Site Generator for Small Open and Private Online Courses

Anton. A. Sutchenkov [1], Anton I. Tikhonov [2]
[1, 2,] *Physics and Technology of Electrical Materials and Components Department*
[1, 2] National Research University "MPEI"
[1, 2] Moscow, Russian Federation
[1]SutchenkovAA@mpei.ru, [2]TikhonovAI@mpei.ru

***Abstract*— The article analyses approach for developing Small Open and Private Online Courses (SOOC and SPOC). The technology and the program for assembling electronic educational resources from heterogeneous fragments, including video clips, images, texts, html, pdf files are considered. The program creates a static site with a navigation structure ready for use in the learning process. You can work with a static site on a personal computer, a tablet or a smartphone. In addition to the static sites generator a server program that provides user and content management is considered. It additionally supports interactive Python applications with the user interface automated generation based on input and output parameters description.**

Keywords—electronic educational resources, assembling from heterogeneous fragments, sites generator, Small Open Online Courses (SOOC) and Small Private Online Courses (SPOC)

I. Introduction

Currently, MOOC – Massive Open Online Courses [1], SPOC – Small Open Online Courses and SPOC – Small Private Online Courses [2] are widely used in the learning process. MOOC are usually created by teams of professionals, the cost of MOOC development and support can reach hundreds of thousands of dollars, that is justified in cases when thousands of users interact with the course. The first difference between MOOC, SOOC and SPOC is the number of students interacting with the course. The second difference MOOC from SOOC and SPOC is a high degree of automation since it is impossible to manually check the knowledge for large groups of students. The third difference is high quality of multimedia content and the availability of interactive applications. Access to SOOC is free, but only university students can work with SPOC.

Universities have to develop a lot of online courses for small groups of students working with them. Therefore, the development and maintenance of e-courses is included in the professional duties of teachers and is carried out with minimal support from professional developers.

SOOC and SPOC include heterogeneous content: text documents (curriculums, lecture notes, support documents for practical and laboratory classes, source code examples, a fund of evaluation tools (tasks, questions, tests, course projects), images and video. All this heterogeneous content should be organized so that students are comfortable working with it. In addition, SOOC and SPOC have to be updated each year when the curriculum and course content are changed.

II. Requirements for SOOC, SPOC Development and Support Systems

We will formulate the main requirements for SOOC, SPOC development and support systems.

- Low entry threshold for SOOC and SPOC development without any need to master new technologies and tools.
- Low labour-intensive development and maintenance of SOOC and SPOC.
- Ability to work with heterogenous content including plain text, HTML, Markdown, PDF, video, images, program source code with syntax highlighting.
- Ability to assemble SOOC and SPOC with only modifying the table of content (TOC), adding and/or replacing fragments of learning content.
- Ability to quickly make changes in SOOC and SPOC content.
- Automated creation of a uniform design for all SOOC and SPOC pages.
- Automated creation of navigation structure based on the table of content.
- Ability to work with SOOC and SPOC with any type of computing devices without installing additional software.
- Ability to work with SOOC and SPOC both online and locally without Internet connection.

The above requirements are mostly self-evident, let's focus only on the fact that the content of SPOC and SOOC includes a lot of text and multimedia fragments.

Hundreds or thousands of heterogeneous fragments included in SOOC and SPOC should be assembled and made easy to use in the learning process. Tools for assembling e-courses should be simple and easy to use for assembly, testing and deployment.

Finally, it is important to be able to use SOOC and SPOC with all types of computing devices not only in classrooms of the university or at home, but also without an Internet connection on the way to university.

III. SOOC and SPOC Development and Support Technologies

Currently, several approaches to the development and SOOC and SPOC maintenance are used. The main is

978-1-7281-5655-2/20/$31.00 ©2020 IEEE

Learning Management Systems (LMS) [3-6]. Content in this case is developed interactively using built-in LMS editors. LMS are server-side web applications, that require permanent connection to the Internet. The result is an electronic analogue of a traditional textbook, that can be relatively easily supplemented with multimedia content and tools to test students' knowledge. It should be noted that it is hardly possible to create a high-quality SOOC or SPOC from scratch without training the developers.

Another class of tools for creating electronic textbooks is Content Management Systems (CMS). Currently, there are a lot of CMSs [7-11] designed for the development and management of educational content. The solution of specific tasks and extensions of CMS functionality are provided with plugins. As a rule, content management in CMS is carried out interactively by means of editors and administrative system.

IV. Video Content Development Technology for SOOC and SPOC

As experience shows, the minimum labor intensiveness in the development of educational content provides the video clips creation. This can be done using a limited set of equipment: a notebook, a headset, and a video camera to record what happens outside the computer. The minimum set of software includes office program (MS Office, Libre Office, WPS Office) for presentations development and text editing, video capture program, for example, TechSmith Camtasia Studio [12] or iSpring Suite [13]. You can use only free software: Libre Office and iSpring Free Cam.

The development technology for video clips is quite simple. First, the course is structured into logical units, lasting no longer than 5-10 minutes [1]. This is due to dissipation of students' attention after 5 minutes of watching video.

Structuring the course into small modules improves the perception of the educational material and simplifies video clips creation. This allows to develop SOOC and SPOC in the process of preparation for the lessons. Splitting the course into small modules makes it easier to change the course content, as no video editing is required.

Before starting SOOC and SPOC developing, one should prepare presentation templates and other documents. It is also necessary to select the video format (frame size, number of frames per second). Usually these parameters are specified in the corporate policy of the university

It is also necessary to make decision on how the SOOC and SPOC will be deployed. The following main cases are possible:

- SOOC is stored locally, distributed on DVD and portable devices.
- SOOC is accessed via the university corporate network, so it is necessary to ensure the bandwidth of the corporate network for operation at peak loads.
- Educational content can be stored on public video hosting such as YouTube. Video hosting have almost unlimited content storage capacity and a content delivery network (CDN) infrastructure. In this case, the ability to manage access to educational content is completely or partially lost, and the continuity of the educational process becomes dependent on the video hosting. This leads to the need to provide alternative ways of content delivering to support the continuity of the educational process.

This approach leads to the creation and maintenance of a lot of video and text files and requires at the planning stage to develop the structure of SOOC catalogues. In addition, developers must maintain a list with files descriptions and unified resource locators (URL). One should not forget about content fragments of other types: programs of disciplines, lecture notes, questions and tasks for checking knowledge, source code of programs, etc.

V. An Assembling Approach to SOOC, SPOC Development

When developing SOOC and SPOC, the main task is to assemble heterogeneous content so that it could be easily handled by students. The problem is divided into the following sub-problems:

- creating web pages with text, images, video clips, pdf files embedded into them;
- creating a uniform set of themes and design styles for all SOOC pages;
- creating a navigation structure and search tools to navigate through the learning content.

It is desirable to automate the assembling of SOOC and SPOC based on the information that should be collected when planning and creating fragments of content.

Nowadays similar tasks are solved by headless CMS or generators of static sites [14-19], for example, Jekyll, Pelican or GitBook [14]. These programs focused on solving narrow tasks with certain types of content.

Common to them is the generation of static web pages linked to each other by hyperlinks. Analysis of existing headless CMSs showed that they do not meet the requirements discussed in Section II.

It should be noted that static sites have limited user and content management capabilities, so from now on, we'll just talk about SOOC.

Organization of heterogeneous content fragments can be done on the basis of a linear-hierarchical structure of SOOC, similar to the traditional book structure (Fig. 1).

When working with SOOC there is always a table of contents (TOC) in front of the user's eyes, that allows to determine where he is in the SOOC structure.

Each SOOC page is an html page with built-in multimedia content. The html page serves as a universal container. The embedding of content into the page should be done automatically.

The TOC is responsible for the hierarchical part of SOOC structure. It is multi-level, the organization of the hierarchy and the choice of the number of hierarchy levels depends on the developer. The main table of contents is dynamic: the user can roll it up and down.

For each level and sublevel of the hierarchy a special page with local TOC is generated. It contains hyperlinks to sub-sections and pages located at the lower level of the hierarchy.

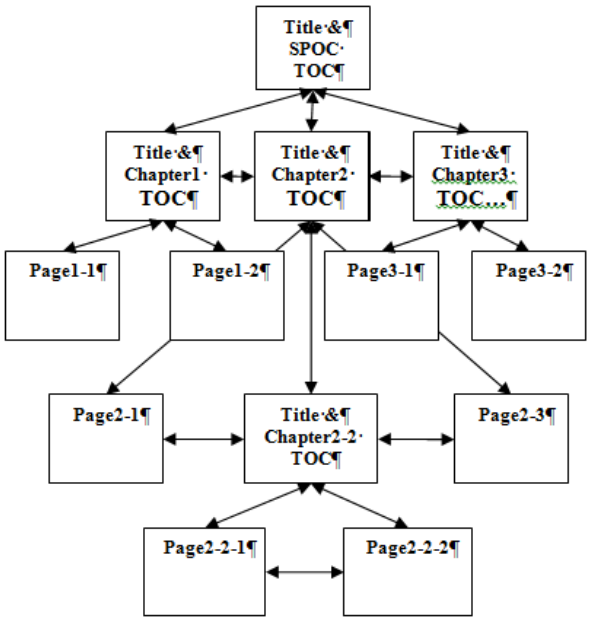

Fig. 1. SOOC linear-hierarchical structure

In turn, each page (except for the first and last page) contains links to at least the previous, next page. The SOOC TOC should either be displayed in a separate panel or, if this is not possible, for example, when viewing SOOC on a smartphone, there should be a link to the main TOC displayed in a pop-up panel.

The linear component of navigation is flipping, allowing user to work with the SOOC page by page.

This SOOC organization is familiar to users, it is easy to work with, while the user always knows where he/she is in the SOOC structure.

VI. SOOC Structure Description

There are many ways to describe linear-hierarchical structures. The choice of approach is determined by the convenience for the SOOC developers. For example, the description of site structure with xml language is used in Asp.Net Forms technology. For a web application an xml file Web.sitemap is created. The root tag is <siteMap>, web application pages are described with <siteMapNode> tags. Page properties (unified resource locator, name, description, keywords, users and groups for whom the page is available) are specified as attributes of the tag. The hierarchy is set of pages and sections descriptions inside tags <siteMapNode>...</siteMapNode>. The linear sequence is specified with the order of page nodes descriptions. Other ways are the use of json and yaml languages. Common to the above ways is the use of languages that are easy to process by software systems, but it takes time and efforts to learn them, and it is easy to make mistakes developing SOOC structure.

We decided to use Excel spreadsheets with a fixed sheet structure. Other structure description formats mentioned above are optionally supported.

The advantage of using Excel is obvious as it is a widely used familiar tool. The main drawback is that Excel lacks convenient tool for hierarchical structures support.

The position of the page in the SOOC hierarchy must be set explicitly by the level number. So the root of the hierarchy (course name) corresponds to level 1, chapters to level 2, subchapters level 3, and etc.

Each SOOC page corresponds to a row in a sheet containing the following cells:

- title – the brief description of the page displayed in the browser window title and SOOC TOC;
- description – the detailed description that is displayed in the page title when it is displayed to the user and in the TOC as a hint;
- ctype – the type of content displayed on the page, the following content types are currently supported:
    - text – unformatted text emdedded in the page;
    - html –html content;
    - code - source text of the program;
    - pdf – pdf document embedded into the page;
    - image – image (.png, .jpeg, .gif) embedded into the page;
    - video – a video clip stored in the local file system with a 4x3 aspect ratio;
    - video169 – a video clip stored in the local file system with a 16x9 aspect ratio;
    - youtube – video clip published on YouTube with 4x3 aspect ratio;
    - youtube169 – video clip published on YouTube, with 16x9 aspect ratio;
    - rel – reference to a sheet; this type of content is used to structure the TOC, e.g., chapters of SOOC can be placed on separate sheets of a spreadsheet; the name of the sheet is specified in the url parameter;
- keywords – comma-separated keywords (tags); used for searching information in the SOOC;
- url – unified resource locator for the content fragment both for local resource and published online;
- author – author(s) of the content fragment (optional parameter);
- lvl – level in the SOOC hierarchy;
- roles – list of roles (user groups), which are allowed to access the fragment (used only in SPOC), if left blank, the fragment will be available to all users.

VII. SOOC Assembly Application

The SOOC assembly application [20] is written in Python and generates a single page web application (SPA) with three inline frames (Fig 2).

A multi-level TOC is displayed in one of the frames. The location of this frame can be changed. Page content is displayed in the main frame. The description of the page is displayed in the frame located above the content frame.

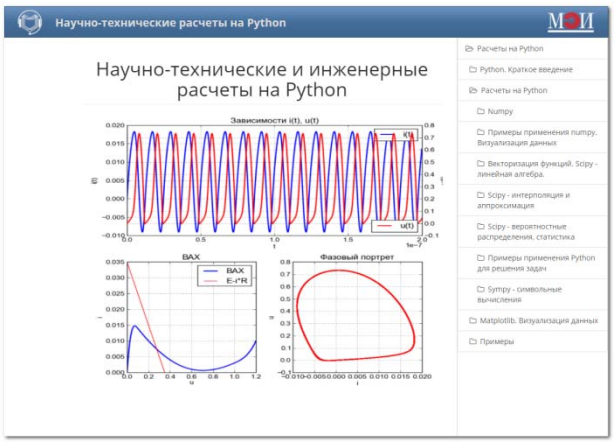

Fig. 2. SOOC single page application

The web application is adaptive. When the screen size is reduced, the frame with the table of contents is no longer displayed, and navigation buttons appear in the upper title bar of the application (Fig. 3).

The structure of the SOOC generator is shown in Fig. 4.

During the assembling process, the content fragments are checked. The content check option can be disabled because it is time consuming.

Warnings and error messages are written to the text log file. Serious errors may stop the assembling process.

As mentioned above, any changes and additions to the SOOC require reassembling, so a differential assembling is implemented to speed up process. The new version of SOOC is formed on the basis of the archive with the previous SOOC version, only the modified fragments are checked and compiled.

The result of the SOOC assembly is a zip archive with the static site. SOOC can be deployed by unzipping the zip archive at either a local computer or a web server. In the first case, it is necessary to run index.html file in any browser, and in the second case to access the web server.

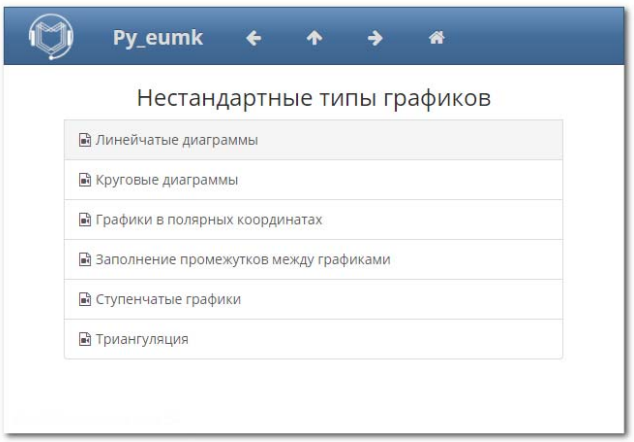

Fig. 3. Local table of content on small screen

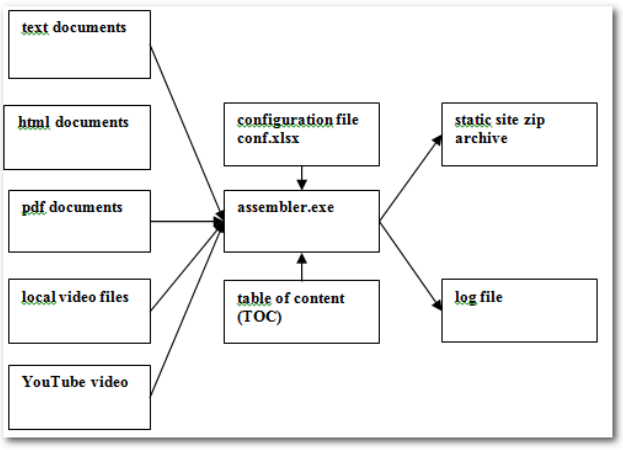

Fig. 4. SOOC generator structure

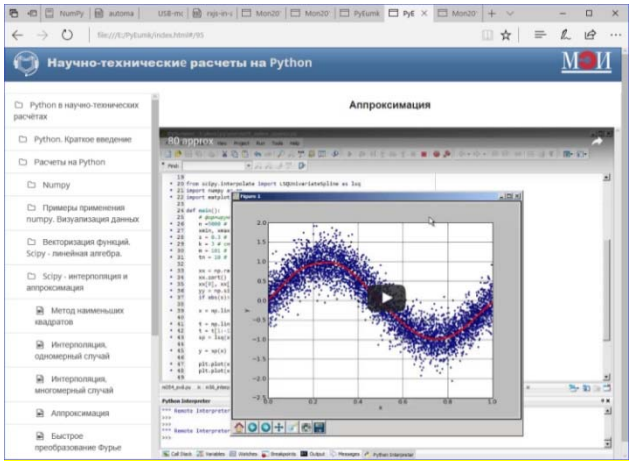

Fig.5. SOOC page with a video published on YouTube

SOOCs assembled using the proposed technology can work with hybrid content, when part of the content is hosted locally and part is published online. Fig. 5 shows the SOOC page with a video published on YouTube.

VIII. Server Side Application

However, the static nature of assembled SOOCs does not allow to include dynamic content, including interactive applications and virtual lab work.

This made it necessary to create a server application. Server application can use the same TOCs and configuration data as static SOOC generator.

Assembling is carried out dynamically. At the moment when the web-application starts the structure is read and the table of contents is formed. On the user demand the page containing a fragment of content is formed on the fly, links to the previous and next pages, a fragment of the table of contents is also is embedded in the page.

Server-side application can manage users and content. Any changes are displayed immediately after TOC description is changed.

Using server side application, it is easy to implement individual versions of SOOCs and SPOCs, for example, to providing access only to authenticated users or only in the university campus.

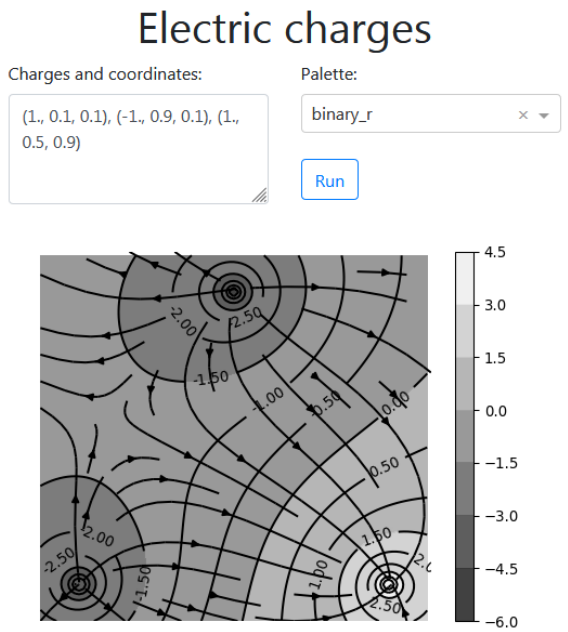

Fig. 6. Visualization of the electric field of the of the electric charges

Individual versions of SPOC can be are created by specifying user groups in TOC description roles fields (see Section VI).

The server web application is implemented with Django framework, in addition to the above-mentioned functionality; it supports publishing and execution of applets. Applets are small interactive web applications written in Python and built into SOOCs and SPOCs. The server application provides an automated creation of user interface based on input and output parameters descriptions. Fig. 6 presents the user interface of applet for visualization of electric field of the electric charges, coordinates and charges are provided by the user.

IX. Conclusion

The article proposes two approaches for quick and easy creation and maintenance of SOOC and SPOC. The first approach generates a static SOOC site with the uniform design of each page. Navigation structure is generated from the table of contents. When a textbook is assembled, its content is checked. The generated static site is a one-page client web application. It can be deployed both on a local computer and on a web server.

The second approach uses server web application. It allows easy content and users management, creation of individual versions of SOOC and SPOC. In addition, the server application supports applets – interactive user Python applications. Applets user interface is generated based on input and output parameters descriptions.

Both approaches use the same description of the textbook table of contents in the form of an Excel spreadsheet and you can quickly make changes and additions to the content of electronic textbooks.

References

[1] J.-C. Pomerol, Y. Epelboin, C. Thoury MOOCs. Design, Use and Business Model. London: ISTE Ltd., JohnWiley & Sons, 2015.

[2] F. Fox. From MOOCs to SPOCs. Communications of ACM, vol.56, #12, p.38-40, 2013.

[3] J.M. A. Gilbert. edX E-Learning Course Development. Birmingham: PACT Publishing, 2015.

[4] S. S. Nash, E. Rice. Moodle 3 E-Learning Course Development, Fourth Edition. Birmingham. GB: Pack Publishing, 2018.

[5] W. Rice. Blackboard Essentials for Teachers. Birmingham, GB: Pack Publishing, 2012.

[6] A. Berg, M. Korcuska. Sakai Courseware Management. Birmingham, GB: Pack Publishing, 2009.

[7] S. Goldstein. CMS Made Simple Development Cookbook. Birmingham, UK: PACT Publishing., 2009.

[8] B. Messenlehner, J. Coleman. Building Web Apps with WordPress. Sebastopol, CA, USA, O'Reilly Media, 2020.

[9] N. Abbott, R. Jones, M. Glaman, C. Chumley. Drupal 8: Enterprise Web Development. Birmingham: PACT Publishing, 2015.

[10] B. Harwani. Foundations of Joomla. Apress, N.-Y., USA, 2016.

[11] Nigel George. Beginning Django CMS. Apress, N.-Y., USA, 2015.

[12] K. Siegel. TechSmith Camtasia Studio 9: The Essentials, Riva, MD, USA, 2016.

[13] iSpring Suite 9.7. eLearning Authoring Supercharge. [Electronic resource] – Access mode: https://www.ispringsolutions.com/ispring-suite (date of access: November 11, 2019).

[14] W. R. Jiang, J. H. Yan. "Implementation of Static Web-Pages Generator Using JavaScript," Applied Mechanics and Materials, vol. 39, 2011, pp. 588-591.

[15] R. Camden., B. Rinaldi. Working with Static Sites: Bringing the Power of Simplicity to Modern Sites. Boston, USA: O'REILLY, 2017.

[16] StaticGen. A List of Static Site Generators for JAMstack Sites. [Electronic resource] – Access mode: https://www.staticgen.com/ (date of access: November 12, 2019).

[17] Headless CMS. A List of Content Management Systems for JAMstack Sites. [Electronic resource] – Access mode: https://headlesscms.org/ (date of access: November 12, 2019).

[18] K. Ismail. 15+ Static Site Generators to Complement Your Headless CMS. [Electronic resource] – Access mode: https://www.cmswire.com/digital-experience/15-static-site-generators-to-complement-your-headless-cms/ (date of access: November 12, 2019).

[19] The Headless CMS – Content Management Future. [Electronic resource] – Access mode: https://habr.com/ru/post/439782/ (date of access: November 12, 2019) (in Russian).

[20] A. Tikhonov. Static Site Generator for Electronic Tutorials. Certificate of State Registration of the Computer Programme No. 2017663324. Application No. 2017660322. Date of receipt is October 12, 2017. Date of state registration in the Computer Programme Register November 28, 2017 (in Russian).