Insert here your thesis’ task.

Czech Technical University in Prague

Faculty of Information Technology

Department of theoretical computer science

Master's thesis

Admission procedure Automatic processing of applications for master's study program

Bc. Ján Ondrušek

Supervisor: Ing. Tomáš Kadlec

6th June 2012

Acknowledgements

I would like to thank my family and friends for support during writing this thesis.

Declaration

Citation of this thesis

Ján Ondrušek. *Admission procedure*
Automatic processing of applications
for master's study program: Master's thesis. Czech Republic: Czech Technical University in Prague, Faculty of Information Technology, 2012.

Abstract

Primary aim of this thesis is to analyse Conditions for admission and Dean's directive for admission process to master's study programme at CTU FIT. Implement RESTful API, which exposes backend functionality for admission processing using Business Process Management.

Keywords Admission procedure, RESTful API, BPM, jBPM, Spring, Spring Roo

Abstrakt

Primárnym cieľom tejto diplomovej práce je analyzovať Řád přijímacího řízení ČVUT a Směrnici děkana pro přijímací řízení na ČVUT Fakultě informačních technologií. Implementovať RESTful API, ktoré vystaví funkcionalitu backendu pre prijímací proces s použitím Business Process Management stroja.

Klíčová slova Spracovanie prihlášok, RESTful API, BPM, jBPM, Spring, Spring Roo

Contents

List of Figures

List of Tables

Prologue

Motivation and objectives

Every year, hundreds of high school graduates apply for studies at Czech Technical University, Faculty of Informatics. This raises certain requirements, including managing, storing, analysing and processing of all these applications. Each application has its own life cycle, which begins with filling out an on-line form and continues through various steps which an applicant has to pass. The life cycle ends when a decision of acceptance is delivered to the applicant and he either enrolls in the studies or not.

We live in the world of new era of the Internet. Everything goes on-line, web and the latest trend - everything goes mobile. People want things to happen very quickly. They want to access all the information fast, now.

Students and applicants are no different. They expect from this prestigious University, especially from Faculty of Informatics, the most modern and useful gadgets when it comes to software and web.

How do things work now

Currently all applications are processed rather manually. Many man days of administrative work are consumed during the process. Although an electronic form is filled in and submitted by an applicant, the rest of actions almost exclusively fall into the hands of Study Department staff. Some of the work is handled by simple scripts or other utilities. The question is: Why don't we do most of the work automatically?

This work is monotonous and can even lead to men's frustration.

What should be achieved - the goals

Courses at Faculty of informatics teach its students to handle various programming languages, web technologies and techniques. We all know what to expect from a working web application and good looking one is a bonus. This is why knowledge of faculty's students should be used for good of their successors. Fast, reliable, informative and functional system will make them feel more comfortable and perhaps could even some precious time.

Ideal state would be to accept on-line application and automatically generate invitations for applicants, that should attend a test. After the test, process all results and generate decision of acceptance letter for all who passed the test or are accepted without it. The only manual interventions that will remain is to accept apology, appeal and insert the letters into the envelopes.

Pragmatically goals of this thesis could be summarized as follows:

- familiarise with RESTful best practices, patterns and anti-patterns
- familiarise with BPM with main focus on jBPM
- implement RESTful API (backend) according to functionality requested by Android and Web UI teams
- implement admission processing using Java and jBPM processing machine
- explore new and modern Java (JEE) technologies
- follow modern development methodologies
- perform tests during and after development
- use exclusively Open Source software and tools

Let's make things better

Taking the above written into account, this might be a good idea for a master's or bachelor's thesis. However if we want to use all available technologies that have become popular in past years and automatize the majority of admission processing, it turns out to be a very complex project. So why not to create several teams and split necessary work into multiple, both bachelor's and master's, thesis?

This is how project Přiříz was born. It includes web interface for both students and Study Department staff, native Android application and RESTful API with BPM processing machine, which is the subject of my master's thesis.

Structure of this work

Basically, I could divide my work into these main parts, which are then further split into chapters:

- Theoretical introduction

- Analytical part
- Implementation and unit testing
- Integration and regression testing
- Results and conclusion

Appendices at the end of the document are referred directly from the text within the chapters. Smaller figures, tables or other objects are put directly into the content.

CHAPTER 1

RESTful API with JAX-RS

Nowadays, Internet consumers demand fast growth of various services and integration of their favourite ones. As en example I can point out synchronization of contact list between very popular social networks, e-mail providers and phone contact lists.

Other example may be growing amount of `mashups`[1] and uncountable number of `startups`[2], who often provide RESTful or different type of public API.

[1]Applications that are created via combination of multiple different services. Such application, almost exclusively web based, can be created very quickly by consuming several APIs. Not necessarily from the same provider.

[2]Constantly rising amount of web applications, that focus on fast growth of attracted users. They offer various services, which are often very innovative and experimental. One successful example is popular social network and my favorite information channel - Twitter.

CHAPTER 2

Implementation and testing

2.1 Details of realised tests

The scaliness of implemented algorithms were tested on chosen files from Calgary and Canterbury corpus too. The framework to scale the files is different of the testing one. Each file is equally split to 1 000 parts (files with number of lines less than 1 000 are split to 100 parts) by number of lines—the n th part consists of $\frac{n}{1000}$ ($\frac{n}{100}$) lines. Each test runs only 100–10 times (depending up the n—the size of compressed data) because of time complexity. This cycle runs 10–100 times (10 was chosen for this tests) and the minimum time is taken. The file splitting by the number of lines was chosen because of the character of algorithms (word-based)—the splitting by the block of the same size is not so predicative.

There are parameters of the computer used for tests shown in Table 2.1.

2.2 Integers distribution and encoding

The integers encoding of indexes of phrases from the word (non-word) dictionary is possible cause of the only average results of compression ratio of implemented algorithms. The decision is to get the distribution of indexes during the encoding process (and decoding process too). The length of indexes

Table 2.1: Testing computer parameters

Part	Description
CPU	2.2 GHz AMD Athlon(tm) 64 Processor 3200+
MEM	2.5 GB
OS	x86_64 GNU/Linux Fedora release 7 (Moonshine)

located in shown graphs is only hypothetical—the binary code with minimal length.

The graphs of index distribution also shows the differencies between the algorithms with sorted dictionaries (*WLZWS* and *WLZWES*) and the algorithms with unsorted dictionaries (*WLZW* and *WLZWE*). The most frequently used phrases are moved to the front of dictionary in algorithms with sorted dictionary so they get lower indexes. This feature is demonstrated by the growth of number of indexes at the beginning of the distribution. The compression process of algorithms with sorted dictionaries becomes more efficient when the code with variable length of code words (Fibonacci code) is used but the compression efficiency is supposed to be the same at the transition from the *WLZWE2* algorithm to the *WLZWES2* algorithm—the encoding by block code.

Conclusion

The word-based dictionary data compression algorithms (a part of lossless data compression) are the subject of this thesis. The lossless data compression is a very important field of research because the data compression allows to reduce the amount of space needed to store data.

The background of a data compression field was presented in Chapter **??**. There are basic notions and definitions followed by the description of character-based dictionary algorithms. The word-based dictionary compression methods were investigated and discussed at the end of this chapter too.

There is the investigation of index distribution of tested files in Section 2.2. It led to the new modification of semi-adaptive word-based Lempel Ziv Welch (LZW) algorithm—*WLZWE2*. The compression efficiency of this algorithm applied to the large files is better than the other implemented algorithms. However, the compression efficiency of *WLZWE2* algorithm is much worse when it is applied to the small files. The experiments with *WLZWE2* and *WLZWES2* algorithms confirm the assumption from Section 2.2—the compression efficiency of version with unsorted dictionaries (*WLZWE2*) is analogous to version with sorted ones (*WLZWES2*).

The testing of memory used during compression and/or decompression process is one of the possibilities of further research. The experiments with files of greater size or multilingual files could be also good opportunity to gain new improvements of algorithms. The static part of dictionaries could improve the compression efficiency too.

The implemented methods achieve fairly good compression ratio (25–30% at large files) with acceptable compression and decompression time. There are possibilities of further improvements especially at semi-adaptive methods. However, the gain of these improvements is not good enough to top the compression efficiency of other lossless data compression methods (context methods from PPM family). The results of implemented algorithms were not as good as it was expected but the work on this thesis showed new ways of possible further research—word-based version of grammar-based compression algorithms and another possibilities in the field of word-based context methods of data compression.

The Gnuplot 4.2 utility was very useful for generation of graphs in this thesis. There was the drawing editor Ipe 6.0 used for figures creation.

APPENDIX A

Content of CD

readme.txt	- the file with CD content description
data/	- the data files directory
graphs/	- the directory of graphs of experiments
*.eps	- the B/W graphs in PS format
*.png	- the color graphs in PNG format
*.dat	- the graphs data files
exe/	- the directory with executable WBDCM program
wbdcm	- the WBDCM program executable (UNIX)
wbdcm.exe	- the WBDCM program executable (MS Windows)
src/	- the directory of source codes
wbdcm/	- the directory of WBDCM program
Makefile	- the makefile of WBDCM program (UNIX)
thesis/	- the directory of LaTeX source codes of the thesis
figures/	- the thesis figures directory
*.eps	- the figures in PS format
*.pdf	- the figures in PDF format
*.tex	- the LaTeX source code files of the thesis
text/	- the thesis directory
thesis.pdf	- the Diploma thesis in PDF format
thesis.ps	- the Diploma thesis in PS format