Title of the Article: Automatic Reading of Educational Texts for Vision Impaired Students
The Name of the author(s): Jindřich Matoušek, Michal Campr, Zdeněk Hanzlíček, Martin Grůber
Affiliation (University, City, Country): Faculty of Applied Sciences, Dept. of Cybernetics, University of West Bohemia, Plzeň, Czech Republic
Keywords: education support, vision impaired students, automatic reading of mathematical formulas, text-to-speech
E-mail: jmatouse@kky.zcu.cz, mcampr@students.zcu.cz, zhanzlic@kky.zcu.cz, gruber@kky.zcu.cz

1 Introduction

The paper presents the current state of the ongoing project "Automatic Reading of Educational Texts for Vision Impaired Students" (ARET, `http://aret.zcu.cz`). The project aims at an innovation and enhancement of schooling of vision impaired students and also at a facilitation of their self education. The project is solved at the University of West Bohemia (UWB), Department of Cybernetics, in cooperation with the Primary School and the Kindergarten for the vision impaired in Pilsen. Within the project, a specially designed system for automatic reading of educational texts for vision impaired students is being developed. Teachers use the system for a preparation, management and administration of educational texts. In order to facilitate the maintenance of the final system, client-server architecture was chosen. The system is based on the Symfony PHP framework. The educational texts are available to students via system's front-end; the texts are read aloud by means of computer speech synthesis (more specifically, using a text-to-speech engine developed at the UWB). The access to the system is highly configurable; different rights for editors, students and others can be easily set up. With respect to both the purposes of the ongoing project and the main partner (the primary school), the educational texts concentrate on Mathematics and Physics (ISCED 2 level). Hence, the presence of mathematical and physical formulas has to be dealt with, both in the phase of a creation of educational texts and in the phase of automatic reading of the texts. Automatic processing of the formulas, including a transcription of their symbolic notations to corresponding word forms and automatic reading of resulting texts, presents a challenge to the current text-to-speech technology. Despite the current focus on primary-school subjects, the system is capable of reading any texts, including more advanced texts like tertiary level of mathematics, etc.

The TTS module of the developed system could be viewed as an alternative to a screen reader. Screen reader is a more general software, also based on text-to-speech technology, which can read any text information on a screen (PC monitor, TV etc.). Evidently, any screen reader could be utilized to read the project-specific educational texts displayed on a screen as well. On the other hand, the developed system is a more specialized application tailored to the reading of educational texts related to the ARET project and to the designed web-based application. As Mathematics and Physics are included in the texts, special approaches to the processing of mathematical and physical formulas are developed within the system framework exploiting the extra information about mathematical formulas from the system's backend.

Other similar projects for reading technical documents or mathematical formulas also exists. The problem of reading mathematics has been already solved, e.g. in the system

AsTeR (Audio System for Technical Readings) [Raman, 1994] or in the system AudioMath developed at Porto University [Ferreira, 2004]. For the Czech language, the Lambda editor (in which, besides the audio synthesis, the Braille system is also supported) was created at Masaryk university (`http://www.teiresias.muni.cz/czbraille8/`). Within the presented project ARET, a new system for reading mathematical formulas is being developed.
The paper is organized as follows. In Section 2, the framework of the developed system for automatic reading of educational texts is presented, including both system's backend (an administrative tool for creating and modifying educational texts) and system's frontend (a public web interface for displaying and reading educational texts). The text-to-speech technology used for reading the texts aloud is briefly described in Section 3. Special issues related to the solved project and the text-to-speech technology are depicted in Section 4. Finally, conclusions are drawn in Section 5.

2 System Framework

The developed application uses many various technologies and techniques widely used among web developers and web designers. Several programming languages have been employed during the development of the final application, such as PHP, JavaScript, JQuery (JavaScript library), Java and Python. The final application is running on an open-source HTTP server *Apache* with *MySQL* database system.
The core of the system is based on *Symfony*, an open-source web application framework for PHP projects. The PHP programming language has also been used for developing other essential web services, i.e. for the implementation of text-to-speech (TTS), TEX-to-image and MathML-to-text conversions (all of them will be discussed in the following sections). The other languages has been chosen due to their specific qualities and used for different purposes, such as parsing the HTML document to extract a text for reading (JQuery), providing a tool for creating mathematical formulas (Java applet) and creating scripts for text conversions (Python).

2.1 Symfony framework

The Symfony framework has been chosen because it meets the essential requirements, such us a simple usage, well arranged source files for future development, and a good performance of the system.
Symfony is very easy to install on any configuration, so applications can be developed on OS Windows and also run on UNIX-like systems. It is also compatible with various database systems. It is aimed at building robust applications with full control over the configuration and customization. This makes it easy to import third-party libraries and plugins. Symfony is also equipped with additional tools for testing, debugging and documenting. Moreover, the Symfony project benefits from an active open-source community, where many guides, tools and plugins can be found. The application takes advantage of several plugins to handle tasks typical for web applications. For example user security and permission management (permissions are highly configurable — different rights for editors (teachers), users (students) and others (public visitors) can be easily set up), validating forms etc. For database management, *Doctrine* is used, which is a PHP ORM (Object Relational Mapper) for PHP. One of its key features is the ability to write database queries in an object oriented SQL-dialect called DQL (Doctrine Query Language).

2.2 System architecture

The system is being developed as a web application; therefore it is logically divided into two separate sections: frontend and backend. Frontend serves as a public interface for viewing various educational texts (arranged as lessons, or topics) and, at the same time, reading them. Backend, on the other hand, is an administrative interface, where the lessons can be created or modified. Each of these sections makes use of different parts of the whole system shown in Figure 1.

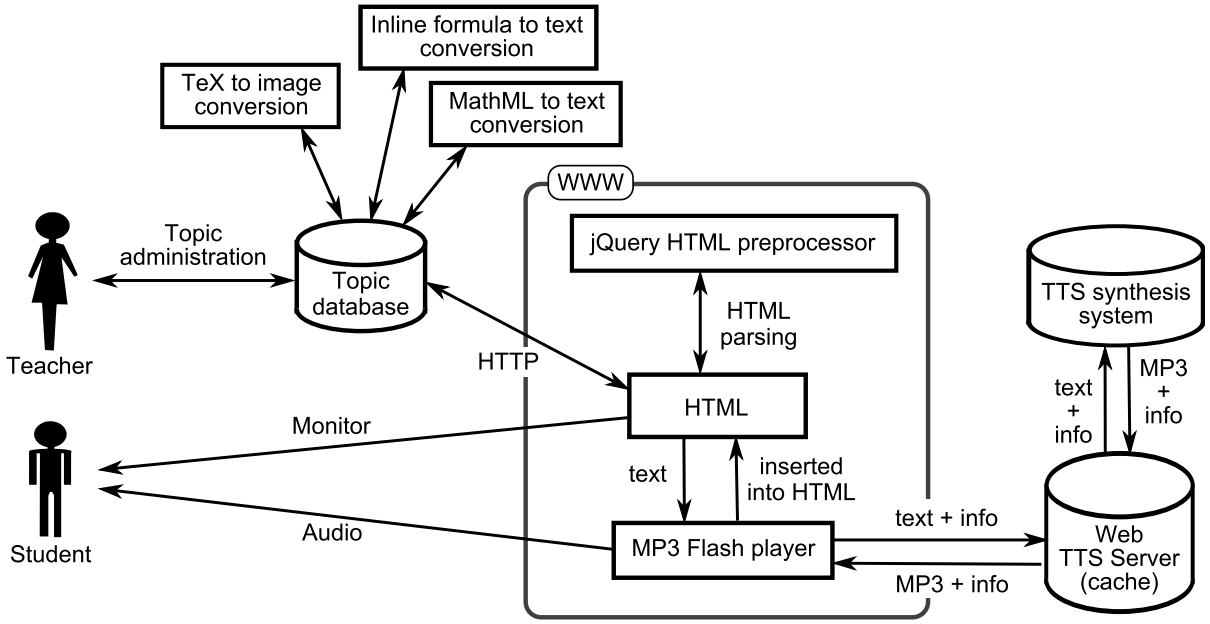

Figure 1: System diagram.

2.3 Backend

System's backend serves as an administrative tool for creating and modifying topics. The content of a topic is represented by a valid HTML document where all information (including formula transcriptions) needed for reading is stored. All these HTML documents are stored in the main database (Figure 1 – Topic DB). Teachers have a direct access to this database through the text editor (Figure 2 on the left), where they can add or edit the contents. There are some project-specific features added to the text editor for inserting *HTML templates* and formulas. The templates can be used to clarify the meaning of a particular fragment of the document, for example the *Warning* template is for highlighting a crucial information to which the students should pay more attention, or the *Example* template is used to display various examples. Currently, five templates are supported: Definition, Warning, Note, Example, and Solution. As templates help to keep the topic contents well arranged and can enable some extra features in future versions (e.g. template-dependent voices), teachers, the editors of the topics, are encouraged to use the templates.

Upon finishing the editing and pressing the *Save* button, additional scripts are run to modify the content — adding formula transcriptions or cleaning up the HTML code.

There are two different ways for inserting mathematical formulas into the document. The first option uses the *inline formula*, which means marking a part of a text by a SPAN

tag assigned with a specific class. This text should represent a simple formula, which can be written in one line of text (for example $x + 1$). This text is then processed by two scripts for creating a text transcription of the formula for reading and a better formatted notation for viewing.
The second option uses the *DragMath* formula editor. This modified Java applet can be used to insert more complex formulas into the document. The applet is able to generate multiple different representations of given formula, for example in MathML or TEX notation. The TEX notation is used to generate an image of the formula and MathML is used for generating the corresponding text for reading. In the HTML code, the formula is then represented by an image with a proper transcription stored in the *alt* attribute.

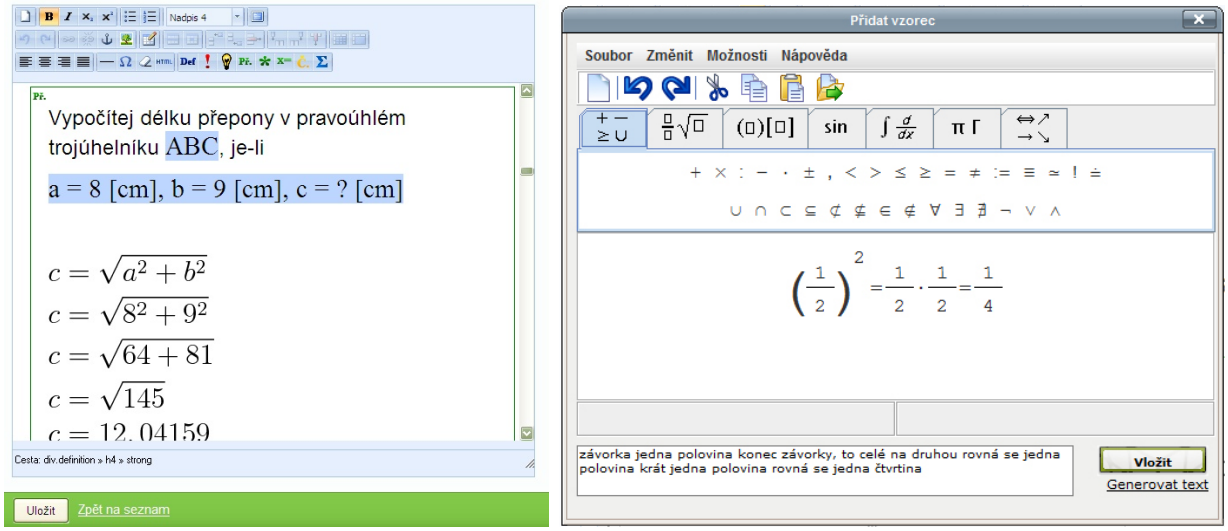

Figure 2: Administration — text editor and formula editor.

2.4 Frontend

System's frontend is a public web interface where the topics are displayed and read aloud to the students. Students need to be properly logged in to gain access to published topics. Various topics can be accessed through a menu in the left column of the web page (see Figure 3). Before displaying the web page, the content needs to be adjusted for proper viewing and reading, a proper formatted notation of inline formulas has to be inserted and the document has to be parsed to extract the text for reading. Both of these actions are accomplished using the JQuery library.
After parsing the document, an array of text segments is created and used as a playlist for MP3 player. In this project, an open-source JavaScript player *JPlayer* is used, which employs Adobe Flash for playing MP3s. This player is inserted into the HTML document and is displayed on the edge of the web page. It is also possible to control the player by predefined keyboard shortcuts. After creating the playlist, the player starts sending requests to the *Web TTS server*. Server's main function is to store the generated audio files. If the server receives a request for MP3, it tries to look for the proper file or forwards the request further to the TTS generator which generates the appropriate audio file (i.e. an MP3 with speech) from the input text using a text-to-speech technology described further in Section 3. An URL pointing to the file located on the cache server is then returned to the player and thus it can be read to the user.

In the playlist, additional information about the meaning of the text segments is also stored, i.e. which text segment is a part of a paragraph or which is a heading. With these information, the player can either jump to the next or previous paragraph or to the next or previous heading. The text, which is currently being read, is also highlighted. An example can be seen in Figure 3.

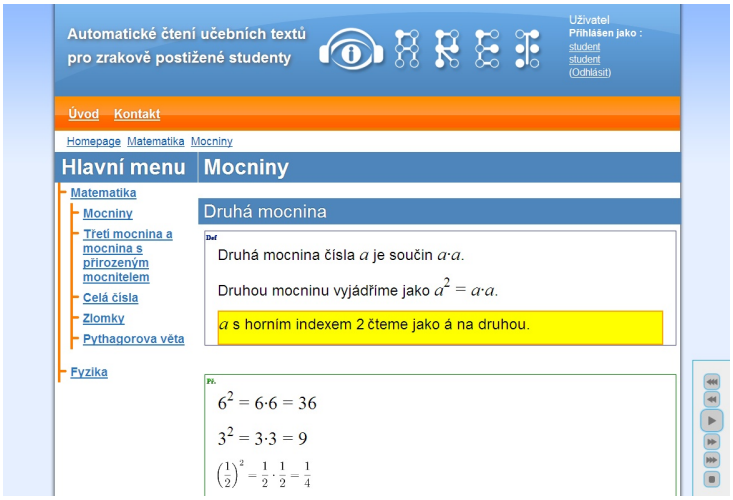

Figure 3: Web page with TTS — text which is currently being read is highlighted by the yellow colour; MP3 player is in the bottom right corner of the web page.

3 Text-to-Speech Technology

For the automatic reading of educational texts in system's frontend, text-to-speech (TTS) technology was utilized. The task of a TTS system is to convert an arbitrary input plain text to the corresponding speech. In our case, a Czech text-to-speech (TTS) system ARTIC (Artificial Talker in Czech) [Matoušek, 2006] developed at the Department of Cybernetics, Faculty of Applied Sciences, University of West Bohemia in cooperation with a firm SpeechTech was adapted.

ARTIC applies a corpus–based concatenative speech synthesis method. Based on a carefully designed speech corpus (a collection of a large number of utterances annotated on orthographic, phonetic and prosodic levels), statistical approach (employing hidden Markov models, HMMs) was employed to perform an automatic phonetic segmentation of the source speech corpus into phones. Based on this segmentation, boundaries between diphones, the basic speech units used in the ARTIC system, were located. As a result, acoustic unit inventory (AUI), the source speech corpus indexed with diphones and prosodic structures, was built. Beside speech waveforms, glottal signals were also recorded using an electroglottograph and used as input signals to glottal pulses (pitch-marks) detection algorithm. Pitch-marks are used as consistent concatenation points during speech synthesis.

During runtime speech synthesis, phonetic and prosodic aspects of an input text are estimated first. Ideally, input text is a subject of a thorough analysis and processing. Due to a complexity of such a task, current text processing in the ARTIC system is somewhat simplified to four main steps:

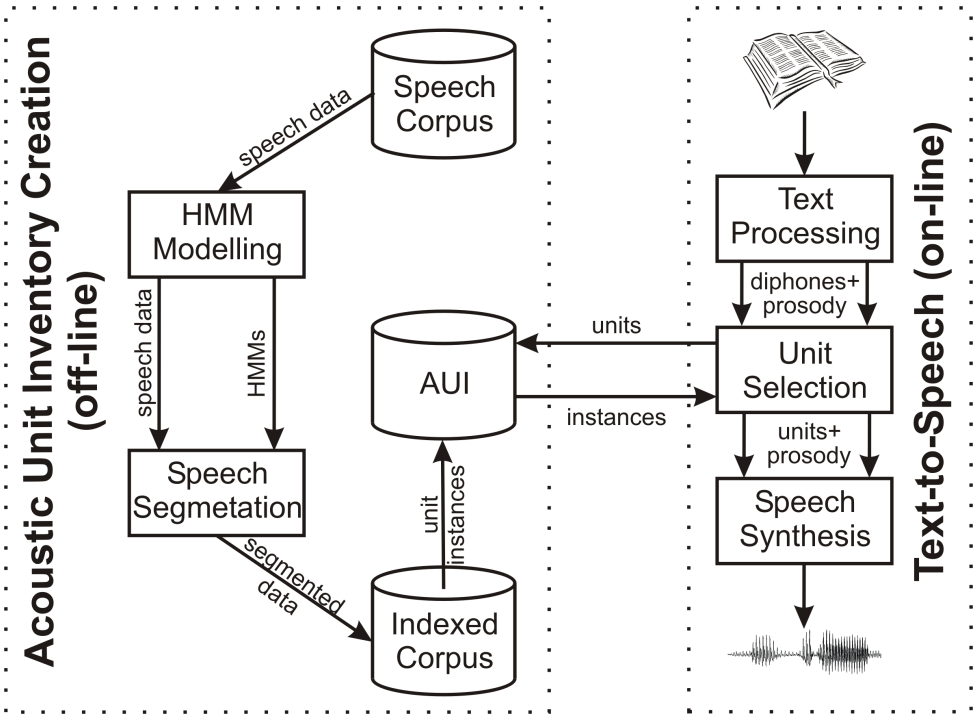

Figure 4: A schematic view of the ARTIC TTS system.

- text normalization of "non-standard" words (digits, abbreviations, acronyms, etc.) – see Section 4.2 for text normalization in the ARET project;
- detailed rule-based phonetic transcription, including pronunciation dictionary of "exceptional" words (mostly foreign words);
- phones-to-diphones conversion;
- prosodic description in terms of prosodic symbols (prosodic clauses, phrases, prosodemes, etc.) using prosodic phrase grammar [Romportl, 2008].

Within the scope of our project, mathematical and physical formulas are treated specially and described with the inline formulas or MathML codes. Such specially marked formulas could be then processed and converted to words (see Section 4.1).
Prosodic analysis includes punctuation-driven sentence clause detection, rule-based word stress detection and symbolic prosodic description [Romportl, 2008]. Symbolic features based on a prosodic phrase grammar, like prosodic sentence, prosodic clause, prosodic phrase, prosodic word, and prosodeme, were used to describe prosodic characteristics and to express prosodic structure of to-be-synthesized texts.
The resulting speech is generated by a *unit-selection* algorithm. Its principle is to smoothly concatenate (according to *join cost*) speech segments (diphones in our case), extracted from natural utterances using the automatically segmented boundaries, from large speech unit inventories according to phonetic and prosodic criteria (target cost) imposed by the synthesized utterance. As there are usually many instances of each speech segment, there is a need to select the optimal (with respect to both target and join costs) instances dynamically during synthesis run-time (using a unit selection technique). To calculate the *target cost*, a prosodic structure of the to-be-synthesized utterance is estimated, and a comparison between prosodic symbolic features (plus some positional features, like position of a diphone in a prosodic word, and contextual factors like immediate left and right

phone) in the utterance and in the unit inventory is carried out. Join cost is evaluated as a distance between spectral features and pitch around the concatenation point of two potentially neighbouring speech units. After selecting the optimal sequence of (diphone) speech segments, neither prosodic nor spectral modifications are made in the ARTIC system except for simple smoothing at concatenation points. To cope with high CPU power and memory cost typical for unit-selection systems, a computational optimization was carried out as described in [Tihelka, 2010]. A schematic view of the ARTIC TTS system is shown in Figure 4. More details about text-to-speech synthesis can be found e.g. in [Psutka, 2006].

4 Project-Specific Issues

Text processing is an important part of a TTS system. Generally, text processing in a TTS system depends on the type of texts that are likely to appear at the input of the system. In the ARET project, educational texts (currently the texts of Mathematics and Physics at ISCED 2 level) are expected as an input of the TTS system. In this section, text processing issues related to the ARET project are described.

4.1 Automatic Reading of Formulas

Reading of mathematical formulas in the Czech language is a very complex task. Especially if the problem is supposed to be solved generally, i.e. there is no limitation for the complexity of the equation structure. Indeed, any final system will be naturally limited by the definition of expected mathematical operations, types of operands, etc. However, the system should be simply extensible by additional definition of reading rules, e.g. for new operators.
As mentioned in Section 2, two different representation of mathematical formulas are employed in our system. Simple formulas with a linear structure can be written and stored as a simple text. For the creation of more complex mathematical expressions, the special editor DragMath is employed and their structure is represented by using an MathML format. In both cases, thanks to a special syntax or marking of the formulas in the HTML code, no detection of formulas is needed.
The problem of reading formulas can be divided into several steps:

- transcription of a formula to its corresponding word form
  - hierarchical decomposition of a MathML code or text representing the formula
  - selecting suitable transcription rules for particular operations
  - applying the selected rules (including inflection of particular operands)
- text-to-speech synthesis
  - after transcribing formulas to their corresponding word forms, the formulas are handled as any other text, and, as such, they are sent at the input of a TTS system (or, a web TTS server, respectively)

For each mathematical operation, several transcription rules can be defined. They differ by their activation conditions, i.e. in various mathematical contexts, for various values or

types of operands, different transcription rules can be selected. For most operators, one basic rule and several additional rules for exceptional cases are defined.
Each transcription rule contains a text template for the resulting expression together with the corresponding grammatical form for each operand (case, number, gender, cardinal or ordinal form etc.)
A simple example of one transcription rule for power operator (in YAML notation; YAML is a recursive acronym for "YAML Ain't Markup Language"):

POWER:
```
- condition: { operand_2_type: [ number, variable ] }
  operands:
  - { type: cardinal, case: 1, number: S, gender: F }
  - { type: ordinal, case: 4, number: S, gender: F }
  template: '{operand_1} na {operand_2}'
  expr_type: expression
```

It is quite easy to define a new set of transcription rules or extend an existing one with rules for new mathematical operations or with additional rules for some rare linguistic exceptions.
The conversion of formulas to text is shown in Figure 1 in blocks "Inline formula to text conversion" and "MathML to text conversion".

4.2 Text Processing

In Section 4.1, the processing of specially marked formulas was described. In this section, analysis and processing of all input texts (i.e. the contents of web pages with topic-arranged educational texts) are briefly described. The whole process of text processing consists of several steps shown in Figure 5 and further described in the following subsections.

4.2.1 Text Filtering

Since the texts are coming from HTML web pages, some unwanted "garbage" characters present in the HTML codes can occur. These characters have to be removed or replaced before further processing. The list of garbage characters includes but is not limited to HTML tags, HTML entity characters, quotation marks, etc.

4.2.2 Text Normalization

Normalization is a process of converting "non-standard" words (digits, abbreviations, acronyms, etc.) to their extended and grammatically correct forms.
At first, the non-standard words have to be detected in the input text. So far, we are able to detect all numbers, most of symbols for physical units and currencies and also some abbreviations that are used in a common text.
The next step is to determine the grammatically correct form of the detected words. This is one of the most difficult tasks for the Czech language, since Czech is very flexible and a single word can have many various forms. The form of a particular word in a particular sentence depends on the syntax and the meaning of the entire sentence. Thus, an exact determination of the correct form is not possible without an extensive semantic

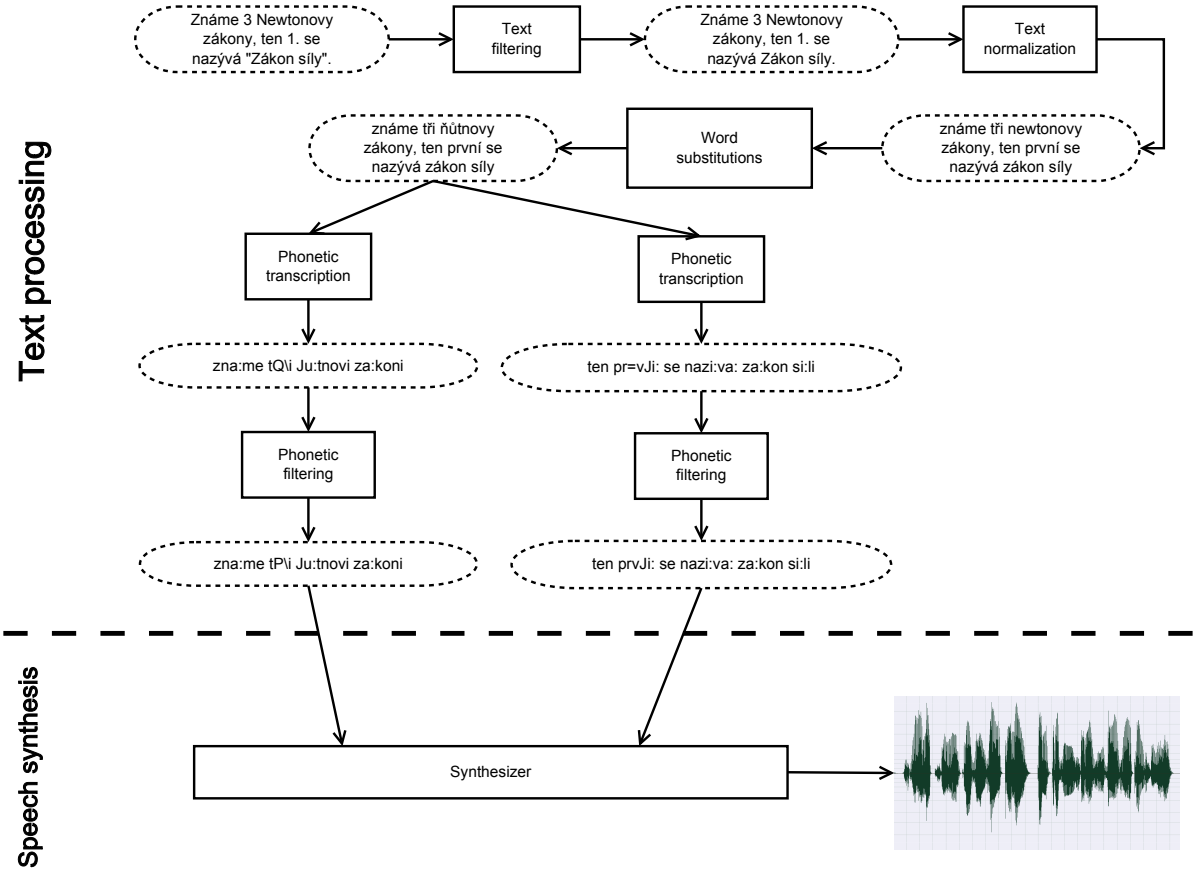

Figure 5: A block diagram of text processing within the ARET project (the upper part).

and syntactic analysis. Therefore, an estimator is supposed to be used for this purpose. At present, we are using TnT tagger, a very efficient statistical part-of-speech tagger that has been trained on a large Czech corpus already tagged by morphological tags beforehand. However, this tagger still works with some errors and this process needs to be improved.

4.2.3 Word Substitutions

Words that cannot be transcribed using standard Czech phonetic transcription rules have to be processed in some other way. For this purpose, we use "dictionary-like" system in which a single word can be replaced with a corresponding "phonetic-friendly" transcription of the word (or even with a sequence of phonetic-friendly transcriptions). This is useful for foreign words, names, proper nouns or abbreviations that are not caught during the preceeding normalization process. This dictionary can be easily modified, so adding new exceptions is very simple. Support for inserting new substitutions was also incorporated to system's backend where an editor of a topic can mark a word as a "pronunciation exception" and, using a special tag "Read-as", can write the proper pronunciation of the word.

4.2.4 Phonetic Transcription

The transcription is a process in which an input text in an orthographic form is transformed into a form represented by phones. As mentioned in Section 3, this process in our system is rule-based since the conversion is almost always unambiguous in the Czech

language. The pronunciation exceptions (foreign words, etc.) are handled as described in Section 4.2.3.

4.2.5 Phonetic Filtering

The transcribed text can contain some characters that are not supported by the speech synthesis engine. Therefore, a final replacement of these characters is needed. At present, all unsupported characters are omitted.

5 Conclusion and Future Work

In the paper, the current state of the ongoing project "Automatic Reading of Educational Texts for Vision Impaired Students" (ARET, `http://aret.zcu.cz`) was presented. Since the project is solved at the University of West Bohemia together with the Primary School and the Kindergarten for the vision impaired in Pilsen, it currently focuses on primary-school students of Mathematics and Physics (ISCED 2 level). Nevertheless, the system framework has been designed to be general and flexible enough to cover other kinds of educational texts (or topics), including more advanced texts like tertiary level of mathematics, etc.
Although the ARET project is still being worked on, the first educational texts are already available on `http://ucebnice.zcu.cz`. There are some sample topics available to public, other topics are available to the students of the partner primary school upon logging in.
Future work will be focused on three main areas:

- **Topics.** The number of topics will be continuously increasing in order to cover the intended goal of the ARET project (subjects of Mathematics and Physics at ISCED 2 level).
- **System functionality.** System functionality (both the backend and the frontend) is also planned to be enhanced. For instance, automatic reading errors caused by the TTS system will be corrected, other rules for reading formulas will be added, etc. We also plan to personalize the system for each user by allowing him/her to change the layout of web pages with topics (e.g. colours of fonts, templates, etc.) or to change the voice used for reading.
- **Compatibility with other tools for the vision impaired.** We will also try, in cooperation with the partner primary school, to make the developed system more compatible with other tools and systems that vision impaired use. For instance, we will optimize keyboard shortcuts. Moreover, problems with undesirable concurrent reading of educational texts by the embedded TTS system and by a screen reader a student is used to utilize for reading other information will be solved.

6 Acknowledgements

This project is co-funded by the European Social Fund and the State Budget of the Czech Republic.

References

Books

PSUTKA, Josef – MÜLLER, Luděk – MATOUŠEK, Jindřich – RADOVÁ, Vlasta. *Mluvíme s počítačem česky.* 2006. Academia; Praha. ISBN 80-200-1309-1.

RAMAN, T. V. *Audio System for Technical Readings.* Ph.D. thesis 1994. Cornell University; New York, NY.

Articles

FERREIRA, Helder – FREITAS, Diamantino. Enhancing the Accessibility of Mathematics for Blind People: The AudioMath Project. In *Lecture Notes in Computer Science*, vol. 3118. Berlin, Heidelberg: Springer, 2004. s. 678–685. ISSN 0302–9743.

MATOUŠEK, Jindřich – TIHELKA, Daniel – ROMPORTL, Jan. Current State of Czech Text-to-Speech System ARTIC. In *Lecture Notes in Computer Science*, vol. LNAI 4188. Berlin, Heidelberg: Springer, 2006. s. 439–446. ISSN 0302–9743.

ROMPORTL, Jan. Prosodic Phrases and Semantic Accents in Speech Corpus for Czech TTS Synthesis. In *Lecture Notes in Computer Science*, vol. LNAI 5246. Berlin, Heidelberg: Springer, 2008. s. 493–500. ISSN 0302–9743.

TIHELKA, Daniel – KALA, Jiří – MATOUŠEK, Jindřich. Enhancements of Viterbi Search for Fast Unit Selection Synthesis. In *Interspeech 2010: proceedings of 11th Annual Conference of the International Speech Communication Association 26-30 September, 2010 Makuhari, Japan.* 2010. s. 261–272.

Electronic articles

Návrh české osmibodové normy, matematický editor Lambda (informační portál) [online]. [cit. 2011-01-18].
Available in URL <http://www.teiresias.muni.cz/czbraille8/>.