The Storage Level

The storage level consists in storing, retrieving and modifying the data from the documents provided. The process of extracting the information lies in different data processing and extraction methods, unique criteria and categories for every type of document (dictionary, timeline or authored works) based on the information in these documents.

The main technology used here is *ElasticSearch* (ElasticSearch, 2019). Documents are indexed based on a specific set of rules, so there is no need for a full text search in the documents, the system can find that value in the already created index. This reduces query execution time based on these already created indexes.

The secondary part of this level is represented by the database used for User Roles and Permissions and static data storage, mainly powered by *MongoDB* (MongoDB, 2019).

5. Platform features

On the client side *Vuetify* was used (Material Design Component Framework, 2019) on top of *VueJS* (Kyriakidis & Maniatis, 2017) which offers a variety of new features based on the basic ideology of component reusability.

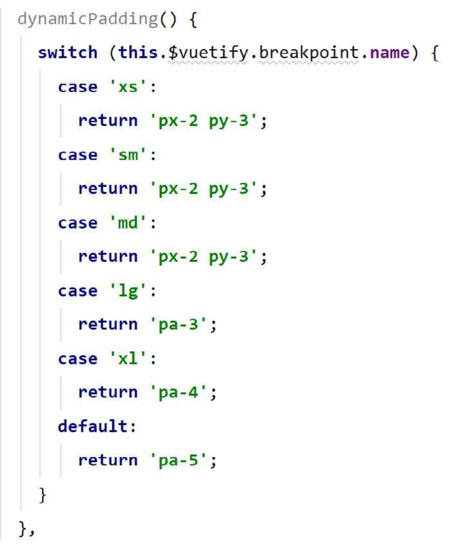
Vuetify, like other major responsive design frameworks has a 12-column grid system. This system is built using flex box but has fallbacks for some older browsers. The grid system is used in order to create a different application layout. It contains 5 standard types of media breakpoints that are used for targeting specific screen sizes or orientations. This media breakpoints are not CSS *(Cascading Style Sheet)* level and are represented by a global framework specific dynamically updated layout object.

The grid chain is: v-container > v-layout > v-flex and each of them is a flex box element.

Personalized layouts that can accommodate any type of application, screen or view port can be designed.

In the INTELLIT platform the object layout system presented above was used to control the spacing between the elements on the pages by modifying it based on the breakpoint the page reached, thereby achieving the dynamic padding ideology that allows the page to adjust to every screen (Figure 3).

Beside the margin and the padding, almost every component was checked and adjusted to a specific screen. This method is used along with the enforcement of the class properties (for example checking if a certain breakpoint has been reached and adjusted according to the rules stated in the class). Depending on the screen resolution the elements in the pages have a different behavior and its modified according to the size (Figure 4 and Figure 5).



```
dynamicPadding() {
  switch (this.$vuetify.breakpoint.name) {
    case 'xs':
      return 'px-2 py-3';
    case 'sm':
      return 'px-2 py-3';
    case 'md':
      return 'px-2 py-3';
    case 'lg':
      return 'pa-3';
    case 'xl':
      return 'pa-4';
    default:
      return 'pa-5';
  }
},
```

Figure 3. The layout system for dynamic padding

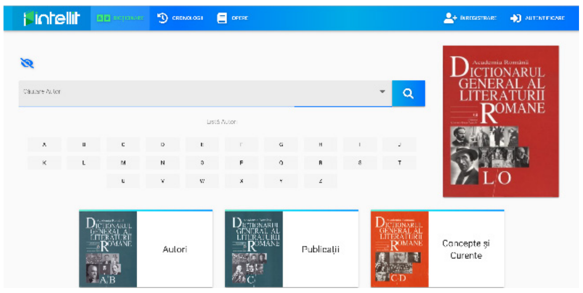


Figure 4. Page displayed on a wider screen

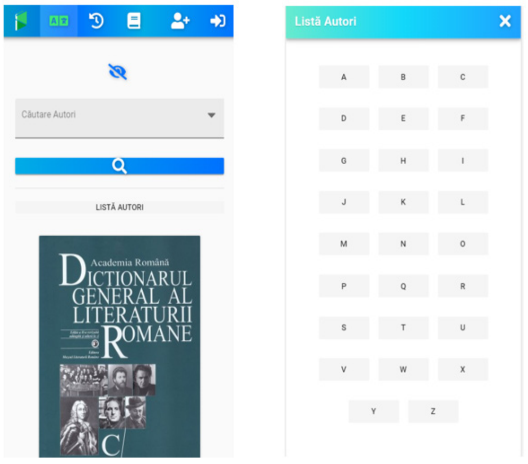


Figure 5. Page displayed on smaller screen (mostly smartphone, tablets etc.)

We also implement some custom CSS utility classes that are structured according to the BEM (Block – Element – Modifier) System. These classes help us achieve a minimal code duplication and also increase the dynamism of the layouts and other components (we toggle these utility classes based on certain parameters).

One major point in our architecture is structuring our components in two categories:

- presentational components that just take the input data (as props) and present them to the user;
- container components that format and process the date and then deliver it to the underlying presentational components.

By using such an approach, we can separate these functions into individual components, leading to an increase in modularity.

Also, the time necessary to load the page is a vital part of providing a responsive user experience. Studies show that most users stay between 3 and 10 seconds to wait for a page to load or they don't use it anymore. Thus, it has officially become a ranking factor for mobile searches and has been one for the desktop for a while now (Average Page Load Times, 2018).

The last few years were all about mobile, most users were more comfortable using a smartphone than a desktop, just as the recent studies shows, so the pages need to be optimized and reduced in size, the fonts and other extension need to be minimized in the production version so that it won't require a considerable amount of time to load (Mobile Page Speed, 2018).

In the INTELLIT platform all these problems are resolved. The platform is built around the PWA standards presented in the last chapter with the all the features offered by this kind of configuration and thus by minimizing and optimizing the JavaScript and CSS bundles we have achieved a 59% reduction in file size, from the initial 328 kB bundle to 135 kB.

To mitigate the various problems presented above webpack 4 with our custom-made configuration and other resource loaders was used. Beside the application shell that is loaded and cached the very first time someone uses the application, all other content is split into chunks specific to different routes. This is done by using a special webpack directive with comments on each JavaScript route one intends to split up into chunks (it will also generate a unique hash as a file identifier). These chunks will pe preloaded and they will only be loaded and cached the moment someone enters that specific route. That way major performance boost can be achieved but without loading the whole application content and routes all at once.

Another major point is regarding how one should approach resource loading, more specific, images. All of the static images included in the platform are automatically converted into much smaller versions of themselves by using an imagine manipulation library that is tied to a webpack image loader. They are afterwards dynamically converted to base64 and injected into the generated template code. This way, no matter how large the final images are the layout stays in place without distorting the page and it also shows clear signs of progression (from grey square to smaller image and then to final image). These images (Figure 6) are also cached when first loaded.

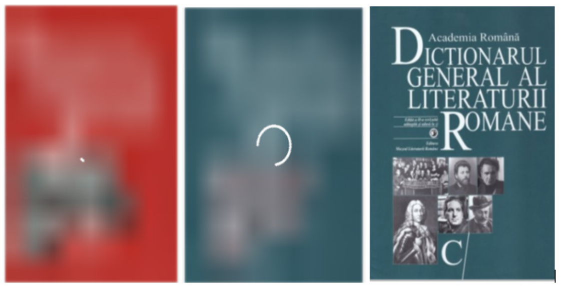


Figure 6. Image Loader

Finally, every bit of CSS and JavaScript in the application described above was minimized and optimized in order to substantially reduce the bundle size.

6. Conclusion and Future Work

The main goal of the INTELLIT platform is to provide easy access to information starting from Romanian author's life, complete biography and timeline, highlight of the most important moments from our culture, complete timeline for the events in the 1994 until 2000, canonical work from our authors and many more.

The primary achievement of this project was the analysis and compilation of the newest available technologies and methodologies in order to create a high performance, unique and user-experience centric platform. This goal was also achieved by implementing a complex component-based architecture on all application levels, ranging from the client all the way up to the storage level.

All of these points were harmoniously achieved and implemented through the new INTELLIT platform as a result of the modular based approach and a complex and highly scalable architecture.

As such, these results are not just about various functionalities implemented or offered but also focused on creating and maintaining a pleasant user experience regardless of the devices used.

The future developments include the processing side, algorithms and methods to process the extracted texts, to normalize the text as well as developing the rest of the platform components (the timeline and canonical works modules).

Acknowledgements

This work was supported by a grant of the Romanian Ministry of Research and Innovation, CCCDI – UEFISCDI, project number PN-III-P1-1.2-PCCDI-2017-0821 / NR. 54PCCDI ⁄ 2018, within PNCDI III, "Romanian literary patrimony preservation and valorization by using intelligent digital solutions for extracting and systematization of knowledge" (Prezervarea și valorificarea patrimoniului literar românesc folosind soluții digitale inteligente pentru extragerea și sistematizarea de cunoștințe).

REFERENCES

1. Ater, T. (2017). *Building progressive web apps: bringing the power of native to the browser*. O'Reilly Media, Inc.

2. Average Page Load. Available at: <https://www.machmetrics.com/speed-blog/average-page-load-times-websites-2018/>, last accessed: 2018.

3. Bajenaru, L., Marinescu, I. A., Tomescu, M. & Savu, D. (2017). National Library of Programs: A New Approach to Management of Software Products, *Romanian Journal of Information Technology and Automatic Control - Revista Romana de Informatica si Automatica*, *27*(4), 25-38.

4. Baldwin, C. Y. & Woodard, C. J. (2009). The architecture of platforms: A unified view, *Platforms, markets and innovation*, 32.

5. Banciu, D. (2015). *Educaţie şi cultură în era digitală*. Bucureşti, Editura Niculescu.

6. Biblioteca Naţională a României. Available at: <http://www.bibnat.ro/>, last accessed: 2019.

7. Ciurea, C. & Filip, F. G. (2019). Virtual Exhibitions in Cultural Institutions: Useful Applications of Informatics in a Knowledge-based Society, *Studies in Informatics and Control*, *28*(1), 55-64, DOI: doi.org/10.24846/v28i1y201906

8. Diaconu. V. (2013). Canonul literar, *Revista Ramuri*, 8.

9. ElasticSearch. Available at: <https://www.elastic.co/>, last accessed: 2019.

10. Encyclopedia Britannica. Available at: <https://www.britannica.com/>, last accessed: 2019.

11. Europeana collections. Available at: <http://www.europeana.eu/portal/en>, last accessed: 2019.

12. Kyriakidis, A. & Maniatis, K. (2017). *The Majesty of Vue. js*. Packt Publishing Ltd.

13. Material Design Component Framework. Available at: <https://vuetifyjs.com/en/>, last accessed: 2019.

14. MongoDB. Available at: <https://www.mongodb.com/>, last accessed: 2019.

15. NodeJS. Available at: <https://nodejs.org/en/>, last accessed: 2019.

16. Page Speed. Available at: <https://think.storage.googleapis.com/docs/mobile-page-speed-new-industry-benchmarks.pdf>, last accessed: 2018.

17. Romanian Academy Library. Available at: <http://www.biblacad.ro/>, last accessed: 2019.

18. Simion, E. (coord.) (2018a). *Cronologia Vietii Literare Românești*. Academia Română.

19. Simion, E. (coord.) (2018b). *Dicţionarul General al Literaturii Române*, Ediția a II-a revizuită, adăugită și adusă la zi, 2 vol. (A-B; C-D). Academia Română.

20. Simsons, K. (2015a). *You Don't Know JS*, 1 edition February 23. O'Reilly Media.

21. Simsons, K. (2015b). *You Don't Know JS*, 1 edition December 17. O'Reilly Media.

22. The Progressive JavaScript Framework. Available at: <https://vuejs.org/>, last accessed: 2019.