SYSTEM OF AUTOMATED CHECKING OF TEXTUAL DOCUMENT DESIGN RULES

Maria Zhigalova, Alexander Sukhov

Abstract*: The majority of text documents are made out according to predetermined rules, which regulate requirements to design of the given class of documents. However, the process of paper checking is quite time-consuming and requires from performer of high concentration and appropriate qualification. The paper presents the main methods that minimize the effort required to control the text on the absence of design errors; the analysis of software solutions is given, illustrating the principle of operation of these methods. To automate the verification of formal correctness of the text document the system designed for the compliance assessment of documents in accordance with the parameters set by the user was developed. The domain analysis is carried out; the domain conceptual model is given. The functionality of the system was described and the tools used in its development were presented. However, the verification of a text document is not reduced only to analysis of rules of its design. Also it is required to fulfill verification of document structure: presence of the required sections. The visual domain-specific language, which allows to describe structure of documents, and also connections between different documents, is designed for this purpose. The language has a simple graphical notation; therefore it can be used by as IT-specialists and clients who are not professional programmers. In practice, the developed system can be used to verify compliance with the formal requirements of projects and dissertations, scientific publications, technical documents, etc.*

Keywords*: formatting rules, text document, compliance assessment, design documentation, domain-specific languages.*

ACM Classification Keywords*: I.7 Document and Text Processing: I.7.2 Document Preparation – Format and notation; D.2 Software Engineering: D.2.2 Design Tools and Techniques – Computer-aided software engineering (CASE).*

Introduction

Conformity of the text to the formatting rules is one of the priority requirements for documents reflecting the results of scientific, technical and engineering activities. Uniformity of appearance and structure of text documents is mandatory for any field of knowledge since such standardization allows easier understanding of information provided. In addition to that, strict adherence to standards simplifies the process of storing and processing documents in databases.

Due to the fact that the key role in manual check of formatting rules is played by a specialist for whom increased attentiveness and special qualifications are compulsory, this process can be extremely time-consuming and inefficient in terms of time expenses. For this reason, automation of the text check seems to be a highly relevant issue. A software product that automates the compliance assessment of documents can be used by a wide range of users, including students, teachers, technical writers in organizations, etc.

In a more general case the problem considered involves determining the quality of publications, identification of potential duplication, plagiarism, partial borrowings, classification and clustering of documents, formation of databases and extensive collections of texts. Despite the fact that document check in accordance with formatting rules does not imply detailed text processing, it should be noted that this problem in one way or another is related to the general text analysis and has its own specific features.

The purpose of this work is the development of a system of text document check according to specified formatting rules. Such system will significantly reduce the amount of time and effort required for document analysis in comparison with the manual check.

Related Works

To date there are two basic methods of control of the text on the absence of formatting errors and verification of a document in accordance with certain standards including the use of ready-made formatting templates and software solutions in the process document check.

The existence of a large number of word processors (Apache OpenOffice Writer, Microsoft Word, Pages, etc.) with extensive capabilities of text editing allowing creation of complex documents for various purposes does not negate the fact that the formatting of these documents is still a laborious process. The problem is partially solved by using styles with specified parameters of fonts and paragraphs, but still, there is a high probability of violation of styling when copying text from other documents. That is why it makes sense to apply formatting templates.

One of the means of creating such templates is a markup language DocBook, an application of XML/SGML, which provides a user with a unified set of tags for setting formatting of a text document [Berdachuk, 2015]. An example of a document in a DocBook format is shown in Fig. 1.

In this approach the content of a document is isolated from its style representation. The apparent advantage of DocBook is that a predefined set of tags eliminates the possibility of errors in formatting and allows a large number of users to work with the same text simultaneously.

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE chapter PUBLIC "-//OASIS//DTD DocBook XML V4.4//EN"
"http://www.oasis-open.org/docbook/xml/4.4/docbookx.dtd">
<chapter lang="ru">
  <title>Анализ требований</title>
  <sect1>
    <title>Постановка задачи. Анализ</title>
    <listitem>
      <para>Возможность работы ...</para>
    </listitem>
  </sect1>
</chapter>
```

Figure 1. Document Structure in DocBook

Formatting templates are also proposed by the LaTeX publishing system which provides the capability for automation of the process of text entering and formatting. The content of a LaTeX document, as in the case of DocBook, is represented by structural and semantic markup; the appearance of a document is established by adding a special style file [Lvovsky, 2006] which defines formatting rules, specific to each document type. A sample document in a LaTeX format is shown in Fig. 2.

Despite a vast variety of functional characteristics, it should be mentioned that LaTeX has a number of disadvantages: firstly, working with LaTeX documents requires a special development environment installed on a personal computer, and secondly, the process of creating a document may cause difficulties for users whose primary occupation is not connected with information technologies.

Automation of text checking is implemented in a number of software products, one of which is an intelligent web-based system for spell checking "Orogrammka". Text document check is performed in terms of the norms of spelling, grammar, punctuation and stylistics [Orfogrammka, 2015]. Moreover, it is possible to conduct the compliance assessment of research papers and dissertations in accordance with requirements that are set in standards. The service has an intuitive and simple interface, however, it should be noted that text check is limited to a strictly predefined set of formatting rules (margin sizes, parameters of a title sheet, applications, reference list, etc.) without the possibility of expanding the functionality by the user, for example, checking numbering formats or tables design.

```
\usepackage[koi8-r]{inputenc}
\usepackage[english, russian] {babel}
\inputencoding{koi8-r}
\usepackage[T2A] {fontenc}
\usepackage{letterspace}
\usepackage{a4}
\usepackage{epsfig, graphicx, euscript}
\title{Пример включения файла в документ}
\author{И.И.Иванов}
\date{}
\begin{document}
Документ должен быть отформатирован правильно.
\newline
  \begin{eqnarray}
    s &=& \frac{m_0}{\sqrt{1-\frac{v^2}{c^2}}}
  \end{eqnarray}
\end{document}
```

Figure 2. Document Structure in LaTeX

Another tool for automated formatting rules check was developed on the basis of Volgograd State Technical University [Sokolov, 2013]. This software solution is a Microsoft Word 2007 add-in which allows checking documents and fixing detected errors. In spite of convenience and ease of use, the service has a significant drawback: users whose personal computers are not running Microsoft Office Word are deprived of the opportunity to perform the compliance assessment of text documents.

Requirements for Formatting of a Text Document

As it was previously stated, the necessity of automation of the compliance assessment of documents is caused primarily by a substantial amount of effort required and significant time costs in case of manual text check. The developed system is aimed at automation of text document check according to formatting rules pre-established by the user.

Formatting rules are the settings applied to the content of a document in order to determine its structure and appearance. Requirements for document formatting are specified in normative documents, standards, etc.

Generally, a majority of documents are formatted in conformity with specific requirements that might change over a period of time. Therefore, it was decided to equip the system with the functionality allowing to manually add and modify formatting rules of documents of a particular class.

The analysis of documents demanding certain formatting resulted in identification of a number of essential parameters for assessing the accuracy of text document formatting including:

1) page size (name, width, length);
2) page orientation;
3) margin sizes;
4) maximum and minimum amount of work (in pages);
5) size of headers and footers;
6) page numbering (number position, number formatting);
7) description of style:
    a) style name;
    b) font:
        – name;
        – size;
        – color;
        – italic;
        – bold;
        – underlined;
    c) paragraph:
        – indent sizes (left and right);
        – first line indent size;
        – paragraph spacing (before and after the paragraph);
        – line spacing;
        – alignment.

A conceptual domain model made in the notation of Entity Relationship Diagram is shown in Fig. 3.

Once formatting rules have been entered by a user and a text document has been inputted, the system provides a report containing the results of the document check.

In this regard, main functional requirements for the system involve adding data on formatting rules of documents; editing an information entered; loading document for the check; performing the document check in accordance with the selected rules; generating a report containing the results of the document check.

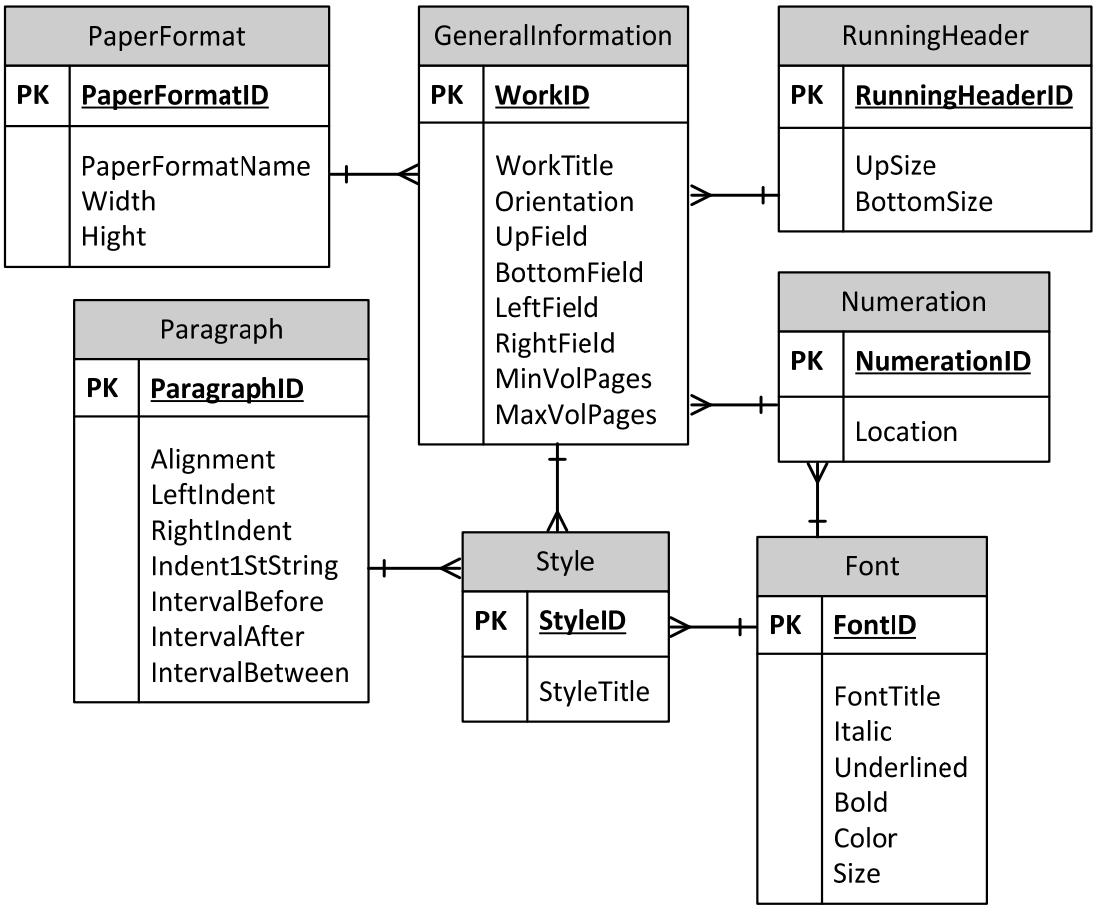


Figure 3. Conceptual Domain Model «Text Document Formatting Rules»

Office Open XML Document Format

Office Open XML (also known as OOXML or OpenXML) is a default file format of Microsoft Office electronic documents appeared for the first time in Microsoft Office 2007. The format was initially standardised by Ecma [Standard ECMA-376, 2015] and then redefined in ISO/IEC 29500 standard [ISO/IEC 29500, 2015]. OOXML is a structured archived file in a ZIP format that contains markup text of a document in an XML format, graphical information and other data included in this text document.

The advantages of an Open XML format [OpenXMLDeveloper.org, 2015] include:

1. Interoperability. The capacity of the format to interact and function with a large set of both custom and commercial applications provides a high degree of compatibility of documents for different tasks.

2. Backward compatibility. The ability of transformation of MS-DOC files into Open XML format with high accuracy allows end users to convert these documents to the Open XML format, and then programmatically access the converted documents.

3. Programmability. Minimum requirements for working with Open XML include a tool that can open and save ZIP files and an XML parser/processor. ZIP and XML libraries allow to create documents in Open XML format on a software level.

4. Integration of business data. Office applications support custom XML schemas that can extend the capabilities of the existing Office document types. Thus, users can export data from existing systems to the documents in the Office file formats.

5. Compact file format. For storing documents Open XML format uses the technology of ZIP compression, providing the possibility of reducing the storage space. Opening the file causes the automatic unpacking of the archive, and saving the file results in its compressing.

Compliance assessment in accordance with formatting rules is implemented for Microsoft Office Word text documents with the DOCX extension. A markup language for processing text files in an Open XML format is called WordprocessingML. The structure of WordprocessingML consists of a set of basic elements including main document, comments, settings, endnotes, header/footer, styles, fonts table, document glossary. Figure 4 illustrates parts of a document TestFile.docx opened with a tool Open XML Package Editor PowerTool for Visual Studio that allows to view the file hierarchy of the document archive and the relationships between them, and also to modify their markup.

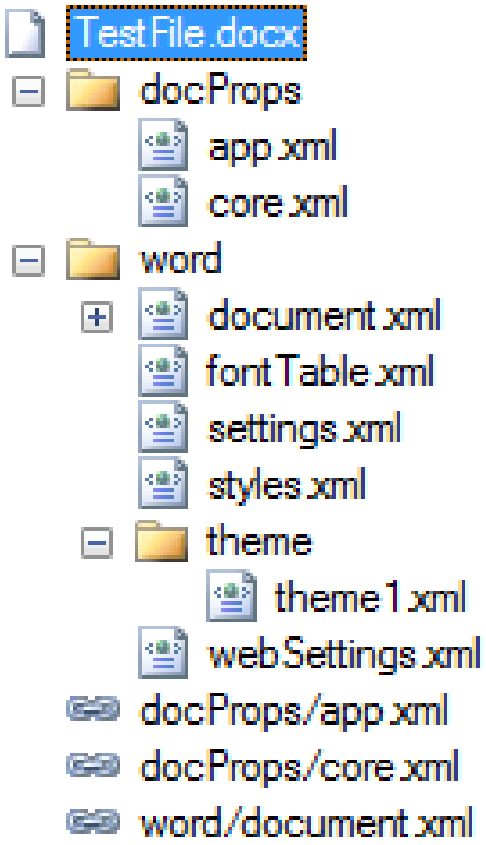


Figure 4. Text Document Structure in DOCX

In the main document part paragraphs (w:p) and tables (w:tbl) can be child elements for document body (w:body), table cell (w:tc) or text box (w:txbxContent). Paragraphs, in their turn, are a run-level content container for text runs (w:r), or images – a VML document (w:pict) or a DrawingML object (w:drawing). Finally, sub-run-level content incorporates multiple text elements (w:t).

System Development

Formatting of a text document with the use of Microsoft Word implies implementation of various styles with parametres included in styles.xml file of a document archive [Vugt, 2015]. This file contains data on styles of paragraphs, characters and tables, latent styles and standard settings of styles for an entire document (document defaults). Styles of paragraphs, characters and tables comprise information about current formatting of a document, whereas hidden styles are not used directly and serve primarily as a cache repository for style settings, for example, the ones copied from a template. Standard styles store default values for formatting of an entire document. However, it should be noted that styles.xml file does not involve data on formatting numbered and bulleted lists that is included in a special numbering.xml file.

The fact that content of a document can be formatted on multiple levels leads to a problem of determining a comprehensive set of formatting parameters used for a particular paragraph or a run of text. These levels of fomatting are schematically represented in Fig. 5.

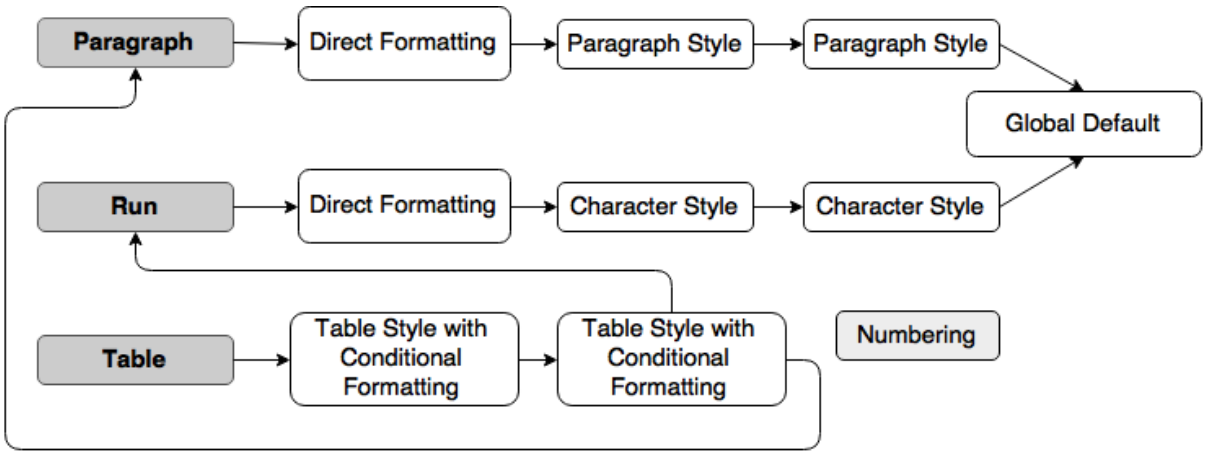


Figure 5. Levels of Microsoft Word Text Document Formatting

Thus, if it is needed to retrieve information about paragraph (e.g. line spacing or indentation), the first aspect that has to be checked is direct formatting which is specified in a file called document.xml. Yet, paragraph parametres might not be indicated in this file, and, in this case, it is necessary to inspect style

which is referred to in paragraph properties. It might appear that this style does not contain data on the paragraph formatting, then it is required to check the styles from which it inherits. If this action did not bring any results, then the only option left is to process the contents of the node Global default, i.e. default settings of all styles in a document.

Similar approach is credible for check of formatting of runs of text (defining such font settings as size, name, etc.); the only difference is that character styles are considered (as opposed to paragraph styles) that can also form an inheritance hierarchy.

Data on tables formatting is defined in styles with conditional formatting that specify properties of rows and columns. Table styles are also inherited. Text inside table cells is checked according to algorithms of determining formatting of paragraphs and runs. In case of numbering, each list item may include formatting from a paragraph, a numbering format in numbering.xml or a style that is indicated by this format.

Overall, the major difficulty of text document formatting check lies in determining precise formatting parameters for paragraphs, tables, numbering and runs of text for the purpose of conducting as extensive an analysis of conformity of a document to specified rules as possible.

In order to work with WordprocessingML markup, it was decided to use Open XML SDK 2.5 for Microsoft Office. Retrieval of information on document content formatting was performed by using the FormattingAssembler module which is a part of PowerTools for Open XML.

Open XML SDK built on the System.IO.Packaging API allows manipulating documents that adhere to the Office Open XML File Formats Specification, e.g. documents created with Microsoft Office applications. This package provides a set of strongly-typed classes to obtain data about the formatting of a document and makes it possible to modify an original document (for example, to add comments).

Despite the fact that .NET offers standard interop assemblies for working with Microsoft Office, the preference was given to Open XML SDK. COM Interop (Component Object Model) provides access to Word objects (sections, paragraphs, characters, etc.) and has functionality for creating and editing documents, however, it does not support server-side automation and processes documents markedly slower than SDK.

For the reason that for all numerical characteristics (margin sizes, font sizes, line spacing, etc.) in Open XML SDK are defined in such unit of measure as points, it was needed to transform them into millimeters and centimeters which are more familiar to a user setting formatting parameters for checking. Therefore, it was decided to use the iTextSharp class library that allows converting all of the values by implementing the only method – PointsToMillimeters().

In addition, in order to view a Word document from an application, WinWordControl developed by Matthias Haenel was used. WinWordControl modified in accordance with the latest version of

Microsoft.Office.Interop.Word DLL works with a set of basic functions of Win32 API. The appearance of this control with loaded document is represented in Fig. 6.

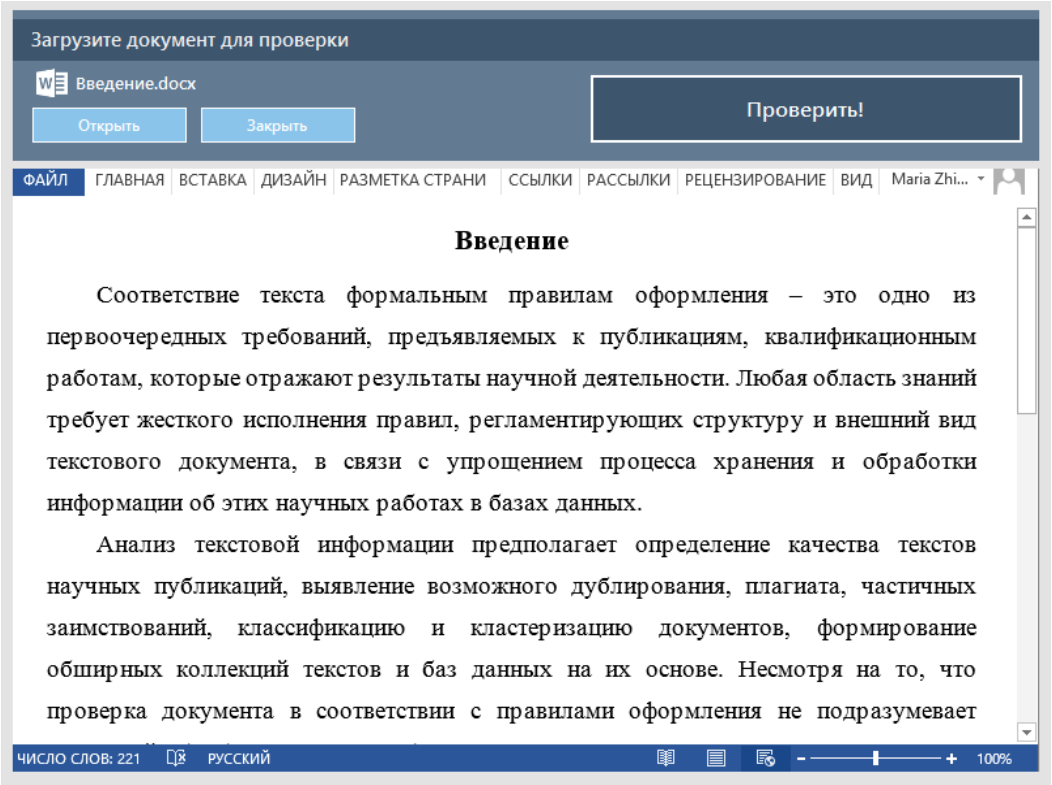


Figure 6. WinWordControl for viewing Microsoft Word documents

The necessity of using third-party control can be explained by the fact that Windows Forms does not provides standard tools to view Microsoft Office documents. The principle of operation is that an instance of Microsoft Word with an open user's document is started inside an application window, while the version of Microsoft Word corresponds to the one installed on the computer. Thus, this control can be used only if a user's computer runs Microsoft Word.

The solution "FRC System", implementing the system, contains two projects: "FormattingRulesLibrary" (class library for working with formatting rules of documents) and "FRC System" in which the application user interface was created and methods for text document check were developed.

The main window of the application that opens when the system is launched (see Fig. 7) is logically divided into two parts. On the right, there are the controls for displaying the formatting rules of text documents, downloaded from a database. The navigation through records of the database is performed by means of special navigation controls. Left side of the window (where the checked document is displayed) contains WinWordControl. Buttons "Open", "Close", "Check" provide functionality for opening, closing and document check, respectively. When the document is loaded into the system, the title of the checked Word document replaces the label "Document is not selected".

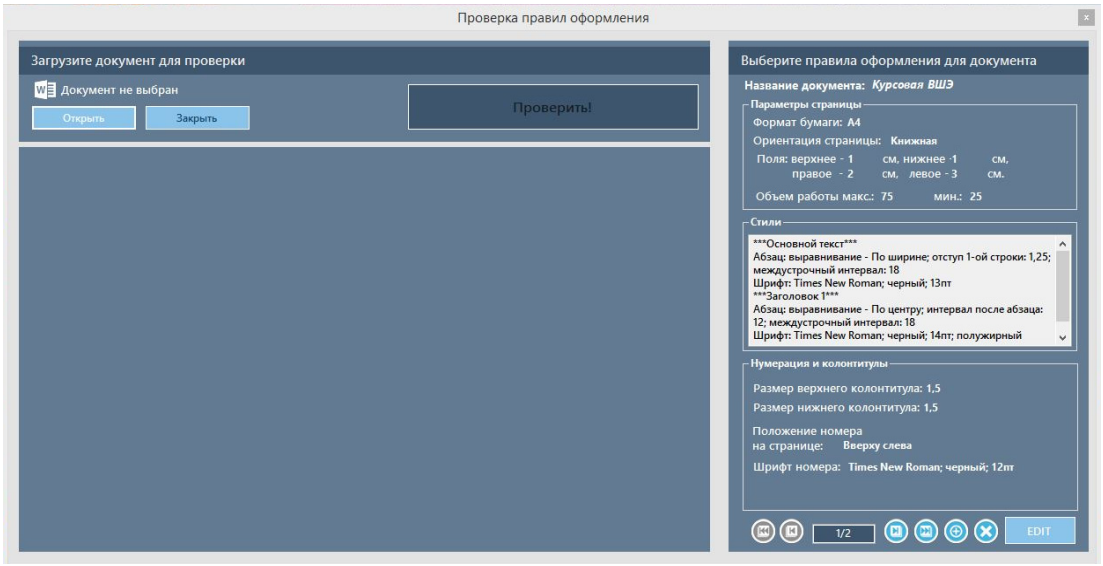


Figure 7. Main Window of the System for Document Check

The window for adding and editing new formatting rules of text documents (see Fig. 8) contains tabs, multiple input fields, comboboxes and other controls, allowing selecting necessary formatting parameters of a document.

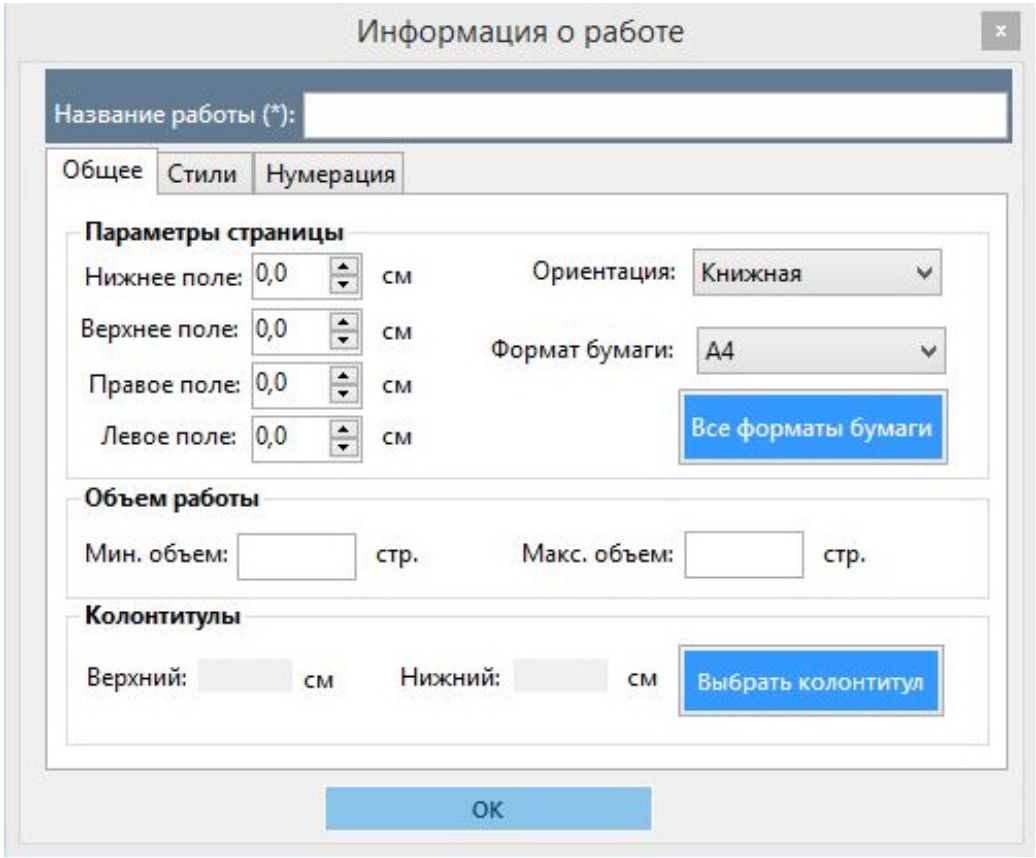


Figure 8. Window for Editing Formatting Rules

A mandatory field to fill in is the "Title of work". Various tabs allow specifying different requirements for formatting of a document. The tab "General" comprises information about page settings, number of pages, header and footer. On the tab "Styles", the user can examine styles of text formatting used in the document, add new styles or delete existing ones. The tab "Numbering" includes means for viewing and editing data on the numbering. Saving added/edited information about formatting rules occurs when the "OK" button is pressed.

The window for viewing and editing information about styles that are used for document formatting is demonstrated on Fig. 9.

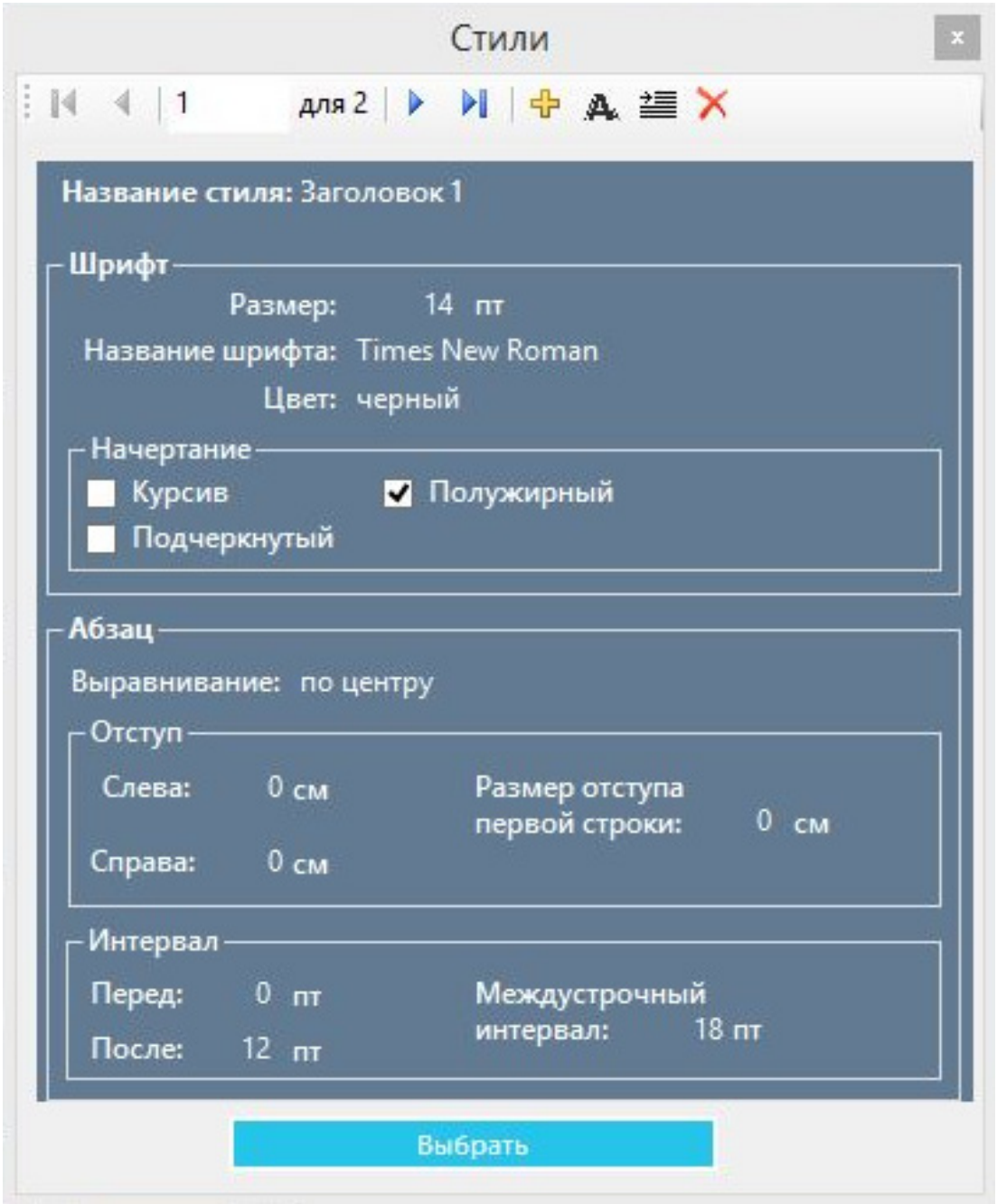


Figure 9. Window for Working with Styles

To modify font parameters used in the style, it is needed to press the button with the letter "A" on the navigation panel; editing paragraph properties requires selecting the control nearby (image depicting lines of text). Choosing style for a document occurs when the button "Choose" is pressed.

Loading a document into the system after pressing the "Load" button in the main application window is performed as follows: first, a user selects a needed file by the use of OpenDialog, then the document is opened in WinWordControl which allows displaying text of the document in the main window.

Document check is executed by the PerformChecking() method. First, a loaded Microsoft Word document is opened as an instance of the WordprocessingDocument class for working with Open XML SDK, and then, it is divided into sections and checked with the use of such methods as HeaderFooterCheck(), NumPagesCheck(), MarginCheck(document, sectPr), etc.

The interaction between a user and the system can be performed according to different scenarios. The first scenario (see Fig. 10) assumes that a user enters formatting rules manually, and then loads an original document for check. In this case, the system provides a user with either the resulting document containing the notes or the one with formatting corrected in accordance with rules specified by a user.

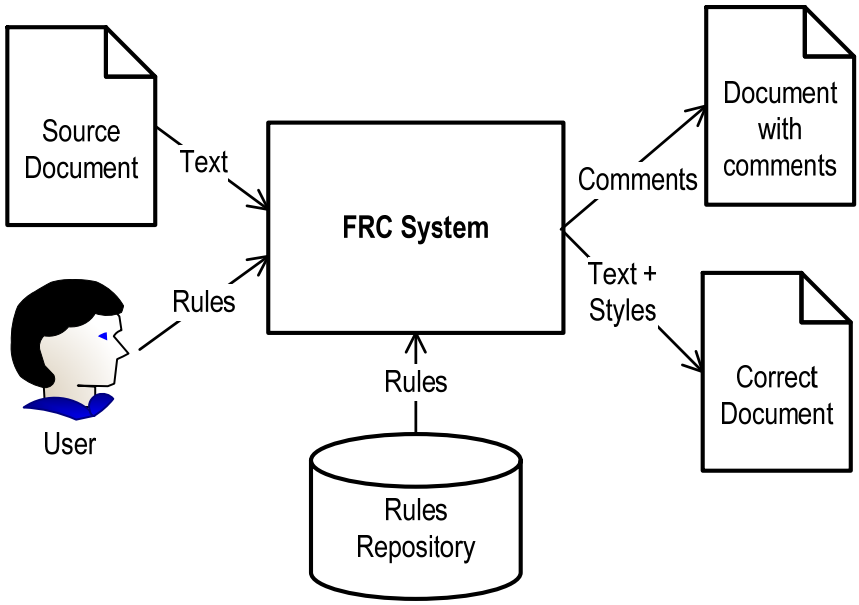


Figure 10. Correct Document Generation in FRC System

According to the second scenario (see Fig. 11) a user loads a properly formatted original document, the system performs its analysis and downloads its formatting rules into the rules repository. This significantly simplifies the entry of formatting rules of a document.

The third scenario of the interaction (see Fig. 12) suggests that a user manually enters formatting rules of a document, the system saves them in the repository, and then generates a document template with an automatically created styles which a user can use for further work with a document.

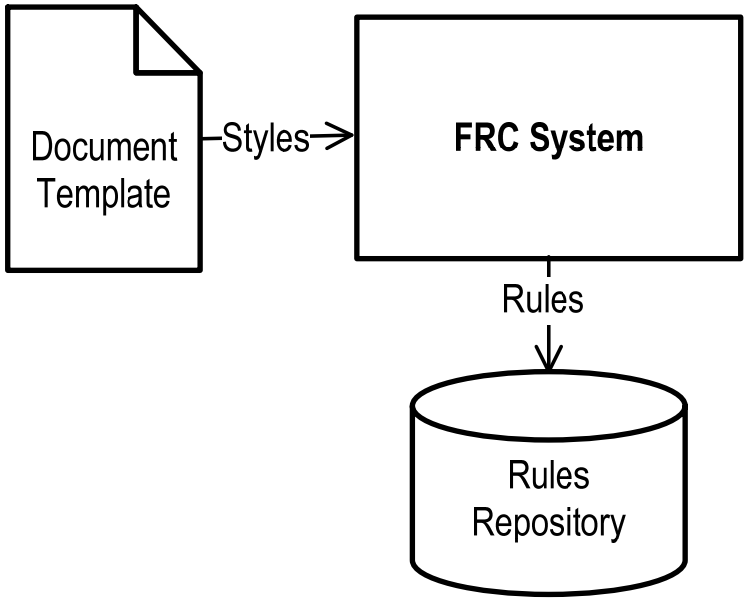


Figure 11. Creation of Rules based on Document Template in FRC System

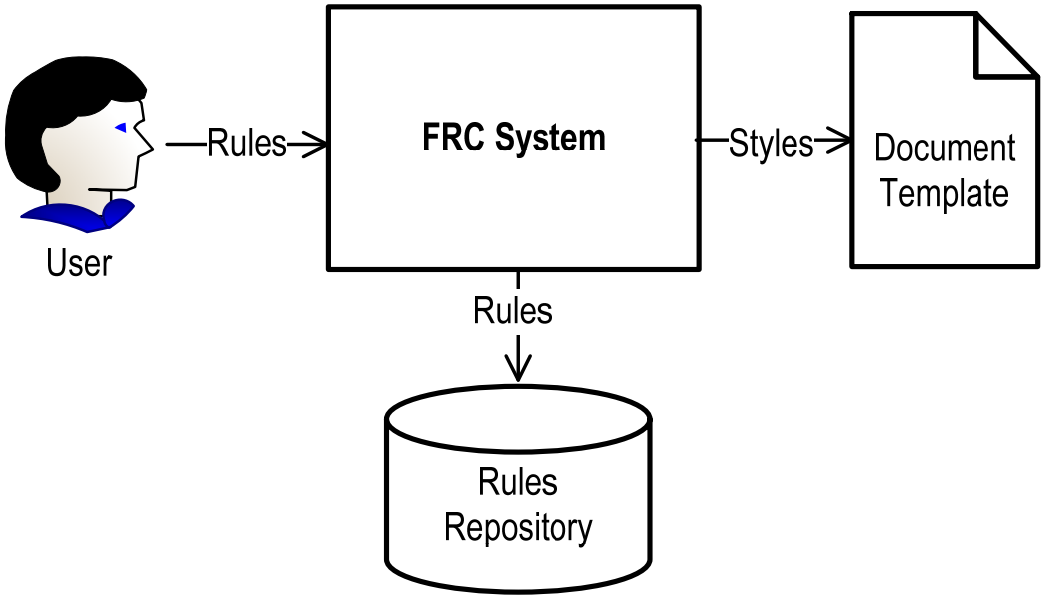


Figure 12. Creation of Document Template based on Rules in FRC System

Text Document Structure Check

As noted earlier, the task of a text document analysis is not reduced to formatting rules check. In the more general case, it is necessary to analyze document structure, i.e. verify that all required sections are included. This problem often arises in preparation of design documentation, for example, in the process of developing information systems. Design documentation has a normative function, i.e. it contains mutual obligations of participants of a project that helps to avoid misunderstandings and abuses at the stage of handover-acceptance [Zaboleeva-Zotova, 2007; Orlova, 2011].

The types and completeness of project documents are standardized. However, due to the fact that all technical documents are structurally very similar (they all consist of sections and subsections, may include additional documents, diagrams, tables, etc.), a special language for defining document structure and links between different documents can be developed. It will allow automating the process of analysis of an original set of project documents and generation of the new ones [Zhigalova, 2014]. In the same way, it is reasonable to develop tools for extracting system requirements from the project documentation, and then control their compliance in the process of implementing the system. However, the process of creating design documents is quite a laborious task that requires precise knowledge of a document structure. This process can also be automated. Means of automating the generation of project documentation will allow generating a document template on the basis of descriptions of different sections of a document specified in a convenient visual user interface. This template can later be modified manually.

In order to describe documentation used in the process of information systems design, visual domain-specific language can be developed. Domain-Specific Language (DSL) is a modeling language designed for solving problems of a certain class in a particular domain. Unlike general-purpose modeling languages, DSL is more expressive, easy to use and intelligible to various categories of professionals, since it operates with familiar terminology of the domain. Therefore, a large number of DSMLs is designed nowadays in order to describe systems in different domains: artificial intelligence systems, distributed systems, mobile applications, real-time and embedded systems, simulation systems, etc.

Since description of project documents implies not only determining their structure, but also specifying the relations between them, the developed domain-specific language describing project documentation has two levels [Zhigalova, 2015].

The first level of the language makes it possible to describe a set of documents and relations between them, the second level – the structure of a particular document. Due to a simple graphical notation of the language, the system can be used by IT‑specialists, as well as clients who are not professional programmers.

Conclusion

The main result of the work done is the developed system that automates the check of a text document in accordance with formatting rules specified by a user. As it was tested, the system substantially reduces the complexity of operations performed and makes the process less time-consuming.

Moreover, the visual DSL for describing the structure of a document was created. This language can be integrated into the support system of work of an analyst when information systems are designed. On the one hand, this provides means to perform analysis and parsing of a set of design documents loaded into

system, presenting the sections of a document as individual elements of a model. On the other hand, with the use of the developed language an analyst can describe each section of a design document separately, and then generate a single text description on their basis.

Despite the fact that the system performs all the main functions, there is still space for improvement. The system can be upgraded by developing web-interface for more convinient use and expanding the set of criteria for document check in order to perform more comprehensive compliance assessment.

Acknowledgements

This paper is supported by the Russian Foundation for Basic Research (grant 14-07-31330).

Bibliography

[Berdachuk, 2015] S. Berdachuk. Eclipse RCP. File Manager. Use the DocBook for Documentation Writing. [http://www.berdaflex.com/ru/eclipse/books/rcp_filemanager/ch01s04.html] (Checked: 21.10.2015).

[E-iceblue, 2015] E-iceblue. Your office development master. [http://www.e-iceblue.com] (Checked: 21.10.2015).

[ISO/IEC 29500, 2012] ISO/IEC 29500. Information technology – Document description and processing languages – Office Open XML File Formats. International Organization for Standardization, Geneva, Switzerland, 2012.

[Lvovsky, 2006] S.M. Lvovsky.Typing and formatting in LaTeX System. Moscow: MTSNMO, 2006.

[OpenXMLDeveloper.org, 2015] OpenXMLDeveloper.org. Benefits of Open XML. [http://openxmldeveloper.org/wiki/w/wiki/benefits-of-open-xml.aspx] (Checked: 21.10.2015).

[Orfogrammka, 2015] Orfogrammka. Spelling Checking Web Service. [http://orfogrammka.ru] (Checked: 21.10.2015).

[Orlova, 2011] Yu.A. Orlova. Product Requirements Document Text Analysis Methods. In Izvestiya TSU. Engineering Sciences, 2011. No 3. P. 213-220.

[Sokolov, 2013] A.A. Sokolov, A.M. Dvoryankin, A.Yu. Uzhva. Development of the Method of Process of Technical Documentation Normative Control Automation. In Izvestia VSTU, 2013. No. 22 (125). P. 116-117.

[Standard, 2015] Standard ECMA-376. Office Open XML File Formats. [http://www.ecma-international.org/publications/standards/Ecma-376.htm] (Checked: 21.10.2015).

[Vugt, 2015] W. Vugt. Open XML Explained. [http://openxmldeveloper.org/cfs-file.ashx/__key/communityserver-components-postattachments/00-00-00-19-70/Open-XML-Explained.pdf] (Checked: 21.10.2015).

[Zaboleeva-Zotova, 2007] A.V. Zaboleeva-Zotova, Yu.A. Orlova. Automation of Procedures of the Product Requirements Document Text Semantic Analysis. In Izvestia VSTU, 2007. No 3. Vol. 9. P. 52-55.

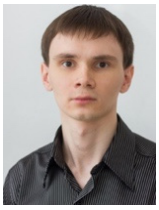
[Zhigalova, 2014] M.A. Zhigalova, A.O. Sukhov. Validation of the Design Documentation Based on Domain-specific Language. In Bulletin of young scientists PSNRU. Vol. 4. P. 224-228.

[Zhigalova, 2015] M.A. Zhigalova, A.O. Sukhov. Domain-specific Language for Describing Documents Used in Information Systems Design. In Izvestiya SFedU. Engineering Sciences, 2015. No 2. P. 126-134.

Authors' Information

Maria Zhigalova *– National Research University Higher School of Economics, Department of Information Technologies in Business, student; 38, Studenceskaia St., Perm, 614070, Russia; e-mail: mariezhigalova@gmail.com.*

Major Fields of Scientific Research: Software Engineering, Business Informatics, Domain specific modelling



Sukhov Alexander *– National Research University Higher School of Economics, Department of Information Technologies in Business, associate professor; 38, Studenceskaia St., Perm, 614070, Russia; e-mail: ASuhov@hse.ru.*

Major Fields of Scientific Research: Software Engineering, Modelling languages, Visual modelling, Domain specific modelling, Domain specific language, Language workbenches