International Journal of Business, Human and Social Sciences
ISSN: 2517-9411
Vol:9, No:2, 2015

Wasting Human and Computer Resources

Mária Csernoch, Piroska Biró

***Abstract*—**The legends about "user-friendly" and "easy-to-use" birotical tools (computer-related office tools) have been spreading and misleading end-users. This approach has led us to the extremely high number of incorrect documents, causing serious financial losses in the creating, modifying, and retrieving processes. Our research proved that there are at least two sources of this underachievement: (1) The lack of the definition of the correctly edited, formatted documents. Consequently, end-users do not know whether their methods and results are correct or not. They are not aware of their ignorance. They are so ignorant that their ignorance does not allow them to realize their lack of knowledge. (2) The end-users' problem solving methods. We have found that in non-traditional programming environments end-users apply, almost exclusively, surface approach metacognitive methods to carry out their computer related activities, which are proved less effective than deep approach methods.

Based on these findings we have developed deep approach methods which are based on and adapted from traditional programming languages. In this study, we focus on the most popular type of birotical documents, the text based documents. We have provided the definition of the correctly edited text, and based on this definition, adapted the debugging method known in programming. According to the method, before the realization of text editing, a thorough debugging of already existing texts and the categorization of errors are carried out. With this method in advance to real text editing users learn the requirements of text based documents and also of the correctly formatted text.

The method has been proved much more effective than the previously applied surface approach methods. The advantages of the method are that the real text handling requires much less human and computer sources than clicking aimlessly in the GUI (Graphical User Interface), and the data retrieval is much more effective than from error-prone documents.

***Keywords*—**Deep approach metacognitive methods, error-prone birotical documents, financial losses, human and computer resources.

I. Introduction

IN the Graphical User Interface (GUI), end users almost exclusively apply surface approach methods for computer-related problem-solving. As a result they focus on providing outputs, instead of solving problems [1], [3], [4], [13], [15]. This approach has led to a high percentage of error prone administrative (birotical) documents, which are highly demanding of human and computer resources, causing serious financial losses both in the productive and retrieval processes [8], [12]–[16], [18]. It is not common knowledge that non-traditional software environments are also algorithmically driven, and to solve problems in these programs the same approach should be applied as with "real" programming [2], [5], [17], [19].

M. Cs. is with University of Debrecen Faculty of Informatics (phone: +36-52-512-900/75128; e-mail: csernoch.maria@inf.unideb.hu).

P. B. is with University of Debrecen Faculty of Informatics (e-mail: biro.piroska@inf.unideb.hu).

Faced with these problems, we have developed deep approach metacognitive methods for computer-related problem-solving in "birotical" environments, focusing on text management in the present paper. The methods are adapted from high level programming languages, and follow problem-solving and debugging methods which have so far proved effective and efficient.

II. Theoretical Background

A. Metacognitive Approaches to Computer-Related Problem-Solving

To cover all computer related activities, both in traditional and non-traditional programming environments, Case and Gunstone's [6] well accepted system of metacognitive problem solving approaches had to be extended (Fig. 1) with one deep (CAAD, Computer Algorithmic and Debugging-based) and one surface approach (TAEW, Trial-and-Error Wizard-based) category [7].

B. Levels of Mastery

Generally speaking, non-traditional software environments are not considered programming environments. The GUI and the support from the software companies suggest that user-friendly environment does not require any algorithmic skills, and there is no need for any computational thinking [20]. However, this is not so; to work effectively in "birotical" environments, similar to programming, the order of the three levels of mastery should also be followed [7]:

- *Familiarity:* The student understands what a concept is or what it means.
- *Usage:* The student is able to use or apply a concept in a concrete way.
- *Assessment:* The student is able to consider a concept from multiple viewpoints and/or justify the selection of a particular approach to solve a problem.

The GUI-support approaches leave out the first level of mastery, and the focus is on the usage. However, with this approach there are at least two fundamental problems:

- The one mentioned earlier; the high number of error prone documents and the wasted human and computer resources,
- Even more series the other consequence; leaving out the first level of mastery, we would never have a chance to reach the third level, the assessment.

We have to note here that there are sources which claim that our failure in teaching Informatics and Computer Sciences are rooted in Word and Excel [21]. However, we are convinced that Word and Excel should not be blamed. They are harmless, algorithmic based pieces of software. The popular and widely accepted surface approach methods and those who teach and

apply them are responsible. We claim that instead of Word, Excel and PowerPoint we should teach text and spreadsheet management, we should teach the different aspects of the correctly designed and formatted documents [21].

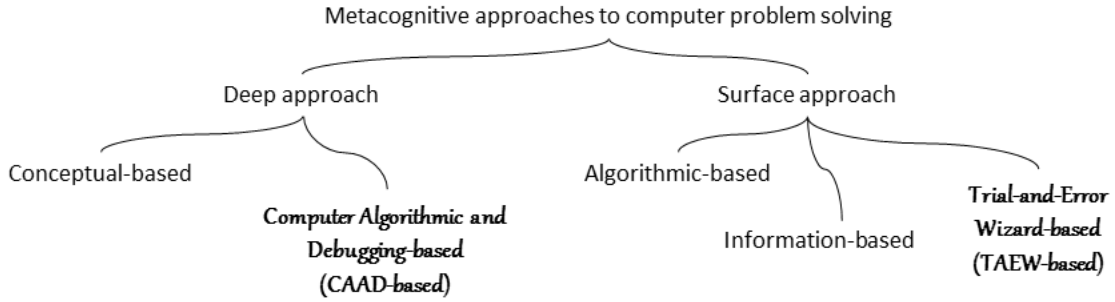


Fig. 1 Metacognitive approaches to computer-related problem-solving. We have extended Case and Gunstone's original typology with approaches to computer-related activities (in Italics)

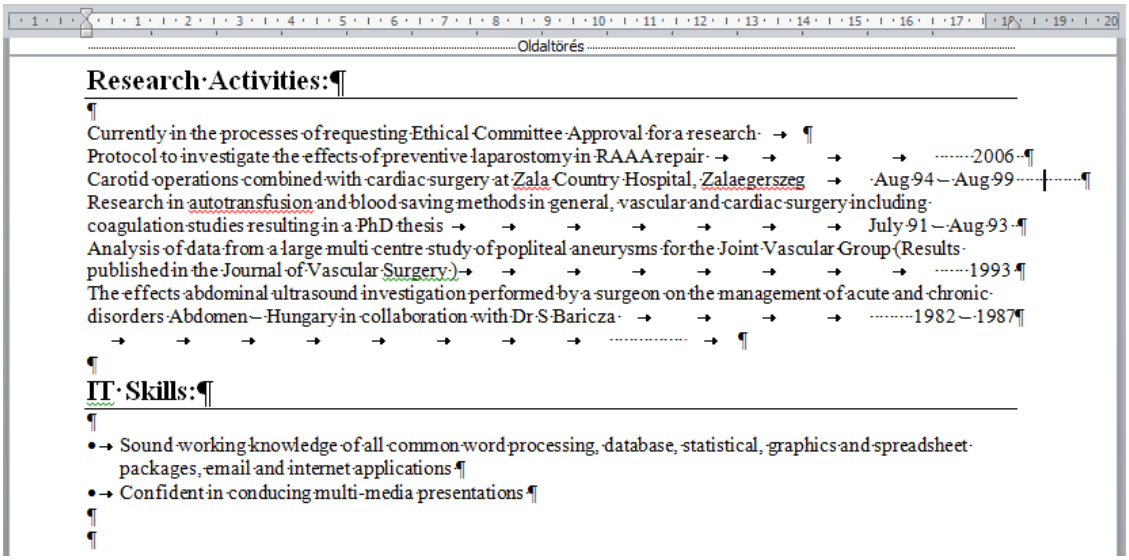


Fig. 2 The text is loaded with layout errors. Its author is a confident but ignorant end-user, who is unaware of their ignorance

III. Results

Computer assisted natural language text management is still a great challenge and will continue to be both in the short and long term. The complexity of the problem makes it clear that surface approach methods to text management will not succeed; instead, deep approach methods are needed to carry out text handling and data retrieval effectively and efficiently. The method introduced in this paper, entitled Error Recognition and Classification (ERaC), is a CAAD-based method (Fig. 1) [9], adapted from programming languages for handling text-based "birotical" documents [2], [5], [17], [19].

A. Correctly Formatted Text

To apply the ERaC deep approach metacognitive method to computer assisted text management – regardless of whether the output is paper-based or in an electronic format–, it is necessary to define the correctly edited and formatted text. Any lack of this definition would lead to a different understanding of mastery and knowledge, and indeed this is the case at present (Fig. 2). The author of the text in Fig. 2 states that he has a "sound knowledge of all common word processing", which is obviously not so. The reason for this misunderstanding is the lack of the definition of the correctly edited, formatted text [8], [10]–[14]. The ignorance of the author of Fig. 2 prevents them to realize their lack of knowledge. They do not know that any changes to the body of the text would evoke further unplanned sequence of changes, which requires additional time, human and computer resources.

Definition: A test is correctly edited and formatted if it is invariant to modification and fulfills the requirements of printed text-based materials.

The definition allows and supports modification of the body of the text, but only those modifications are accepted which conform to the intentions of the user. Any correction – i.e. typing (including deleting) and formatting – beyond the user's original intention is not allowed. The definition also considers rules related to any printed materials – paper or e-output [12], [13] (for details see Chapter B).

The text in Fig. 2 cannot bear any modification – deleting or

adding pieces of text – or any formatting; even a simple change of font size would ruin the arrangement of the text (see Figs. 4 and 7).

B. Classification of Errors

To provide error free texts, users have to be familiar with the different kinds of errors and the nature of these errors. However, the high number of possible errors does not allow them to be handled individually. Considering these requirements, the following major error classes are defined:

- layout (break up),
- formatting,
- syntactic,
- semantic,
- typographic.

However, we must note that these error groups are not exclusive; one error of a certain type would indicate another type (Figs. 4, 5, and 7, 9; layout errors and formatting errors cause typographic errors, respectively).

C. Examples of Errors and Their Classification

Layout errors: Layout errors are those which break the text into meaningless chunks to imitate formatting. These are the most demanding errors, because they prevent any modification and proper formatting of the text. There are some automated corrections available in order to lessen the number of these errors, but only a very few are successful. The non-printable characters clearly present the most common layout errors in text management: extra spaces (Figs. 2-5, 9, 11, 12), extra tabs (Figs. 2, 3 B), extra end of paragraph marks (enters) (Figs. 2, 3 A, 4 A, 5, 9 B, 11), and manual paragraph numbering (Figs. 3 A, 5, 9 B, 11).

Layout, formatting, and most of the typographic errors are independent of languages. In the figures we use the authentic texts with the original languages to demonstrate that these errors are not the specialties of the English language. Not only the problems but handling them is also language independent.

To demonstrate the consequences of the incorrectly used end of paragraph character we selected a sample presented in Fig. 4. In the first section there is one enter at the end of the paragraph (following *Picture ID*), while in the second sections enters are typed at the end of each line (*available*, *and*, *must be*, respectively, blue arrows). After decreasing and increasing the font size (samples B and C, respectively) the second section does not hold the original arrangement of the text. Beyond this error, the extra enters between paragraphs – replacing spacing before and/or after paragraph – are resizable (Fig. 4, red arrows and boxes), consequently with a change in font size the space between the paragraphs is also changed.

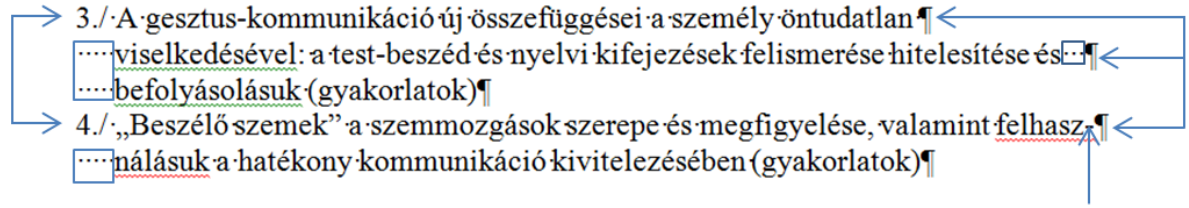
Fig. 5 is an example of high level bricolage. Instead of using the equation editor, the fraction is typed as text: the numerators and the denominators are placed in separated paragraphs and further "tricks" are applied.

It is also common that layout errors evoke syntactic and typographic errors. The incorrectly used spaces in Figs. 3, 4, 9, 11, 12 (lower sample) generated syntactic errors, and some of these errors evoked typographic errors, also.

Syntactic and semantic errors: Most end-users are familiar with the concept of syntactic errors, and even spellcheckers provide assistance in avoiding some, but not all of them. The most frequent syntactic errors occur in connection with punctuation marks and other special characters (Fig. 4). However, semantic errors, which are in close connection with the syntactic errors, are less known and recognizable. In the sample of Fig. 4 we can find marked and unmarked examples of syntactic errors, caused by the incorrect use of spaces, and there is one marked semantic error.

Fig. 6 shows two examples of semantic errors which were overlooked by the spellchecker. The reason for this failure is that both words incorrect in the context are in the dictionary of the spellchecker: *here* instead of *hear* and *boat* instead of *coat*, respectively.

A



B

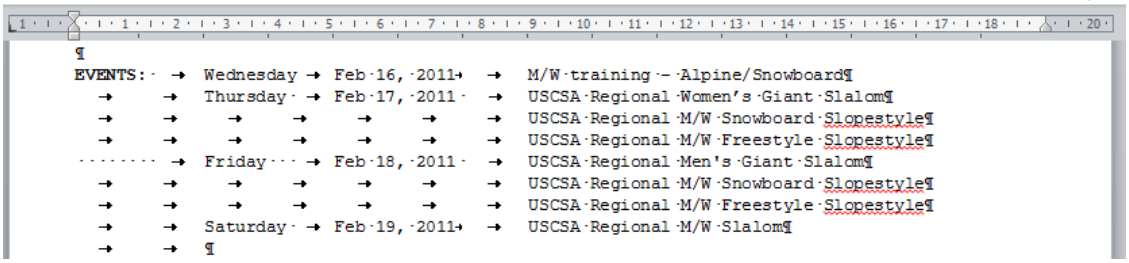


Fig. 3 Layout errors: Manual paragraph numbering, extra spaces, end of paragraph marks in the middle of the sentences, and manual hyphenation (A), incorrectly used spaces and tabulator characters instead of positioned tabulators (B)

A

MAKE-UP EXAMS:¶
···All make-up exams will be administered in the Assessment Center (or may given by me by arrangement.) You will need to arrange with me to have a test left in the Assessment Center. To take a test there, you MUST show a PICTURE ID. ¶
¶
¶
EXAMS for FLEX class:¶
····ALL topic and map test will be taken at one of the NHMCCD Assessment Centers. (IF the tests become available ¶
on WebCT at a later date, I will let you know) The current events quizzes WILL be available online by Thursday and¶
due by Sunday. You will be notified as to when tests will be in the Assessment Centers and when they must be ¶
completed. ¶

B

MAKE-UP EXAMS:¶
···All make-up exams will be administered in the Assessment Center (or may given by me by arrangement.) You will need to arrange with me to have a test left in the Assessment Center. To take a test there, you MUST show a PICTURE ID. ¶
¶
¶
EXAMS for FLEX class:¶
····ALL topic and map test will be taken at one of the NHMCCD Assessment Centers. (IF the tests become available ¶
on WebCT at a later date, I will let you know) The current events quizzes WILL be available online by Thursday and¶
due by Sunday. You will be notified as to when tests will be in the Assessment Centers and when they must be ¶
completed. ¶

C

MAKE-UP EXAMS:¶
···All make-up exams will be administered in the Assessment Center (or may given by me by arrangement.) You will need to arrange with me to have a test left in the Assessment Center. To take a test there, you MUST show a PICTURE ID. ¶
¶
¶
EXAMS for FLEX class:¶
····ALL topic and map test will be taken at one of the NHMCCD Assessment Centers. (IF the tests become available ¶
on WebCT at a later date, I will let you know) The current events quizzes WILL be available online by Thursday and¶
due by Sunday. You will be notified as to when tests will be in the Assessment Centers and when they must be ¶
completed. ¶

Fig. 4 The consequences incorrectly used end of paragraph marks, enters

d) d + c _ 2d _ 3c + d d _ c ¶
3 4 6 12 12 6 6¶
¶
4d _ 4d + d 2d + 3c _ 3c _ 2c *¶
12 12 12 12 12 12 12¶
¶
d _ c 18 + 12 _ 3 + 2 _ 1 *¶
12 6 12 6 2 2¶

Fig. 5 Fractions created as text; high level bricolage

A How Did You Here About Hills Touch?

B **Patience: The 36-year-old donned a metallic gold dress underneath her furry black boat while she accessorised with a pair of gold drop earrings**

Fig. 6 Examples of unnoticed semantic errors

Formatting errors: Formatting errors occur when a formatting command is carried out but it is not correct. One type of formatting errors results texts which cannot be modified without any unintended formatting or typing (Figs. 7-12). The other type causes typographic errors (Figs. 8, 12 (upper sample), Figs. 13-16)).

Fig. 7 presents five consecutive paragraphs and their corresponding rulers. We can read from the rulers that in these five paragraphs center alignment is imitated by left indentation. The indentations are carefully set in each paragraph, which is an extremely tire-some process. By changing the font size of these paragraphs (Fig. 7, button sample) it is clear that the left indentation cannot be used for centering paragraphs. The fabrication of these paragraphs is only waste of time.

There are other formatting errors in this text:

- The margins are set to zero.
- The line spacing is set to a constant value. Consequently, the upper part of the characters do not necessarily fits into the high offered by the line spacing.

We cannot go unnoticed by the fact that this text is the production of a textbook publisher. (The language of the text is irrelevant, since we focus on the formatting errors.)

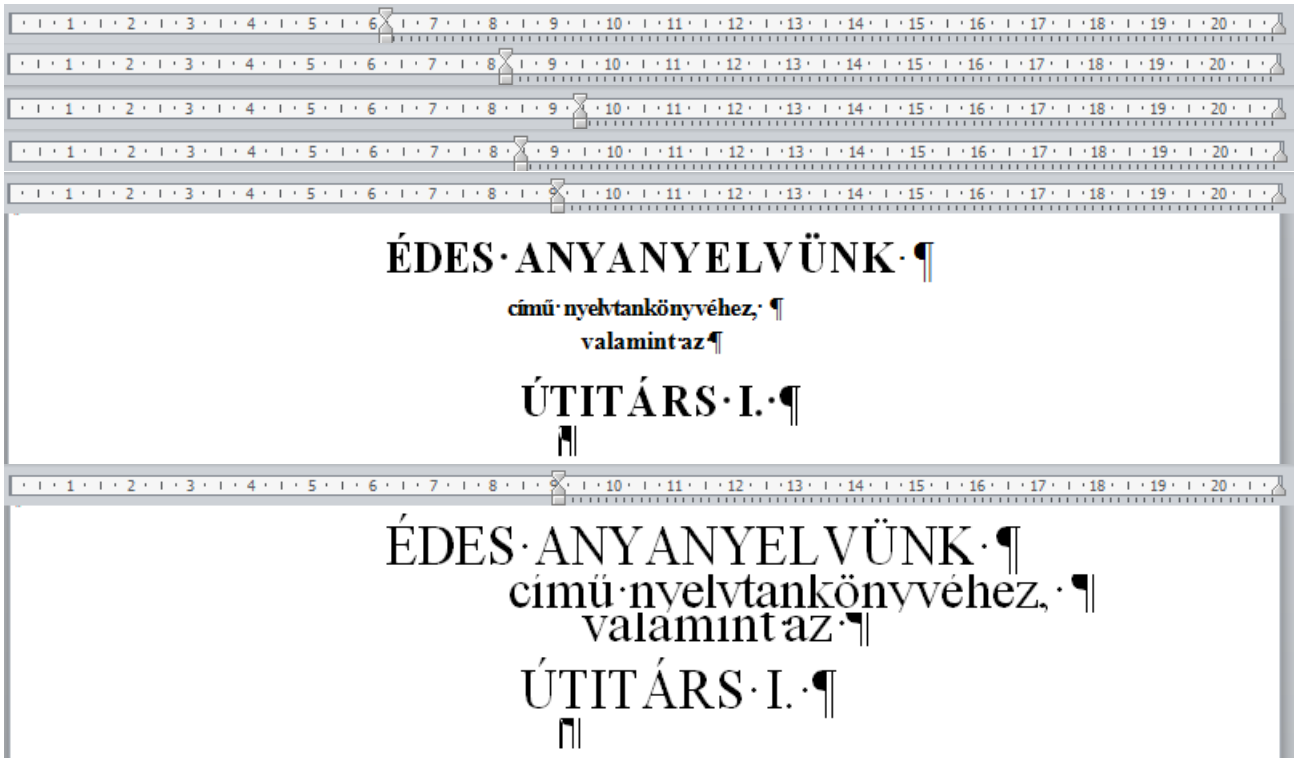


Fig. 7 The formatting errors of five consecutive paragraphs and the results of changing the font size in all the five paragraphs

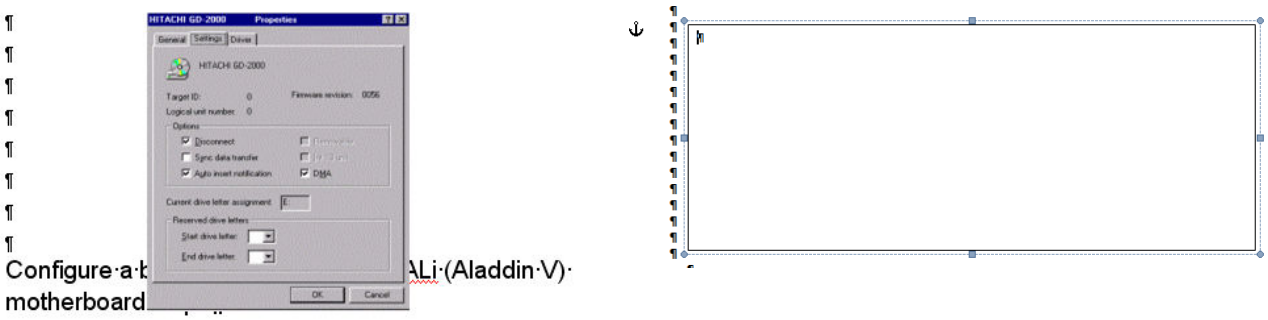


Fig. 8 Incorrectly set picture layouts

The most frequent formatting errors occur in connection with the following activities:

- Setting picture layout (Figs. 8, 9),
- Setting vertical and horizontal alignments (Figs. 7, 3 B, a positioned left tabulator at the left edge of the paper),
- Selecting newspaper instead of parallel columns (Fig. 10), and the other way around.
- Setting the table rows and columns incorrectly (Fig. 11).

The layout of the picture in Fig. 8 is set to *In front of text*, which is not correct. The picture in the left sample is placed on the top of the text in spite of the numerous, but obviously not enough, empty paragraphs. The picture should be placed in a separated paragraph, with the layout *In line with text*. We often meet the same problem with creating space holders (Fig. 8, right sample).

In Fig. 9, just the other way around, the picture layout is incorrectly set to *In line with text*, instead of *Square*.

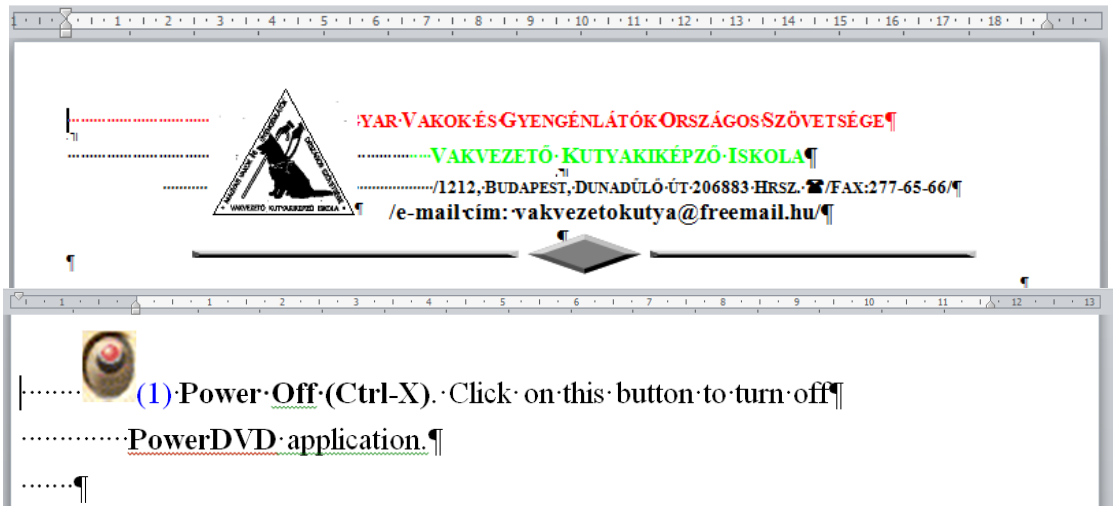
In Figs. 10, 11 imitated parallel columns are presented. There is confusion between newspaper and parallel columns. The difference between newspaper and parallel columns is the order of the reading. In newspaper columns we finish reading a column before we advance to the next one. In parallel columns we read the contents related to each other in one block, placed side by side, and we only return to the first column when a block is completely read.

In Fig. 10 instead of parallel columns newspaper columns with column breaks are used. With this extremely tiresome solution the text is fragile, and cannot bear any modification.

In Fig. 11 sample A and B there are less and more rows than needed, respectively. In Fig. 11 A, the user realized that to create parallel columns a table is needed. The number of columns is set correctly, however, the number of rows is not,

consequently, the text is broken into meaningless chunks, so it cannot be modified. In Fig. 11 B, there is one row for the title of the table – loaded with layout errors for imitating horizontal and vertical alignments – and one more row for all the other contents, using numerous tricks to separate the logical blocks.

A



B

Fig. 9 Formatting errors caused by setting improper picture layout

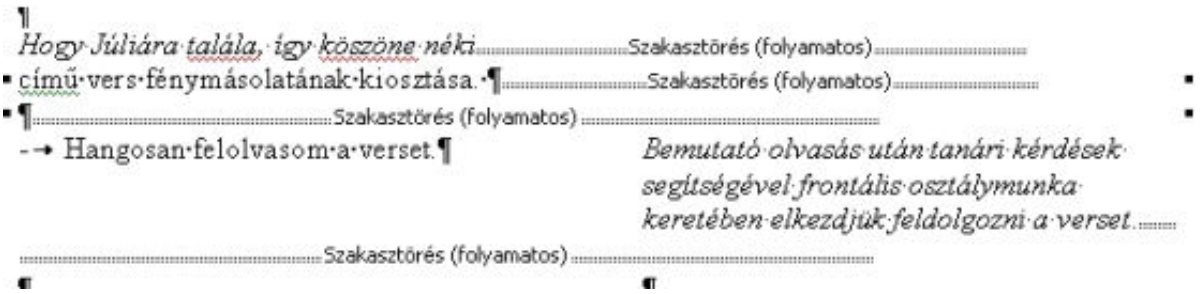
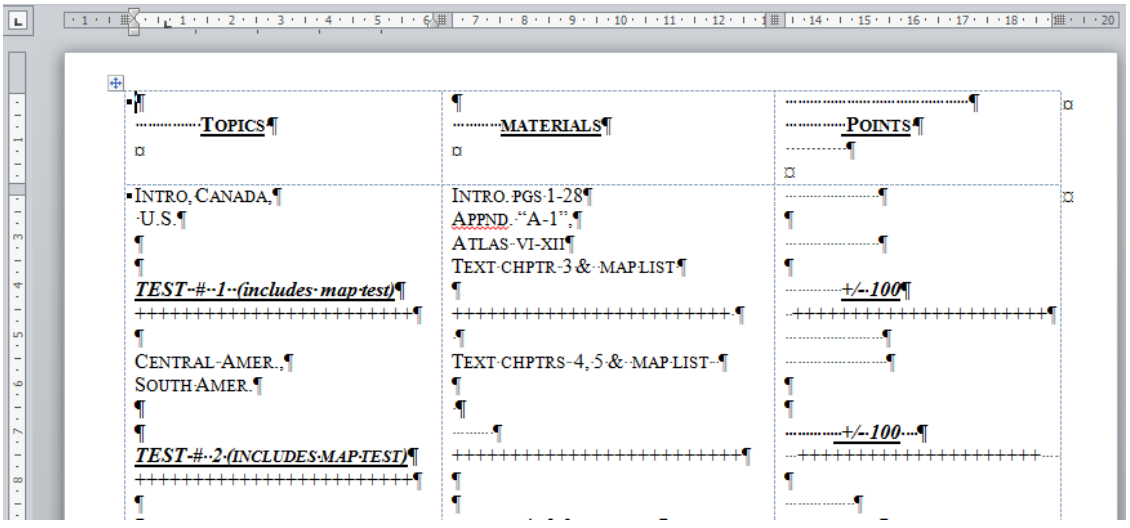


Fig. 10 Formatting error by selecting newspaper columns instead of parallel columns, realized in the sample by table

A



B

Beschreibung	Tätigkeit	Fertigkeit/Methode	Kontrolle	Lernziele
1. Erwärmungs-	Frontalarbeit	Sprechen	Die Lehrerin	Einstimmung
gespräch - Fragen			korrigiert die	
beantworten im			Fehler.	
Thema *Freizeit*				
2. Kontrollieren	Frontalarbeit	Sprechen	Die Lehrerin	Wiederholung,
der Hausaufgabe	Lösungen sagen		korrigiert die	Üben
	und korrigieren		Fehler.	

Fig. 11 The number of rows set incorrectly in a parallel column structure

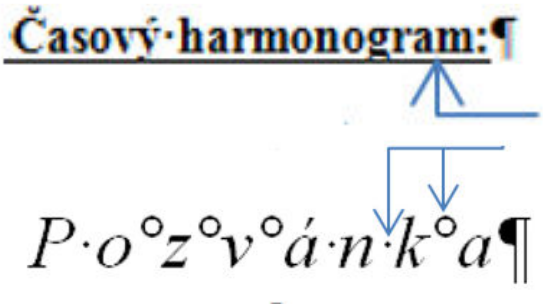
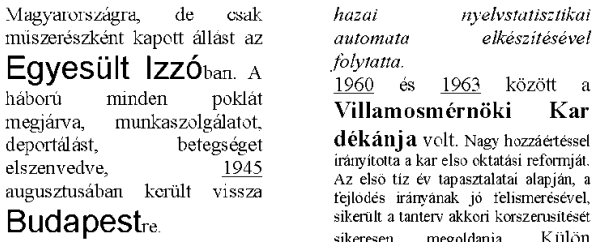


Fig. 12 Errors of one type evoking another type

Typographic errors: Typography provides the rules and advices governing the final appearance of the documents. Unfortunately, these rules and advices are not as widely known as they should be, and frequently violated not only by the end-users but also by the software companies who provide formatting buttons on center stages. End-users, not knowing any better and not caring, frequently use these buttons and create documents both for paper and e-output.

Typographic errors are frequently evoked by formatting or layout error. One of the most commonly found typographic error is the underlined text (Fig. 12, upper sample), which is carried out by a formatting. The samples of Figs. 13-16 are overloaded with different kind of typographic mistakes.

Overleaping errors: As it was mentioned and demonstrated in the previous chapters, there are errors which belong to more than one class. In addition to them the lower sample of Fig. 10 presents a very common method: normal and non-breaking spaces used to substitute *Character spacing*. This error is a layout error, however, it can be considered either as a syntactic or a typographic mistake.


Magyarországra, de csak miszerészként kapott állást az Egyesült Izzóban. A háború minden poklát megjárva, munkaszolgálatot, deportálást, betegséget elszenvedve, 1945 augusztusában került vissza Budapestre.

hazai nyelvstatisztikai automata elkészítésével folytatta.
1960 és 1963 között a **Villamosmérnöki Kar dékánja** volt. Nagy hozzáértéssel irányította a kar első oktatási reformját. Az első tíz év tapasztalatai alapján, a fejlődés irányának jó felismerésével, sikerült a tanterv akkori korszerűsítését sikeresen megoldania. Külön

Fig. 13 Typographic errors caused by the ill-use of character formatting

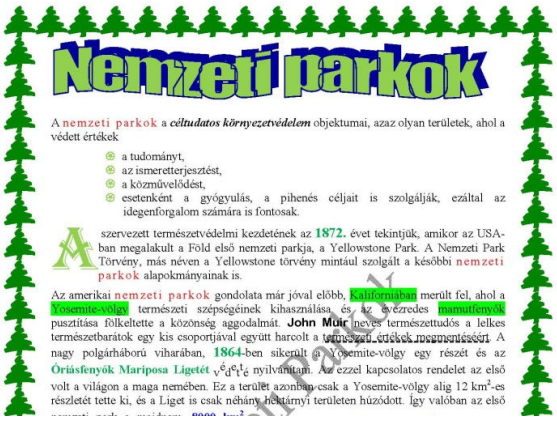


Fig. 14 Overused formatting of different kinds makes the text difficult to read

The errors and error classes are not exclusively the characteristics of the printed documents. The very same errors can appear in documents designed for e-output. The results are similar; creating these documents is highly demanding and they do not serve their original purposes, since they do not reach the desired public. Fig. 16 presents samples from presentations, demonstrating typographic errors.

We have to note, however, that at present there are no tools which can indicate either typographic or formatting errors. Consequently, end-users should be responsible for these rules, and they should be able to recognize and avoid them.

IV. Conclusion

In general, we can conclude that there are a great number of rules to consider during the process of handling text, and only a limited number of automated helps are available, and they are not necessarily reliable. Considering the complexity of text management, CAAD-based approaches are required in order to solve these kinds of problems. With CAAD-based methods, end-users would be able to build algorithms to carry out the steps involved in (1) building, (2) formatting, and finally (3) debugging the texts. With the Error Recognition and Classification deep approach method, adapted from programming, we would reduce the number of error-prone documents, and lessen the time and human and computer resources needed to carry out the creation-modification of the

texts, and the information retrieval based on them.

Fig. 15 Unfitted illustrations to programming tasks

Building algorithms and caring about debugging play crucial roles in a CAAD-based process of text management, supported by computational thinking. On the other hand, coding in GUI is easy; there is nothing else to do except click on the buttons to carry out the commands. At present, the emphasis is on the coding, providing an extremely high percentage of error-prone documents. However, we argue that for effective text management, end-users must switch the focus from coding to building algorithms and thoroughly debugging, and to this end we propose the Error Recognition and Classification method.

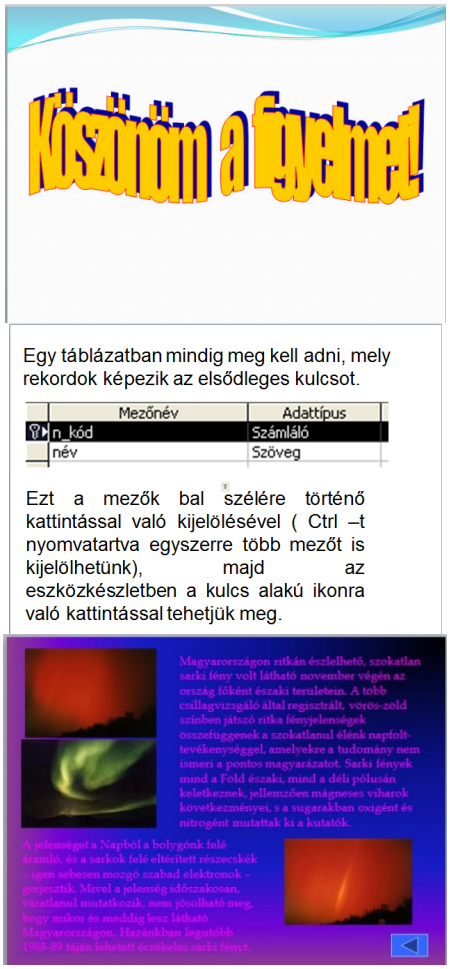


Fig. 16 Typographic errors in presentations

ACKNOWLEDGMENT

The research was supported by the TÁMOP-4.2.2.C-11/1/KONV-2012-0001 project. The project has been supported by the European Union, co-financed by the European Social Fund.

The research was supported partly by the Hungarian Scientific Research Fund under Grant No. OTKA K-105262.

REFERENCES

[1] M. Ben-Ari, "Bricolage Forever! PPIG 1999". 11th Annual Workshop. 5–7 January 1999. Computer-Based Learning Unit, University of Leeds, UK. Retrieved April 12, 2014 from http://www.ppig.org/papers/11th-benari.pdf.

[2] P. Biró, and M. Csernoch, "Deep and surface structural metacognitive abilities of the first year students of Informatics." 4th IEEE International Conference on Cognitive Info-communications, Proceedings, Budapest, 2013, pp. 521–526.

[3] P. Biró, M. Csernoch, K. Abari, and J. Máth, „First year students' algorithmic skills in tertiary Computer Science education." KICSS 2014. Springer Series: Advances in Intelligent Systems and Computing (AISC), 2014, ISSN 2194-5357. Accepted.

[4] P. Biró, M. Csernoch, J. Máth, and K. Abari, „Measuring the level of algorithmic skills at the end of secondary education in Hungary." Procedia - Social and Behavioral Sciences (2015), IETC 2014. Accepted. www.elsevier.com/locate/procedia.

[5] S. Booth, "Learning to program: A phenomenographic perspective." Gothenburg, Sweden: Acta Universitatis Gothoburgensis, 1992.
[6] J. M. Case, and R. F. Gunstone, "Metacognitive development as a shift in approach to learning: an in-depth study." Studies in Higher Education, 27(4), 2002, pp. 459–470.
[7] Computer Science Curricula, "The Joint Task Force on Computing Curricula Association for Computing Machinery" (ACM) IEEE Computer Society. 2013, http://dx.doi.org/10.1145/2534860. Retrieved April 18, 2014.
[8] M. Csernoch, and P. Biró, "Digital Competency and Digital Literacy is at Stake." ECER 2014, The Past, the Present and the Future of Educational Research. Porto, 2–5 September 2014. http://www.eera-ecer.de/ecer-programmes/conference/19/contribution/31885/
[9] M. Csernoch, and P. Biró, "Spreadsheet misconceptions, spreadsheet errors." Oktatáskutatás határon innen és túl. HERA Évkönyvek I., ed. Juhász Erika, Kozma Tamás, Publisher: Belvedere Meridionale, Szeged, 2014, pp. 370-395.
[10] M. Csernoch, and P. Biró, "The power in digital literacy and algorithmic skill." International Conference on New Horizons 2014. Paris, France 25-27 June, 2014, Procedia - Social and Behavioral Sciences 2014, INTE 2014. Accepted, www.elsevier.com/locate/procedia.
[11] M. Csernoch, and Gy. Bujdosó, "Quality text editing." Journal of Computer Science and Control Systems. 2/2, 2009, pp. 5–10.
[12] M. Csernoch, "Teaching word processing – the theory behind." Teaching Mathematics and Computer Science. 2009/1. pp. 119–137.
[13] M. Csernoch, "Teaching word processing – the practice." Teaching Mathematics and Computer Science. 8/2, 2010, pp. 247–262.
[14] M. Csernoch, "Clearing Up Misconceptions About Teaching Text Editing", Proceedings of ICERI2011 Conference, ICERI 2011, 4th International Conference of Education, Research and Innovation, 11-14 November 2011, Madrid, ISBN 978-84-615-3324-4.
[15] R. Panko, and S. Aurigemma, "Revising the Panko-Halverson taxonomy of spreadsheet errors." Decision Support Systems 49, 2, 2010, pp. 235–244.
[16] S. G. Powell, K. R. Baker, and B. Lawson, "Impact of errors in operational spreadsheets." Decision Support Systems, 47(2), 2009, pp. 126–132.
[17] R. Lister, B. Simon, E. Thompson, J. L. Whalley, and C. Prasad, "Not seeing the forest for the trees: novice programmers and the SOLO taxonomy", in Proceedings of the 11th annual SIGCSE conference on Innovation and technology in computer science education, New York, NY, USA, 2006, pp. 118–122.
[18] A. Van Deursen, and J. Van Dijk, "CTRL ALT DELETE. Lost productivity due to IT problems and inadequate computer skills in the workplace." 2004, Enschede: Universiteit Twente. Retrieved May 18, 2014. http://www.ecdl.ch/fileadmin/ECDL/CH/Dokumente/Studie_CTRL-ALT-DELETE-en.pdf.
[19] P. Warren, "Learning to program: spreadsheets, scripting and HCI", in Proceedings of the Sixth Australasian Conference on Computing Education – vol. 30, Darlinghurst, Australia, 2004, pp. 327–333.
[20] J. M. Wing, "Computational Thinking." Communications of the ACM, March 2006/Vol. 49, No. 3.
[21] M. Gove, "Michael Gove speech at the BETT Show 2012". Published 13 January 2012. Retrieved 15 July 2014 from https://www.gov.uk/government/speeches/michael-gove-speech-at-the-bett-show-2012.

Mária Csernoch was born in 1963 in Szentes, Hungary. She received her MSc in 1986 at Kossuth Lajos University, PhD in 2006 and Habil. in 2012. Currently she works as associate professor at the University of Debrecen, Faculty of Informatics in Hungary.

Her research interests are didactics of Informatics – specialized in developing algorithmic skills, computational thinking, and teaching programming languages –, computational linguistics and computer aided language teaching and learning.

Piroska Biró was born in 1983 in Ditro, Romania. She received her M.Sc. degree in Computational Mathematics in University of Babes-Bolyai, Faculty of Mathematics-Informatics. Currently she works as assistant lecturer in the University of Debrecen, Faculty of Informatics in Hungary.

Her research interests are didactics of Informatics, algorithmic thinking, high level programming languages and computer aided education.