Indonesian Journal of Electrical Engineering and Computer Science
Vol. 34, No. 2, May 2024, pp. 1315~1323
ISSN: 2502-4752, DOI: 10.11591/ijeecs.v34.i2.pp1315-1323

An effective secondary personalization file system driven by FileForge module

Gaojian Liu, Yufei Hu, Ngai Cheong
Faculty of Applied Sciences, Macao Polytechnic University, Macao, China

Article Info

Article history:

Received Dec 1, 2023
Revised Feb 2, 2024
Accepted Feb 16, 2024

Keywords:

Administrative simplification
Documentation delivery
Efficiency in documentation
Secondary personalization file systems
Technological frameworks

ABSTRACT

Digital service platforms provided by academic support departments in Macao assist academic staff and students in various areas such as registry, student affairs, academic activities, and research. As the number of undergraduate students increases and new departments are established, academic staff often face the challenge of dealing with paperwork that contains similar content but different formats. This situation results in redundancies and a waste of time. This paper presents our endeavors to simplify administrative procedures in higher education by automating restructured documentation and developing secondary file systems. The paper presents two case studies: Scenario One focuses on streamlining the publication system for academic staff who submit papers in different formats. At the same time, Scenario Two aims to simplify the daily paperwork process for academic staff. Both cases involve transforming the distribution of administrative documents, transitioning from a standardized form with guidelines to a customized form with concise tips. This approach allows academic staff to handle only the necessary information, which may not be available in the database or requires verification. The case studies serve to demonstrate the effectiveness of this administrative simplification.

Corresponding Author:

Ngai Cheong
Faculty of Applied Sciences, Macao Polytechnic University
Macao, China
Email: ncheong@mpu.edu.mo

1. INTRODUCTION

Kennedy *et al.* [1], [2] said an academic staff usually needs to handle a variety of standardized documents and prepare for several academic summarized reports/feedbacks/statistics each semester. As a result, a series of similar/duplicate documents are required, which only have a little difference across departments [3], [4]. Lasley and Haberman [5] introduced these kinds of documents formed with data typically sourced from specific distributed/centralized databases or file systems in the different departments. Kabakus *et al.* [6] proposed that sourcing data, seemingly, is a simple and repetitive task. It must consume a considerable amount of manpower. Heck *et al.* [7] and Yidana *et al.* [8] pointed out that whenever a large demand for integrating information across systems and formats appears, the task is more complicated to solve. For example, holding conferences, merging across sectors, or customizing documents has a huge documentary workload. Alvarez-Sández *et al.* [9] and Rizos *et al.* [10] suggested that meeting these challenges, we should rapidly build up an efficient and intelligent tool of document automation is an urgent need.

The concern of automatic document generation is raised across the field of higher education. Vito D'Orazio *et al.* [11] introduced a method for automatically classifying document based using support vector machines, realizing the automatic classification of different documents. Lin *et al.* [12] proposed a method for automatically generating satellite constellation health status management documents, facilitating knowledge reuse and generic content generation in document drafting, and establishing a standardized and effective document preparation process. Sun *et al.* [13] proposed an automatic test document generation method based on knowledge extraction, addressing a series of problems caused by flexible and varied test records, conducive to the effective use of test results. Wang *et al.* [14] designed an automatic document generator based on module-driven design, allowing customization according to user needs. Miao *et al.* [15] proposed a new method that comprehensively considers the total number of label references in the tag and the total number of labels in the configuration file, designed to calculate the tag weight in user configuration files and resource configuration files; a resource correlation calculation method based on cosine similarity calculation and comprehensive matching of the number of tags was also introduced. Qu *et al.* [16] analyzed the characteristics and basic data types of data summary documents, proposing a document automatic generation model. This model defined necessary operations and document templates during the document generation process and transformed document templates into specific date data summary documents through iterative calculations. Xingqiang and Na [17] designed an automatic generation method for electronic medical record-sharing documents, encompassing diagnosis and treatment activities, events, document retrieval services, data services, document synthesis services, and shared document databases in six steps. Moreno *et al.* [18] an approach for automatically generating release notes, which can avoid missing important information when filling in manually, improving work efficiency and saving time. Klahold *et al.* [19] and Xiaobo *et al.* [20] used C#.NET to automate Word document operations, including creating, opening, saving, closing, inserting, and deleting functions. Ge and Wu [21], [22] introduced the SGMG model, based on BERT and multi-head graph attention networks, which effectively extracts text thematic structures. The model can generate texts similar to manual templates, outperforming traditional methods by 4.3% on the ROUGE-L evaluation metric [23], [24]. Pisaneschi *et al.* [25] proposed an innovative document layout analysis method, improving document object detection by training with synthetic pages and neural networks. Starting from 218 single-column articles, this method achieved a remarkable 49% improvement in detection performance.

Many researchers have made different achievements in the automatic generation of various documents. Their research and methods involve processing standardized documents, obtaining data, integrating information, automating document generation, and layout analysis. However, they do not do much research on academic documents such as journal submissions and course outlines. The studies provided a foundation and insight into the progress of automatic document generation. It can simultaneously address the automatic generation of multiple formats, such as MS Word and MS Excel. The greater the number of document recipients and the greater the variety of documents, the more efficient the overall system becomes. This paper explores the intricacies of document standardization processes and proposes a novel framework for generating secondary files. Simplifying an action (understood as a process) involves reducing the number of intermediary steps [26].

2. MODEL CONSTRUCTION AND ALGORITHM RESEARCH

In traditional manuscript submission systems, authors must submit their works to the journal editors in a predetermined format. Upon receiving these submissions, the journal editors often manually undertake extensive editing and formatting adjustments to ensure conformity with journal standards, as shown in Figure 1. This process involves multiple file formats and intricate typesetting rules, significantly consuming manpower and time. Figure 1 illustrates the complex version of selection in a submission system. We need to develop a new type of file-processing system to address this issue. In this system, an author only needs to submit basic information like titles, author details, and abstracts and upload their articles. The system can automatically parse this information, format it, and then integrate it according to the journal format. The whole process not only significantly reduces the workload of journal editors but also allows the authors to focus more on the content of the articles, ensuring the efficiency and professionalism of submitted manuscripts.

The general execution flow of this system is shown in Figure 2. Firstly, the user logs in to the system and selects the supported journals from the system library. Then, the user fills in the author, abstract, keywords, and other necessary information using the provided template. The user uploads the file to the system. Simultaneously, the system automatically populates the user's saved information from the system database according to the journal's specific requirements. Finally, the system seamlessly integrates all the materials into a paper submission document that adheres to the journal's format requirements. This process significantly saves authors' time while submitting the final draft of a journal article, relieving them from concerns about paper formatting and other requirements.

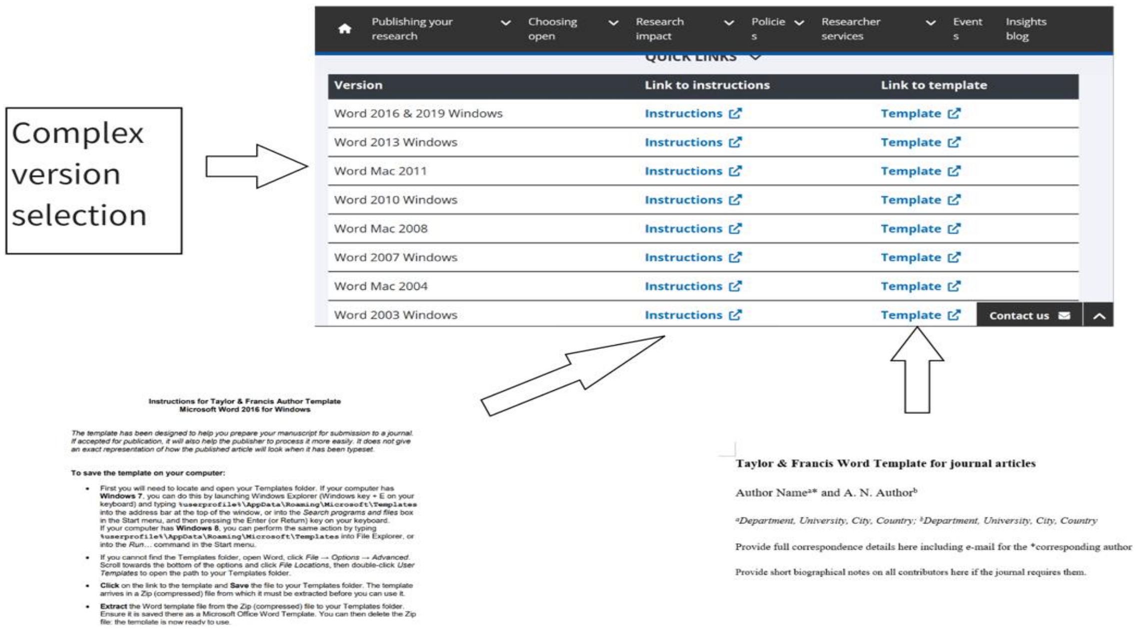


Figure 1. The complex version of selection in a submission system

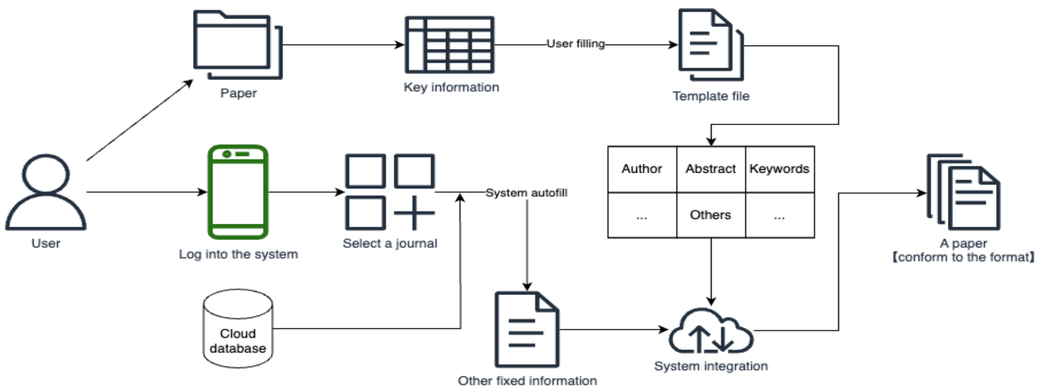


Figure 2. The general execution process of the paper submission system

The framework and methods are personalized in practical applications. There is often a need to extract data from a document with a fixed format and adjust with minimal change. Then, the extracted dataset is inserted into another type of specific document. However, many of the datasets that need to be changed can be found in another file. The proposed method for universal automated file processing is based on FileForge software. The system framework consists of three parts: editing and selecting template files, data sources, and personalized file generation and output, as shown in Figure 3.

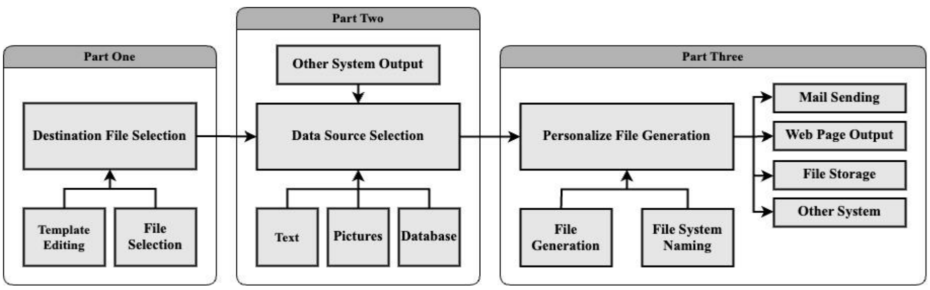


Figure 3. System framework diagram

2.1. Editing and selection of template files

The system's first part involves editing and selecting template files, as shown in part 1 of Figure 2. Data variables are replaced and identified by adding placeholders in the template file. The choice and editing of the template file are crucial steps in the automated document generation process. An effective template file should have a clear structure, enabling users to understand the layout and format of the document.

In our approach, we added specific placeholders into the template file to pinpoint the location of data variables that need replacing. Placeholders are pre-defined special characters or strings representing the data to be inserted. In our system, the placeholders might look like these char strings: "{name1, name2…, }". The "name1", "name2",.. are represented as the names of the specific data variables we wish to insert, which offers us great flexibility to insert any type of data anywhere as needed. In practice, multiple template files might be available for selection to meet the different requirements of journals and different types of documents. For instance, one template might generate reports, while another might be for crafting resumes. Users can select the template file that best suits their needs.

2.2. Data sources

The appropriate selection of data sources is crucial in automated file generation. As shown in part 2 of Figure 3, we categorize data sources into fixed and dynamic datasets. Fixed dataset refers to traditional data types that do not change over time and remain stable in most situations. Fixed datasets include but aren't limited to data in pure Text, Pictures, and Databases. In contrast, the dynamic dataset is more complex as it might change over time or under certain conditions. Such datasets often originate from real-time outputs of other systems, like real-time categorization in personalized learning systems based on the progress and abilities of students. By combining and utilizing these two data sources, we can ensure the accuracy and timeliness of file generation, which better suits various application needs.

2.3. Personalized file generation and distribution

This document generation and distribution system is essentially a personalized document writing system based on the database. As shown in part 3 of Figure 3, this section delves into the final stage of automatic document generation. It consists of three core strategies: 1. file generation mechanism, 2. file naming system, and 3. file output and distribution strategy.

2.3.1. File generation mechanism

The file generation process primarily depends on placeholder information from the template file. Firstly, we parse the template file to identify all placeholders, then use placeholders as query keywords to search the data sources. If a matching dataset is found in the sources, we collect this data, readying it for insertion into the template file in the next step. We can produce new personalized files by replacing the placeholders in the template.

2.3.2. File naming system

The system provides flexible file naming options, allowing users to tailor naming rules according to their needs. The file naming format can include different information, i.e., names, locations, dates, and content descriptions. This naming approach enhances the system's personalization and makes files more identifiable and searchable. Thus, it may directly approach the organizational and searching needs of users.

2.3.3. File output and distribution strategy

After completing the file generation and naming process, the next task of the system is file output and professional distribution (detailed in part 3 of Figure 3). In this phase, we focus on target-oriented file distribution rather than mere output, ensuring every file finds its rightful place based on pre-determined strategy and needs. Ultimately, the system offers multiple distribution options, ranging from direct integration with email systems for automatic dispatch to specific recipients, embedding files in web pages for real-time information access by users or visitors, or saving files to specific directories for long-term storage. Additionally, considering cross-system operation needs, the generated files can be configured as inputs for other systems, facilitating seamless integration with other platforms or apps. This strategy optimizes file output efficiency and ensures accurate and secure information conveyance.

2.4. Function model of personalized document generation

We introduce a function-based model to encapsulate the operations within our system framework mathematically. This model can abstract the process of personalized document generation. The mathematical representation of the automated file generation process is in (1). This formalism allows us to efficiently map out the entire process from template selection to file distribution, thus providing a structured approach to automate document generation. By leveraging this model, our system can reduce manual data handling, minimize errors, and streamline the creation of personalized documents.

$$F(s, f, d, u, r) = O(G(P(T(s), D(f, d)), N(u)), r) \quad (1)$$

In the above model, *T(s)* represents the function of selecting an appropriate template file based on user specifications *s*. *D(f, d)* denotes the data extraction function, which pulls necessary information from fixed *f* and dynamic *d* data sources. Once the relevant dataset is extracted, *P(t, x)* is employed as the placeholder substitution function, replacing placeholders in the template *t* with actual data *x*. Subsequently, *N(u)* defines the file naming function that adheres to user-defined naming conventions *u*. *G(p, n)* represents the file generation function that assembles the personalized document by integrating the data-substituted template *p* and the file name *n*. Finally, *O(g, r)* encapsulates the output and distribution strategy, managing the delivery and storage of the generated file g according to predefined requirements *r*.

3. METHOD PROPOSAL FOR A SPECIFIC APPLICATION SCENARIO CASE AND RESULT

In Scenario Two, we present a personalized file generation for a course syllabus, which was chosen as an application scenario case. The teaching syllabus for each semester is a Word file with a relatively fixed format that seldom changes, and most of the dataset that needs to be modified in this file can be found in other Excel spreadsheets. However, searching manually for necessary datasets within these Excel files by the instructors is inefficient and time-consuming. The personalized file processing method based on FileForge software offers a good solution for this scenario. Figure 4 shows the regular execution process of documents, such as the course syllabus in the system, showing a combination of cooperation and convenience.

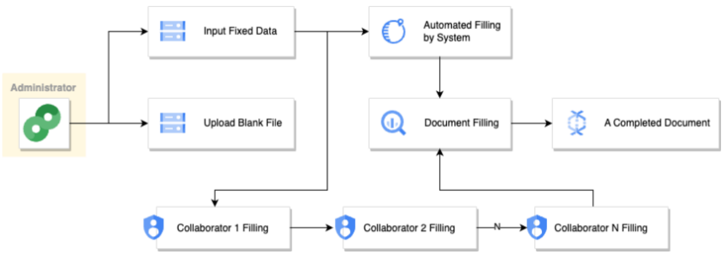


Figure 4. The general execution process of the paper submission system

3.1. Editing and selection of template files

In the case of editing a course syllabus, the template consists of two parts that need to be filled in. The system fills in one, and another is filled in by the instructors. Figure 5 shows an example of the process of editing a course syllabus. The system will retrieve the database according to the selected course and automatically fill in the existing information in the form. For instance, the "Pre-requisite(s)" content can be retrieved from other files, eliminating the need for manual searches within these files. Using placeholders combined with keywords, the system can search and locate the required information within files. Thus, the information for "Pre-requisite(s)" is represented as "{Discrete Mathematics. Pre-requisite(s)}". Additionally, sections like "Medium of Instruction" and "Lecture Hours" are filled in by the system, and the only thing we need to add are corresponding placeholders if we want to edit the template. And other parts that need to be filled in by the Instructors. In the figure of the example, the content for sections like "Content" requires manual input by Instructors. Therefore, there is no need for extensive modifications when editing the template for these sections; simply leave the necessary space for Instructors to fill in.

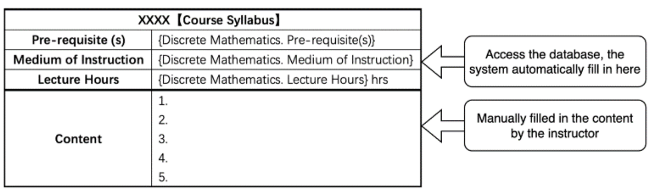


Figure 5. An example of the process of editing a course syllabus

3.2. Data source retrieval

The primary data source in the above course syllabus is an Excel spreadsheet, as shown in Figure 6. The first column of the table lists the names of the courses, while the first row provides the respective dataset of each course. As mentioned earlier, we have a placeholder with a keyword of "{Discrete Mathematics. Pre-requisite(s)}". From this table, we can find the required data filled in "Essential Computer Mathematical." When implementing this in code, we utilized the pandas libraries for development. Firstly, we identified all Excel files within the specified path. Subsequently, based on the keywords inside the placeholders, we search for the corresponding dataset within the Excel sheets.

Learning Module	Class Code	Medium of Instruction	Credit	Lecture Hours	Lecture Hours	Total Hours	Pre-requisite(s)
Discrete Mathematics	MATH121	English	3	45	0	45	Essential Computer Mathematics
xxx	xxx	xxx	xxx	xxx	xxx	xxx	xxx

Instructor	Text Book	Reference Book
Dr [redacted]	Kolman, B., Busby, R. C., and Ross, S. C. (2008). *Discrete*	Rosen, K. H. (1998). Discrete Mathematics and Its Applications. Singapo
Wong [redacted]	Routing Protocols and Concepts, CCNA Exploration Companio	LAN Switching and Wireless, CCNA Exploration Companion
TBA	Management Information Systems	Artificial Intelligence: A Modern Approach

Figure 6. The data source generated by the file

3.3. Generation of personalized files

After completing the above steps, after the personalized file is generated by the system, the teacher can further complete and fill in the details to form the final comprehensive course outline. Figure 7 depicts the Word document generated after the system processing and also shows the part of the document completed by the instructor. Different users can use the generation template of the same "course outline" to automatically generate the same content in the first half and fill in the "personalized" course content themselves. When the system is generated, we use a naming system based on date, major, and course name. For the code implementation of file generation as shown in Figure 8, the corresponding pseudocode is shown in Figure 8(a). The result is shown in Figure 8(b).

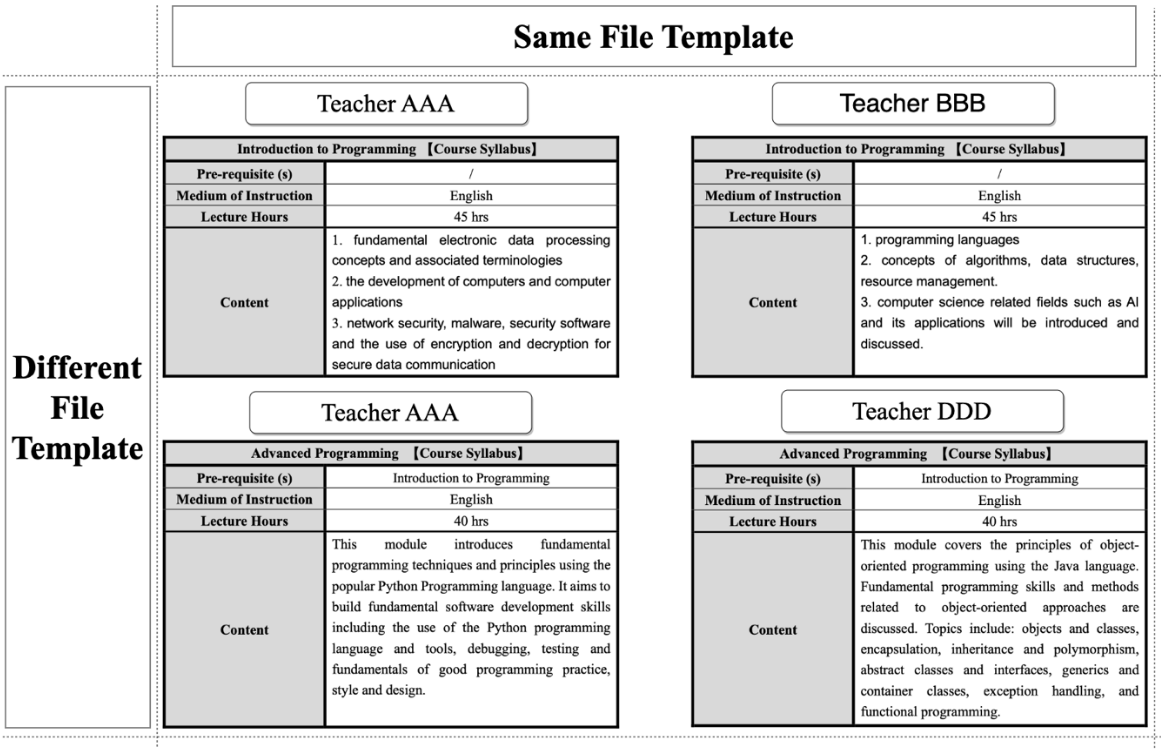
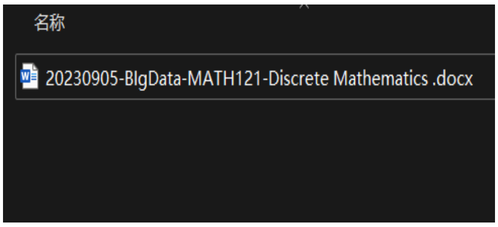


Figure 7. Examples of results of document generation by the system




```
DEFINE GenerateFile(TemplateFile, DataSource):
 FILL systemPart of TemplateFile using DataSource
 RETURN filled file

FOR EACH item in teacherPart:
 TEACHER manually fills content
```

(a) (b)

Figure 8. The code implementation (a) and result (b) of file generation by the system

4. BEFORE AND AFTER USING THE SYSTEM: EFFICIENCY COMPARISON

In the context of journal submission, it is customary to make modifications based on the specific requirements outlined by the respective journal. This often involves filling in a significant amount of pre-existing information. Our proposed design aims to streamline this process by eliminating the need for repetitive steps. Instead, we suggest importing the author's existing information directly into the database and automatically adjusting the format as necessary. Essentially, our design relies on the system's ability to handle most of the modification steps. Table 1 illustrates the changes in the relevant steps of the traditional journal submission process before and after implementing our system.

Table 1. The changes in the main submission processes of traditional journals

Step	Before using the system	After using the system
1. Find submission websites	Need	No need
2. The template of the journal to be submitted is required	Need	No need
3. Fill in the content	Need	Only a part
4. Modify text formatting	Need	No need

The use of the system significantly reduces the manual steps required and improves efficiency. Of course, the course outline system is also based on such ideas. This is particularly evident in preparing course syllabi, where similar steps are greatly reduced. For instance, known content such as course information and introduction can be automatically filled in the syllabus template after the user selects the target course. Additionally, the system supports online collaborative document filling, reducing the need for manual input. This saves time for users when creating the file.

5. CONCLUSION

This paper demonstrates a secondary personalization file systems framework to mitigate inefficiencies inherent in routine document management. Through the FileForge software suite, the system may show a universal file processing paradigm, which is demonstrated through the prism of the journal publishing system and course syllabus customization. It can offer wide-ranging applicability in numerous sectors that require regimented document structures and recurrent processing activities. By automating and personalizing the creation and manipulation of documents, this method substantially abrogates the monotonous burden traditionally associated with such tasks. A two-case study is to demonstrate the efficiency of this administrative simplification. Filling in reference materials will be solved by machine learning in the future. At the same time, there will also be some future considerations for security and other aspects of this framework.

ACKNOWLEDGEMENTS

The support provided by Macao Polytechnic University (MPU Grant no: RP/FCA-06/2022) enabled us to conduct data collection, analysis, and interpretation, as well as cover expenses related to research materials and participant recruitment. MPU investments in our work have significantly contributed to the quality and impact of our research findings.

REFERENCES

[1] D. Kennedy, *Academic duty*, vol. 35, no. 07. Cambridge: Harvard University Press, 1998. doi: 10.5860/choice.35-3991.

[2] N. Kameo and J. Whalen, "Organizing documents: standard forms, person production and organizational action," *Qualitative Sociology*, vol. 38, no. 2, pp. 205–229, Jun. 2015, doi: 10.1007/s11133-015-9302-7.

[3] J. Carlson, M. Fosmire, C. C. Miller, and M. S. Nelson, "Determining data information literacy needs: a study of students and research faculty," *portal: Libraries and the Academy*, vol. 11, no. 2, pp. 629–657, Apr. 2011, doi: 10.1353/pla.2011.0022.

[4] F. Radlinski, P. N. Bennett, B. Carterette, and T. Joachims, "Redundancy, diversity and interdependent document relevance," *ACM SIGIR Forum*, vol. 43, no. 2, pp. 46–52, Dec. 2009, doi: 10.1145/1670564.1670572.

[5] T. J. Lasley and M. Haberman, "How do university administrators evaluate education seans?," *Journal of Teacher Education*, vol. 38, no. 5, pp. 13–16, Sep. 1987, doi: 10.1177/002248718703800504.

[6] A. K. Kabakus, E. Bahcekapili, and A. Ayaz, "The effect of digital literacy on technology acceptance: an evaluation on administrative staff in higher education," *Journal of Information Science*, p. 016555152311600, Mar. 2023, doi: 10.1177/01655515231160028.

[7] R. H. Heck, L. K. Johnsrud, and V. J. Rosser, "Administrative effectiveness in higher education: Improving assessment procedures," *Research in Higher Education*, vol. 41, no. 6, pp. 663–684, 2000, doi: 10.1023/A:1007096803784.

[8] P. Y., J. A.A., H. A.G., and G. M.B., "Evaluation of administrative support services for quality assurance in higher education: empirical review," *Journal of Advanced Research and Multidisciplinary Studies*, vol. 3, no. 1, pp. 87–104, May 2023, doi: 10.52589/JARMS-NTBUSQKI.

[9] D. Alvarez-Sández, K. Velázquez-Victorica, A. Mungaray-Moctezuma, and A. López-Guerrero, "Administrative processes efficiency measurement in higher education institutions: a scoping review," *Education Sciences*, vol. 13, no. 9, p. 855, Aug. 2023, doi: 10.3390/educsci13090855.

[10] S. Rizos, E. Sfakianaki, and A. Kakouris, "Quality of administrative services in higher education," *European Journal of Educational Management*, vol. 5, no. 2, pp. 115–128, Dec. 2022, doi: 10.12973/eujem.5.2.115.

[11] V. D'Orazio, S. T. Landis, G. Palmer, and P. Schrodt, "Separating the Wheat from the Chaff: Applications of Automated Document Classification Using Support Vector Machines," *Political Analysis*, vol. 22, no. 2, pp. 224-242, 2017, doi: 10.1093/pan/mpt030.

[12] M. Lin *et al.*, "Automatic generation method for health management document of satellite constellation," *Computer Systems & Applications*, vol. 30, no. 12, pp. 139–146, 2021, doi: 10.15888/j.cnki.csa.008207.

[13] X. Sun, N. Li, and L. Zhang, "Automatic Generation of Test Documents Based on Knowledge Extraction," in *Proceedings of the 2022 11th International Conference on Software and Computer Applications, Melaka, Malaysia, 2022: Association for Computing Machinery*, pp. 15–20, doi: 10.1145/3524304.3524307.

[14] Chao Wang, Hong Li, Zhigang Gao, Min Yao, and Yuhao Yang, "An automatic documentation generator based on model-driven techniques," in *2010 2nd International Conference on Computer Engineering and Technology*, IEEE, 2010, pp. V4-175-V4-179. doi: 10.1109/ICCET.2010.5485654.

[15] H. Miao, D. Muhong, and C. Hao, "A Personalized Retrieval Method Based on User Profiles," *Application Research of Computers*, vol. 33, no. 2, p. 417, 2016, doi: 10.3969/j.issn.1001-3695.2016.02.022.

[16] M. C. Qu, M. H. Liao, X. H. Wu, and Z. Q. Liu, "Construction of an automatic document generation model and its application," *Jisuanji Jicheng Zhizao Xitong/Computer Integrated Manufacturing Systems, CIMS*, vol. 14, no. 7, pp. 1297–1305, 2008, doi: 10.13196/j.cims.2008.07.51.qumch.008.

[17] W. Xingqiang and M. Na, "The automatic generation method of shared document of electronic medical records," *China Digital Medicine*, vol. 14, no. 8, pp. 103–105, 2019, doi: 10.3969/j.issn.1673-7571.2019.08.034.

[18] L. Moreno, G. Bavota, M. Di Penta, R. Oliveto, A. Marcus, and G. Canfora, "ARENA: an approach for the automated generation of release notes," *IEEE Transactions on Software Engineering*, vol. 43, no. 2, pp. 106–127, 2017, doi: 10.1109/TSE.2016.2591536.

[19] A. Klahold, J. Zenkert, and M. Fathi, "Computer aided writing — a prototype of an intelligent word processing system," in *2017 International Conference on Computational Science and Computational Intelligence (CSCI)*, IEEE, Dec. 2017, pp. 234–239. doi: 10.1109/CSCI.2017.37.

[20] C. Xiaobo, W. Zhixue, and W. Lei, "The application of bookmarks in realizing Word automation operations," *Information Technology*, vol. 37, no. 8, pp. 148-150, 2013, doi: 10.3969/j.issn.1009-2552.2013.08.043.

[21] F. Ge and N. Wu, "A platform for automatically producing Word document based on multiple techniques," *Dianzi Keji Daxue Xuebao/Journal of the University of Electronic Science and Technology of China*, vol. 36, no. 2, pp. 263–266, 2007, doi: 10.3969/j.issn.1001-0548.2007.02.032.

[22] Z. Zhang, N. Li, and Y. Tian, "Stream document structure recognition based on bidirectional LSTM network," *Jisuanji Gongcheng/Computer Engineering*, vol. 46, no. 1, 2020, doi: 10.19678/j.issn.1000-3428.0053702.

[23] F. Chao and Q. E, "Analysis of the processing Excel electronic form with the latest Python," *Computer Age*, vol. 000, no. 4, pp. 69–71, 2021, doi: 10.16644/j.cnki.cn33-1094/tp.2021.04.019.

[24] J. Liu and N. Liu, "Automatic generation of semi-structured texts for bidding documents," *Jisuanji Gongcheng/Computer Engineering*, vol. 49, no. 3, pp. 67–72, 2023, doi: 10.19678/j.issn.1000-3428.0064063.

[25] L. Pisaneschi, A. Gemelli, and S. Marinai, "Automatic generation of scientific papers for data augmentation in document layout analysis," *Pattern Recognition Letters*, vol. 167, pp. 38–44, 2023, doi: 10.1016/j.patrec.2023.01.018.

[26] N. Cheong, "Knowledge-based Recommender System of Conceptual Learning in Science," in *Proceedings of the 2021 5th International Conference on Education and E-Learning, Virtual Event, Japan, 2022: Association for Computing Machinery*, pp. 9–14, doi: 10.1145/3502434.3502443.

BIOGRAPHIES OF AUTHORS

Gaojian Liu born in 1997, received the B.Eng. degree from Beijing Institute of Technology, Zhuhai. He is currently pursuing the M.Eng. degree with the School of Faculty of Applied Sciences, Macao Polytechnic University, Macao. His current research interests include Natural Language Processing and Personalized learning. He can be contacted at email: p2213090@mpu.edu.mo.

Yufei Hu born in 2000, received the B.Eng. degree from Applied Technology College, Soochow University. He is currently pursuing the M.Eng. degree with the School of Faculty of Applied Sciences, Macao Polytechnic University, Macao, China. His current research interests include Machine Learning and IOT. He can be contacted at email: p2311715@mpu.edu.mo.

Ngai Cheong received his bachelor and master degrees in Electronics Engineering from South China University of Science, China and Electronics Engineering, South China Normal University, China in 1986 and 1989, respectively. He received his PhD in Electrical and Electronics Engineering from University of Macau in 2001. He is currently a professor of Faculty of Applied Sciences, Macao Polytechnic University, Macau, China. His research interests include Knowledge-based Recommender System in Conceptual Learning, Computer Aided Design and Theory of E-Government. He is a senior member of the IEEE, member of IET and member of MAPST (Macao Association for Promotion of Science and Technology). He can be contacted at email: ncheong@mpu.edu.mo.