Program implementation of methods for analysis and verification of technical reports

Dmytro Sydorko1, Valerii Hlukhov2

1. Computers Department, Lviv Polytechnic National University, UKRAINE, Lviv, st. Stepana Bandera, 12, E-mail: DmytroSydorko@gmail.com

2. Computers Department, Lviv Polytechnic National University, UKRAINE, Lviv, st. Stepana Bandera, 12, E-mail: Valerii.S.Hlukhov@lpnu.ua

Кеуwords Text analysis; Document analysis; Standard control of documents; MS Word; Docx.

This paper presents a .NET-based solution for analyzing and verifying technical reports in docx format, ensuring compliance with standards. Using WPF and DocumentFormat.OpenXml, it checks formatting, style consistency, and page parameters. Administrators can configure style sequences, and discrepancies are logged for review, streamlining documentation processes and enhancing report quality.

Introduction

Educational, scientific, and design institutions must meet strict document design standards, requiring proper formatting, style consistency, and adherence to predefined structures. Non-compliance complicates information processing and hinders documentation management. Automating regulatory control of reports and technical documents ensures high-quality outputs and efficient management.

This study extends a prior approach to regulatory control by developing software tools for document analysis and verification. Built with .NET, WPF, and DocumentFormat.OpenXml, the solution checks style consistency, page conformance (A4, Letter), and detects empty lines. It also allows administrators to adjust style sequences using “before” and “after” parameters, ensuring structural correctness.

An optimized version of the program reduces size by over 50% while retaining full functionality, improving performance and usability. This approach minimizes resource consumption and enhances inspection efficiency. The solution promotes quality and standardization in technical documentation, supporting advancements in computer engineering and streamlining organizational processes.

Research of references

Creating Creating text documents that adhere to formatting and style standards is essential in education and project work. Failing to meet these requirements complicates document analysis and usage. This article builds on the approach presented in [1] to enhance the regulatory control of documents through improved software tools for analysis.

A new model ensures content consistency by treating each paragraph or sentence as a node, simplifying structure validation [2]. Automatic classification of digital documents further streamlines office file checks [3]. Document classification often requires advanced object detection techniques, and studies have explored effective loss functions for text processing [4].

Additionally, the ALDOCX framework was enhanced to analyze potentially corrupted docx files with meta-functions [5]. The Text Guide method offers optimized text summarization without significant computational overhead [6]. To better manage XML documents, comparisons of storage solutions like Berkeley DB XML and eXist have been conducted [7].

Security is also critical. Research highlights vulnerabilities in OOXML structures [8], while blockchain technologies ensure document integrity [9]. Hybrid corporate solutions provide further protection by tracking all database changes step-by-step [10].

Setting of the task

The task of this study is to improve the software for increasing the efficiency of management and analysis of text documentation in docx format according to the established requirements.

This work focuses on optimizing the size of the previous version [1] of the program to minimize its volume without losing functionality. In addition, the system will automatically check the sequence of styles in documents and allow administrators to make changes to this sequence. The software should also check whether the page format conforms to A4 or Letter standards, whether the page orientation is correct, as well as meets the requirements for the size of the margins, footers and the absence of extra blank lines, providing detailed reports on the discrepancies found. And also the possibility of tracking all changes of already assigned styles directly in the text.

Research results

Based on the studies, a block diagram of algorithms was developed to ensure compliance with document norms. The paragraph verification algorithm (Fig. 1) analyzes paragraphs in the document’s main body, checking their structure and style.

The algorithm then checks if each paragraph matches the required style, logging errors for discrepancies. A final check detects unauthorized modifications. Each paragraph is processed sequentially, ensuring structural accuracy and formatting compliance throughout the document.



Fig. 1. Algorithm for analysis of the main part of the document (Budy).

Implementation and experimental verification of the document analysis program

In the new version of the program, the interface retains the general structure of the previous version, which provides convenience for regular users. However, minor improvements have been made: the password change and settings buttons have received more intuitive names for better understanding (Fig. 2).



Fig. 2. Algorithm for analyzing the correct order of styles.

A new "Strict styles check" checkbox has also been added, which activates strict styles by highlighting the modification to check specific words or groups. If a deviation is detected, the system generates a warning with the text "Edited style" (Fig. 3).



Fig. 3. Highlighting errors in the ANALISED file.

A check for empty lines has been added to the function program. If such a line is found, a message with the text "Empty line" appears in the report (Fig. 3).

The administrator also got the opportunity to adjust the order of styles in the document using the “before” and “after” parameters (Fig. 4). These options allow you to define multiple styles for each position by separating them with commas. If a discrepancy is detected, a message with the name "Invalid order" will appear (Fig. 3).

Administrators can set rules for checking page size, orientation, and indents both for the page itself and for footers (Fig. 4). All inconsistencies are recorded in the report "Report.txt" (Fig. 5), where the type and location of each error is detailed.

One of the priority tasks was to reduce the volume of the program without losing its functionality. Thanks to code optimization and the use of resource compression, it was possible to reduce the size of the previous version of the program by more than 50% — from 337 MB to 157 MB.



Fig. 4. Customizing Styles for Validation.



Fig. 5. The contents of the Report.txt file with detected inconsistencies.

To verify the new functions, a series of tests was conducted using documents of various formats and volumes. The results showed high efficiency in detecting formatting and style violations. All detected errors were successfully recorded in the report, and the interface showed convenience and clarity of use for both users and administrators.

The program is implemented in the C# language using the .NET framework and the DocumentFormat.OpenXml, WPF libraries, in the Jetbrains Reader development environment.

For the correct launch and operation of the program on a computer with a pre-installed Windows system, it is necessary to install only the archive of the program, which contains all the necessary dependencies.

Setting styles

The program allows users to create custom style sets and protect them from unauthorized changes. It has been tested for controlling Bachelor’s theses and articles for the ACPS journal, using ACPS-specific styles.

Changing style settings requires administrator access. After logging in with the default password “admin,” administrators gain two additional options: changing the password and configuring style criteria. A modal window allows editing criteria, adding new styles, or defining keywords that determine valid styles.

If changes are canceled, they are discarded. Upon saving, the operation status is displayed, and the settings are stored in the styleSettings.ini file..

Time characteristics of the program

The test results showed the high speed of the program when analyzing files. In addition, analysis of the 4.78-megabyte note to the Bachelor qualification work (Fig. 6) took only 1.5 seconds, which significantly reduced the time required to manually check document styles.



Fig. 6. Time consumption in the analysis of the statistical note for the Bachelor's qualification thesis.

Conclusion

This paper presents the development of software for the analysis and regulatory control of technical reports in docx format, ensuring compliance with established standards. The solution, built using .NET, WPF, and DocumentFormat.OpenXml, verifies page formatting, style consistency, empty lines, and adherence to page parameters (A4, Letter). Administrators can configure style sequences using “before” and “after” parameters, ensuring structural correctness.

A key achievement is the program’s size optimization, reducing it from 337 MB to 157 MB—over 50%—without sacrificing functionality. The software logs all detected inconsistencies for further analysis, streamlining documentation control, enhancing report quality, and promoting standardization within organizations.

The optimized program analyzes documents up to 100 pages or 5 MB in approximately 1.5 seconds, significantly improving efficiency. This automated approach to documentation control, particularly for bachelor and master’s qualification reports, supports computer engineering advancements and optimizes organizational processes by reducing time and resource costs.

References

1. Hlukhov V. S. and Sydorko D. S., "Algorithms and software for verification of scientific and technical text documents" 2023 Applied Aspects of Information Technology 2023; Vol.6 No. 3:304–317, doi: 10.15276/aait.06.2023.21.

2. D. Jung, M. Kim and Y. -S. Cho, "Detecting Documents With Inconsistent Context," in IEEE Access, vol. 10, pp. 98970-98980, 2022, doi: 10.1109/ACCESS.2022.3204151.

3. S. Eken, H. Menhour and K. Köksal, "DoCA: A Content-Based Automatic Classification System Over Digital Documents," in IEEE Access, vol. 7, pp. 97996-98004, 2019, doi: 10.1109/ACCESS.2019.2930339.

4. K. Nguyen, A. Nguyen, N. D. Vo and T. V. Nguyen, "Vietnamese Document Analysis: Dataset, Method and Benchmark Suite," in IEEE Access, vol. 10, pp. 108046-108066, 2022, doi: 10.1109/ACCESS.2022.3211069.

5. H. Lee and H. -W. Lee, "Hidden Message Detection in MS-Word File by Analyzing Abnormal File Structure," 2020 International Conference on Green and Human Information Technology (ICGHIT), Hanoi, Vietnam, 2020, pp. 54-57, doi: 10.1109/ICGHIT49656.2020.00021.

6. W. Xu, Y. Xu, G. Huo, Y. Yang and Y. Jin, "Optimized Dual-mode Security Encryption Chip Design Based on Hash Algorithm," 2022 IEEE 11th International Conference on Communication Systems and Network Technologies (CSNT), Indore, India, 2022, pp. 566-570, doi: 10.1109/CSNT54456.2022.9787655.

7. R. N. Kulkarni, C. Ganesh, D. K. B K, H. B and A. P. Reddy, "Novel Approach to Detect Plagiarism in the Document," 2023 International Conference on Distributed Computing and Electrical Circuits and Electronics (ICDCECE), Ballar, India, 2023, pp. 1-6, doi: 10.1109/ICDCECE57866.2023.10150442.

8. J. Chang and M. Fanguy, "Collab\_doc\_maker: An Automatic Google-Doc-making Tool," 2021 16th International Conference on Computer Science & Education (ICCSE), Lancaster, United Kingdom, 2021, pp. 806-809, doi: 10.1109/ICCSE51940.2021.9569570.

9. B. Li, Y. Chen and L. Zeng, "Kenet:Knowledge-Enhanced DOC-Label Attention Network for Multi-Label Text Classification," ICASSP 2024 - 2024 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), Seoul, Korea, Republic of, 2024, pp. 11961-11965, doi: 10.1109/ICASSP48485.2024.10447643.

10. M. C. Xenya and K. Quist-Aphetsi, "A Cryptographic Technique for Authentication and Validation of Forensic Account Audit Using SHA256," 2019 International Conference on Cyber Security and Internet of Things (ICSIoT), Accra, Ghana, 2019, pp. 11-14, doi: 10.1109/ICSIoT47925.2019.00008.