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

7th June 2012

Acknowledgements

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

Declaration

I hereby declare that I have completed this thesis independently and that I have listed all the literature and publications used.
I have no objection to usage of this work in compliance with the act 60 no. 121/2000 (copyright law), and with the rights connected with the copyright act included the changes in the act.

In Prague 7th June 2012 .....

Czech Technical University in Prague
Faculty of Information Technology

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 suc-

cessors. 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 Business process management (BPM) with main focus on jBPM
- implement RESTful Application Programming Interface (API) (back-end) 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 is covered by chapters 1 and 2. Following parts will largely draw on the information contained here.
- Analytical part talks about architecture of the whole ecosystem with main focus on RESTful API. Describes technologies used, methodologies applied and tools commonly used during development and testing phases.
- 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.

1.1 Talking about REST, what is it?

REpresentational State Transfer (REST) or RESTful programming is no official standard and there are no official guidelines or rules for it. So what is it then? It is and architectural and programming style for Web. Lots of text has been written about it during past years and describing the whole idea of REST is out of scope of this master's thesis. I can however try to point out the most significant, important and basically, what I personally managed to adopt.

There are several architectural principles, that one should keep in mind when thinking of REST [**?**, p. 3]:

- Addressable resources The key abstraction of information and data in REST is a resource, and each re- source must be addressable via a Uniform Resource Identifier (URI). A uniform, constrained interface Use a small set of well-defined methods to manipulate your resources.

[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.

- Representation-oriented You interact with services using representations of that service. A resource refer- enced by one URI can have different formats. Different platforms need different formats. For example, browsers need HTML, JavaScript needs JSON (JavaScript Object Notation), and a Java application may need XML.
- Communicate statelessly Stateless applications are easier to scale. Hypermedia As The Engine Of Application State (HATEOAS) Let your data formats drive state transitions in your applications.

HATEOAS is often understood as a core principle of REST. It carries an idea of resource representation via links and stateless implementation of services.

1.1.1 Back to the roots, HTTP is reborn

1.2 Why not SOAP?

1.3 REST vs. SOAP

1.4 REST, Java and JAX-RS

CHAPTER 2

BPM and jBPM

CHAPTER 3

Chosen technologies

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