Identifying Critical Skills for the Technical Workplace

by

David Rusk
B.Eng., University of Victoria, 2013

An Industrial Project Report Submitted in Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

in the Department of Computer Science

Identifying Critical Skills for the Technical Workplace

by

David Rusk
B.Eng., University of Victoria, 2013

Supervisory Committee

Dr. Margaret-Anne Storey, Co-Supervisor
(Department of Computer Science)

Dr. Kin Fun Li, Co-Supervisor
(Department of Electrical and Computer Engineering)

Supervisory Committee

Dr. Margaret-Anne Storey, Co-Supervisor
(Department of Computer Science)

Dr. Kin Fun Li, Co-Supervisor
(Department of Electrical and Computer Engineering)

ABSTRACT

Traditional university education in computer science and software engineering focuses on teaching fundamental principles rather than developing skills with particular technologies. The motivation for this approach is that if the student has strong fundamental knowledge then they can learn any technology needed on the job. However, it is becoming increasingly common for employers to look for candidates who already have the desired skills, and expect them to need little to no training or ramp-up time on the job. As a result, students must develop skills with different technologies on their own initiative. However, software development involves an overwhelmingly large number of competing platforms which makes it difficult to decide where to focus one's efforts. In this project I developed a system to analyze job postings to identify key skills, and trends in demand for these skills over time. The results are presented for the public in a web application. The job postings used in the project were from the VIATeC job posting site. VIATeC was supportive of the project and provided positive feedback on the usefulness of the analysis done to date.

Contents

List of Tables

List of Figures

ACKNOWLEDGEMENTS

Thank you to everyone who gave me feedback during the development of this project. Thank you to my supervisors, Dr. Margaret-Anne Storey and Dr. Kin Fun Li, for their support. And especially thank you to my Dad for helping me flesh out the idea for this project, and his continued interest and input throughout the development process.

Chapter 1

Introduction

The software development landscape consists of a confusingly large number of competing platforms. Selecting which technologies to learn and use for development is a challenge for software developers, whether they be developers in training or experienced professionals. Likewise, educators face a difficult decision when selecting technologies to incorporate into the curriculum. In most institutions, the process of selecting technologies to teach students is not closely related to industry requirements.

The mismatch between skills that employers need and those that developers obtain from their formal education (cf., Canadian Coalition For Tomorrow's ICT Skills [3]) has serious economic consequences. The goal of this study is to reduce the mismatch by building a software technology analytics system to help developers in training, practicing developers, and educators make informed choices about development technologies. It will analyze job postings and other sources published on the web to model and predict demand for key skill sets within specific professional groups and geographic regions.

Some related work has already been done on this topic. For example, Kanya and Geetha [7] have developed techniques for automatically creating "information extractors" for job postings. The extractors generate structured and searchable databases from the unstructured job information. Machine learning algorithms are then applied to the extracted data to find association rules.

This previous work provides a good foundation, but does not directly solve the stated problem of reducing the skills mismatch between what employers need and what potential employees from their geographic area can provide. Therefore, this research intends to fill this gap by meeting the following research objectives:

1. To define a set of indicators for software development technology usage.
2. To use these indicators to identify key technical skill sets currently in high demand in computer science and engineering jobs for specific geographic regions.
3. To track and identify changes to these required skill sets over time.
4. To present the results to educators, developers and employers through a web-based analytics system.

Chapter 2

Literature Review

This review focuses on the literature concerning the use of machine learning techniques to identify critical skill sets for IT professionals. It is expected that the data will come from online sources such as web-based job postings. This data will be in natural language, and will be unstructured or perhaps semi-structured.

While this is a fairly specific topic, it actually spans many areas of research including information extraction, data mining, text mining, natural language processing, machine learning and even software engineering education.

Jiawei Han and Micheline Kamber's textbook "Data Mining: Concepts and Techniques" [6] provides a good background in data mining, text mining and information retrieval among other topics. According to them, "simply stated, data mining refers to extracting or 'mining' knowledge from large amounts of data." The use of data mining techniques for the purpose of automated discovery of knowledge from unstructured text is known as text mining. Information extraction is the first step in this process during which specific pieces of data are extracted from natural language documents.

Yorke [13] describes university and college graduate employability as the acquisition of the "skills, understandings and personal attributes that makes graduates more likely to gain employment and be successful in their chosen occupations, which benefits themselves, the workforce, the community and the economy." In order to gain these benefits, it is necessary that both students and educators have a true picture of the skill sets which will be relevant and useful in the workplace [12].

The most common way to identify employer requirements is through qualitative methods such as surveys and interviews [12]. However, these are time consuming and often lack consistency. Terblanche [12] suggests introducing quantitative methods by

analyzing online job postings to develop an ontology of employer demands for different positions (the nursing profession was the focus of that study). The author then hopes this ontology will then enable artificial intelligence techniques to be quickly and consistently applied to job advertisements in order to identify employer expectations of skill sets.

Kanya and Geetha [7] developed machine learning techniques for automatically creating "information extractors" for job postings. This allowed them to automatically generate structured and searchable databases with the job information. To do this they developed a text mining framework called DISCOTEX (Discovery from Text EXtraction). DISCOTEX first has an information extraction phase in which they used RAPIER (Robust Automated Production of Information Extraction Rules) [2] and BWI (Boosted Wrapper Induction) [5]. Algorithms such as APRIORI [1] and RIPPER [4] were then applied to the extracted data in order to find association rules. An example of a rule might be that if the job posting requires knowledge of the Perl programming language and HTML, then the work will likely be done in a Linux environment. Interestingly, the DISCOTEX system was able to take these mined rules and use them to improve the accuracy of its information extraction process, in a sort of feedback loop.

Kong et al. [8] describe the implementation of a Python based system for extracting information from various job posting sites in China, and provide a unified search and application interface for these sites. The system can also generate statistics on what job positions are popular. Another system applied to Chinese job sites is described by Zhang and Gu [14]. That system is able to classify job postings into predetermined classes of jobs.

Once industry-required skill sets have been identified they may guide professional development for students and those already in the workforce. Universities may also use the information to ensure their programs offer the best preparation possible for the local workforce. An example in a different field is given by Layton [9] in which he describes how industry required skill sets were integrated into the bioengineering curriculum at the University of Illinois at Chicago. This study aims to perform a similar function by identifying skill sets required in the Victoria IT sector that can inform developers and educators as to where to focus further skills development to increase developers' employability.

Chapter 3

Methodology

This chapter describes the sources of data considered for identifying software development technology usage. Initially we investigated open source projects on GitHub belonging to Victoria developers. This focused on programming language usage by local developers. However, looking at open source projects does not necessarily show the skills that are commonly used in proprietary software development projects. To study that, we looked at local job postings.

3.1 GitHub

GitHub is a popular platform for collaboration on open source projects. It also provides a rich API to query various aspects of public activity. This combination of a popular social coding website with a rich API presents an opportunity for researchers to gather empirical data about software development practices.

I completed a project which analyzed GitHub data taking into account the developers' location and their technology usage [11]. As part of this project I developed a web-based tool to interact with and visualize the data. In its current state of development, the tool summarizes the amount of code developers have in their public repositories decomposed by programming language, and summarizes data about programmers using specific programming languages. This allows website visitors to get an immediate picture of the programming language usage in their region. Future research could expand this work to technologies beyond programming languages such as frameworks and libraries.

This work will not be discussed further in this report as it has already been

published separately [11].

3.2 Online Job Postings

My project focused on the next source of data that was considered: online job postings. Skill requirements in web-based job postings were used as the predictor for software development technology usage in the project documented by this report.

In particular, job postings from VIATec's job board (http://www.viatec.ca/job-board/results) were archived and analyzed. This job board is primarily for technology jobs in the Victoria, BC area. It includes postings for software developers of all kinds and system administrators. However, it also has postings for less technical positions such as in management, administration, sales and marketing for companies involved in the high technology industry.

Data collection of job postings is done by querying an RSS feed which returns an XML document containing all currently open job postings. This data collection was done twice a day starting on September 9, 2012. Since there were old postings still unfilled, the earliest job postings archived in the system actually date back to June 21, 2012, but the postings prior to September 9, 2012 are obviously incomplete. There is a complete record of all postings from September 9, 2012 onwards to the present day. Since the developed system (see Chapter 4) aims to be as up-to-date as possible, data collection has recently been increased to every two hours between 7:00 am and 7:00pm.

These job postings provide both unstructured and semi-structured information about the skills required. Some information such as the publication date of the job posting are provided in a structured format. However, the skill requirements are in natural language, embedded in a document which is also attempting to market the employer to potential employees. This makes it more challenging to extract the relevant information.

Next, Chapter 4 discusses the data wrangling, analysis and visualization performed to extract meaning from this data.

Chapter 4

Data Product

This chapter describes the "data product" created by this project. We use the term data product to mean an interactive system which:

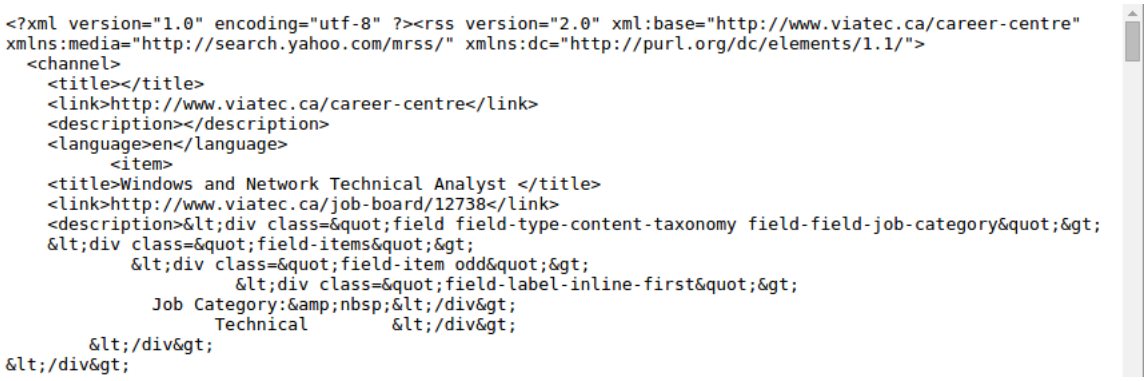
1. processes raw input data into a format suitable for analysis (data wrangling)
2. performs the analysis
3. visualizes the result
4. allows for the process to be repeatable

The remainder of this chapter will be structured by discussing each of these aspects in turn.

4.1 Data Wrangling

Data was collected from the VIATeC RSS feed (http://www.viatec.ca/job-board/feed). Figure 4.1 shows the first fews lines of an RSS feed as an example. The feed is too long to show in its entirety, and it is not easy for a human to read due to the job description being in escaped HTML (i.e. it contains strings like `&lt;/div&gt;` which are not part of the content of the job description, they are just for formatting rendered versions).

The RSS feed contains a list of job postings in XML format. To get a better idea of what information is available, Table 4.1 describes each of the elements present in a job posting.



```
<?xml version="1.0" encoding="utf-8" ?><rss version="2.0" xml:base="http://www.viatec.ca/career-centre"
xmlns:media="http://search.yahoo.com/mrss/" xmlns:dc="http://purl.org/dc/elements/1.1/">
  <channel>
    <title></title>
    <link>http://www.viatec.ca/career-centre</link>
    <description></description>
    <language>en</language>
          <item>
    <title>Windows and Network Technical Analyst </title>
    <link>http://www.viatec.ca/job-board/12738</link>
    <description>&lt;div class=&quot;field field-type-content-taxonomy field-field-job-category&quot;&gt;
    &lt;div class=&quot;field-items&quot;&gt;
            &lt;div class=&quot;field-item odd&quot;&gt;
                    &lt;div class=&quot;field-label-inline-first&quot;&gt;
              Job Category:&amp;nbsp;&lt;/div&gt;
                    Technical       &lt;/div&gt;
        &lt;/div&gt;
&lt;/div&gt;
```

Figure 4.1: First few lines of RSS feed.

Element	**Description**
title	The title of the job.
link	URL to the original job posting on the VIATeC site.
description	A free-form description of the job. It will contain the required skills in natural language. It is the actual HTML from the online job posting, so contains markup.
job category	A designation such as Technical, Sales and Marketing, Management and Operations, Administrative, or Other. A job can only be in one category.
job specialization	A more specific categorization of the job, such as software development, web development, etc. A job can only have one specialization.
tags	A job can have multiple tags to annotate it even further than with just the category and specialization.
group	A reference to the company which posted the job. VIATeC gives them a unique id number.
pubDate	A timestamp from when the job posting was created.
creator	The email address of the person who created the job posting.
isPermaLink	Indicates that the link to the job posting will not be permanent (ex. after the job is filled). I have only ever seen the value False.

Table 4.1: The elements that make up a job posting in XML format and their descriptions.

Parsing fields other than the description is fairly straight-forward. Sometimes the information is awkwardly embedded in a URL that is an attribute of the XML element, but the information is not difficult to extract. Parsing the free-form description field to extract skills is more challenging. The software system as currently implemented only looks for programming languages. It does this by tokenizing the description, and then searching the tokens for the names of programming languages.

For example, "Experience with Python" is tokenized to a list of tokens: ["Experience", "with", "Python"]. Next, when scanning the tokens, Python is identified as one of the programming languages for which the system has been configured to recognize. The Python Natural Language Toolkit (http://www.nltk.org/) is used for tokenizing. A regular expression based tokenizer is used. Generally it splits on words, but the regular expression takes into account special cases such as "Objective C" which should be considered a single token.

As currently implemented, programming languages are found in the description of the job posting and then associated with that job posting. The Analysis section presented next gives more details.

4.2 Analysis

Once the job postings have been parsed, they are stored in an SQL database. Figure 4.2 shows the entity-relationship diagram for the database.

This database allows a wide range of queries to be made, facilitating any analysis we wish to do.

Current analyses performed include:

1. programming language trends showing how many jobs required the programming language over time

2. companies' job posting trends over time

3. a summary of programming languages and the total number of job postings they have appeared in

4. a summary of companies and the total number of jobs they have posted

5. lists of job postings by programming language or company

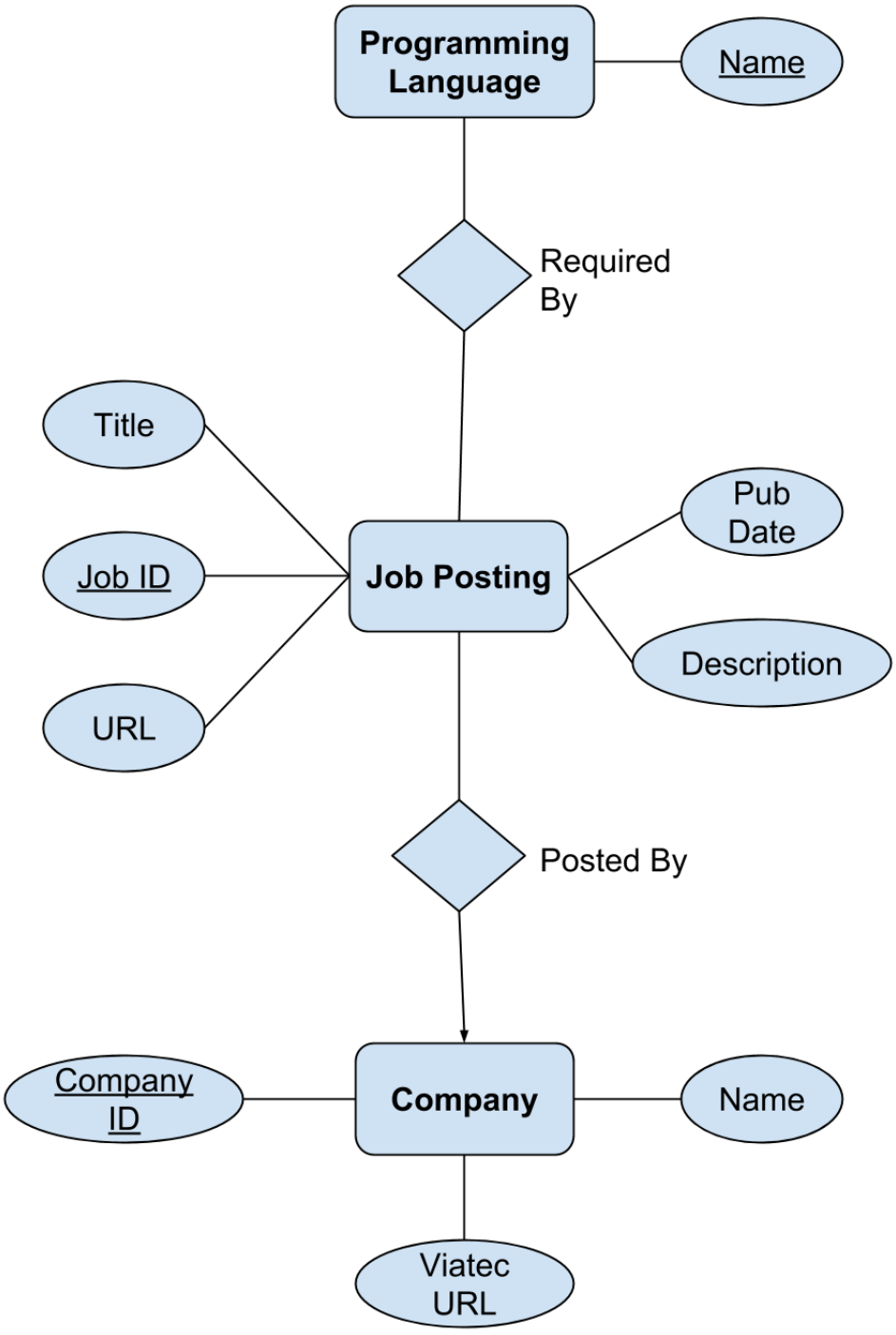


Figure 4.2: The entity-relationship diagram for data stored by the job parsing system.

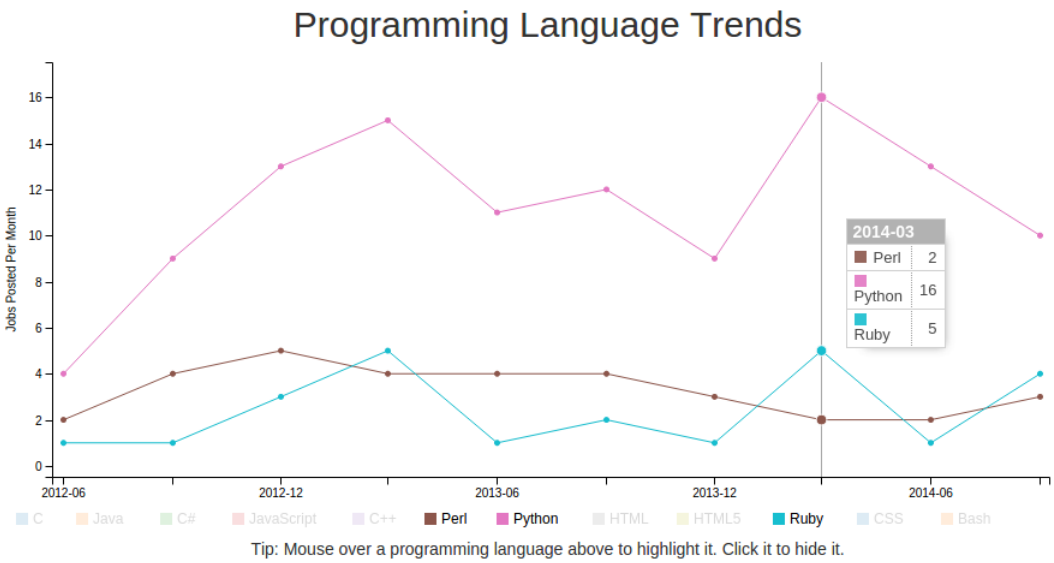


Figure 4.3: A comparison of job posting trends for Python, Ruby and Perl.

In addition to these analyses, the system also simply acts as an archiving service where the full text of old job postings can be viewed even if they are no longer available elsewhere.

The next section discusses the visualizations related to these analyses. Chapter 6 proposes future planned analyses.

4.3 Visualization

The visualization types currently used in the project are line charts, data tables, bar charts and tag clouds. Work on determining which visualizations work best is still being done. The two primary visualizations currently implemented are line charts and data tables. They will be discussed further below.

4.3.1 Line Charts

We use line charts to show time series data such as trends in programming language popularity in Figure 4.3.

Note that individual lines can be highlighted by mousing over the label in the legend, and hidden by clicking on the label in the legend. The values of the different lines at any point can be found by mousing over them.

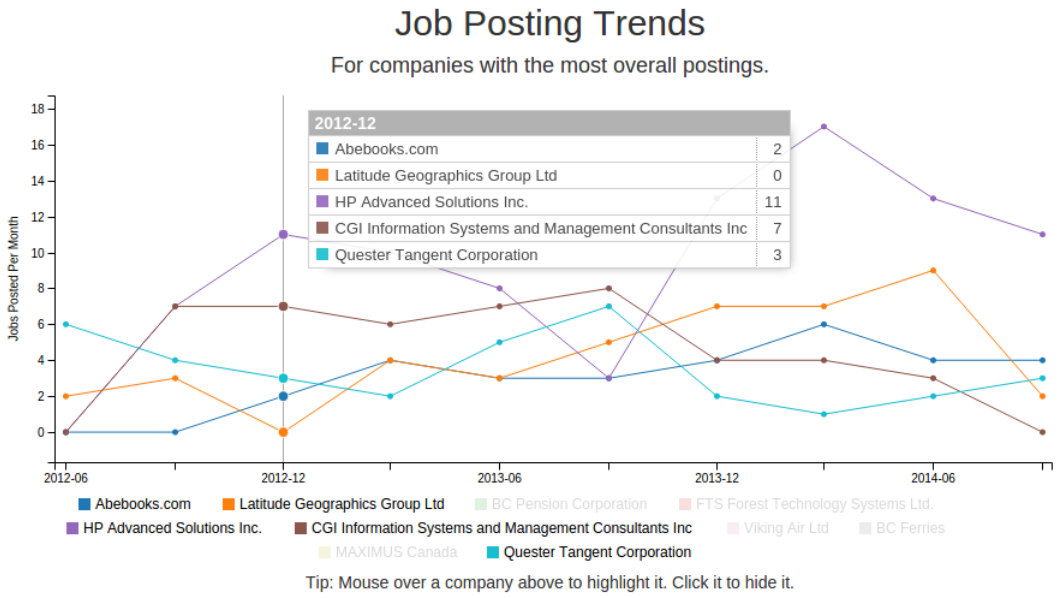


Figure 4.4: Job posting trends for various companies.

Figure 4.4 shows another example where the number of jobs posted by different companies is tracked over time.

4.3.2 Data Tables

Data tables in this project are simply tabular data which has been enhanced by making the columns sortable, paginating results, and providing search capability to filter down results. Figure 4.5 shows an example where job postings are listed, and the user has entered "java" as a filter.

Figure 4.6 shows another example of a data table displaying the number of job postings that each programming language has been mentioned in.

4.4 Repeatability

This system is constantly evolving as we try to extract more information and perform new analyses. If we only stored the parsed results in the database, the whole database would have to be regenerated every time we extracted some new piece of information into a new field. This would be tedious and time consuming. Therefore the full text of the job description is also stored in the database. Then when a new field is added to the schema based on parsing new information, a data migration can be performed

Show 10 entries Search: java

Job	Company	Date Posted
Java Developer	Priologic Software Inc.	September 25, 2014
Intermediate Java Developer	Pacific GeoTech Systems Ltd.	September 03, 2014
Senior Java Developer	GameHouse Canada	August 15, 2014
Intermediate Java Developer	Andrew Sheret Limited	July 28, 2014
Senior Java Developer	GameHouse Canada	July 02, 2014
Java Developer	CGI Information Systems and Management Consultants Inc	February 17, 2014
Front-end Web Developer (JavaScript)	Foundry Spatial Ltd.	January 20, 2014
Intermediate Java Developer	Andrew Sheret Limited	January 09, 2014
Senior Java Developer	Kinetic Systems Inc.	December 29, 2013
Senior Java Developer/ Architect	CGI Information Systems and Management Consultants Inc	October 07, 2013

Showing 1 to 10 of 25 entries (filtered from 1,554 total entries)

Previous 1 2 3 Next

Figure 4.5: A data table which has been filtered to show jobs mentioning Java in the job title.

Show 10 entries Search:

Programming Language	Job Count
JavaScript	242
HTML	202
Java	178
CSS	173
C#	154
Python	112
C++	97
C	92
HTML5	80
Perl	33

Showing 1 to 10 of 12 entries Previous 1 2 Next

Figure 4.6: A data table showing how many job postings different programming languages were mentioned in.

to update existing entries in the database.

As will be discussed in Chapter 5, the system is implemented using the Django web framework. As of Django version 1.7, there is a built-in database migration tool which can be used to create schema migrations (ex: adding new fields) and data migrations (ex: populating new fields). This tool makes it much simpler to update the database models.

Chapter 5

Implementation

This chapter discusses some details of the implementation of the project. Since this was termed an "industrial project" I thought it would be appropriate to spend the extra time to apply best practices which are widely used in industry, but generally overlooked in academic projects. I took full advantage of this opportunity to further my experience with test-driven development, continuous integration, configuration management, and so on.

The project is primarily implemented in Python, using the Django web framework (`https://www.djangoproject.com/`). In the past I have favoured Flask (`http://flask.pocoo.org/`), but Django is arguably the most popular Python web framework and much more mature than Flask, so I wanted to experiment with it. Additionally, in the spirit of the project, I used the Unix utility `grep` to search the job postings for Django and Flask. Flask was never mentioned, while Django had many mentions. Therefore, it also seemed like a valuable technology to learn.

The front-end uses Bootstrap (`http://getbootstrap.com/`) for structure and styling. JavaScript is used for features such as creating charts. In particular, C3.js (`http://c3js.org/`) is used for creating trend charts. It is a library built on top of D3.js (`http://d3js.org/`), providing a high level interface for creating a variety of common charts using SVG. The DataTables plugin for jQuery (`http://www.datatables.net/`) is used to make tables, such as lists of jobs, sortable, paginated and filterable.

5.1 Version Control

Git was used for version control, and GitHub was used for hosting the Git repository. The repository is currently being kept private, but I plan to make it public soon since it contains no sensitive information. GitHub was also useful for tracking issues and feature requests.

5.2 Testing

Test-driven development was used in this project due to its benefits such as:

1. improved design by focusing on the interface before the implementation
2. more modular code because units need to be tested individually
3. ability to break complex tasks into small incremental steps
4. ability to safely refactor
5. fewer bugs and regressions

The book *Test-Driven Development with Python* [10] was a great source of information about testing a web application developed using Django. Django provides utilities for unit testing the server-side Python code, while Selenium (`http://www.seleniumhq.org/`) was used for functional testing from the site visitor's point of view. Jasmine (`http://jasmine.github.io/`) was used for unit testing the JavaScript code.

5.3 Continuous Integration

Continuous integration is the practice of merging changes to a software project back into the main branch as soon as possible. This avoids integration problems that can occur if development occurs in a separate branch for a prolonged period. An important part of continuous integration is validating code changes. This is done by running the test suite whenever new commits are pushed to the GitHub repository.

In the past I have used Travis-CI (`https://travis-ci.org/`), which is a hosted continuous integration service which is free for public repositories on GitHub. Since this project is currently private, I instead installed and configured Jenkins (`http:`

//jenkins-ci.org/) on my own server. In the event that a regression is introduced and caught by the test suite, an email alert is sent.

The server does not have a GUI, so in order to run the Jasmine JavaScript unit tests, a headless browser called PhantomJS (http://phantomjs.org/) is used. For the functional tests, it was desirable to still use real web browsers to capture any strange quirks, so a virtual display called Xvfb (X Virtual Framebuffer) was used.

5.4 Configuration Management and Deployment

Setting up servers in an automated way greatly increases reliability and reproduceability of production environments. For this task, a tool called Ansible (http://www.ansible.com/home) was used. It installs all required software, configures the HTTP server NGINX (http://nginx.org/), configures the Python WSGI HTTP server Gunicorn (http://gunicorn.org/), and sets up the cron jobs for downloading data. Since there are multiple versions of the site (testing, staging, live), it would be tedious and error prone to repeat this configuration manually for each one.

Virtualenv (https://virtualenv.pypa.io/en/latest/) is used to create an isolated Python environment where libraries used by the site are installed. This helps greatly in managing dependencies and versions.

The actual deployment of the code is automated using Fabric (http://www.fabfile.org/en/latest/). It performs operations such as generating the secret key for the application and injecting Google Analytics for the appropriate host. In the future deployment may also be done with Ansible to streamline things further.

5.5 User Feedback

The system has been shown to some VIATeC employees and mentioned on the CHISEL blog and Twitter account. The VIATeC employees were positive about the value of the project. One individual was involved in teaching women how to program, and she said a frequent question from students was which programming languages should they prioritize learning in order to get a job. This system directly addresses that issue. As more of the currently planned features are implemented, further feedback will be collected.

5.6 Google Analytics

Google Analytics was installed on the website to gather quantitative data for evaluating the project. It was added on November 6, 2014. As of the morning of December 2, 2014 there were 96 recorded sessions across 33 users resulting in 470 page views. There were 4.90 pages viewed in the average session which lasted 7 minutes and 19 seconds. The bounce rate (sessions where the user left the site without exploring) was 33.33%.

These statistics suggest that a high percentage of users do not know what they can do with the site when they see it and simply leave. However, those who do stay tend to explore a reasonable amount, staying over 7 minutes on average. Future work should try to address this by making the user interface more engaging for visitors, and providing better direction to the content relevant for their use case.

Chapter 6

Future Work

This project is rich in possibilities for future development. One of the most important for widespread adoption of the tool would be improvements to the user interface (UI) which draw the visitor in to explore site content. As the application is now, the visitor may not realize what can be done with the system. Dr. Margaret-Anne Storey and PhD student Alexey Zagalsky have lent their expertise in human-computer interaction (HCI) to provide feedback in this area. Essentially they recommend the user interface should present the main use cases for the visitor to choose between, and then guide them to the parts of the site which are relevant for their use case. Work in this area will continue throughout the remainder of the academic term. Google Analytics was installed on the site so after reworking the UI various metrics, such as bounce rate, can be compared before and after changes to measure whether the changes were effective.

Another important area for further development is expanding the technologies which can be parsed from the job postings. Technologies other than programming languages should be recognized, such as libraries, frameworks, databases, operating systems, version control systems, and so on.

There are also many more analyses and visualizations that would be interesting to provide. For example, analyzing the technologies used by a specific company so that site visitors can get an idea of the technology stack used at companies in which they are interested.

6.1 Commercialization

If this project were to become more than a master's degree project, one potential way for it to make money would be to offer a job posting service itself. This could be done in such a way that skills are entered explicitly, rather than having to be parsed. In order to be viable, the site would have to stand out as the best available service to match up your skill set with job postings. Therefore, perhaps users should have a profile where they can list their skills, or import them from another service like LinkedIn. Then job posting reports and notifications can be generated, tailored to users' interests.

The domain name GoodJobs4U.com was registered for this project and a cloud server on Linode (`https://www.linode.com/`) was set up for development and project deployment. The DNS entry for GoodJobs4U.com resolves to this server. In the future it would be easy to deploy more sites such as `http://vancouver.goodjobs4u.com`, `http://seattle.goodjobs4u.com`, etc. if the project was found to be commercially viable.

Chapter 7

Conclusion

In this project I built a system for automatically archiving and analyzing job postings from the VIATeC job board. These job postings serve as an indicator of skills that are in demand in the local job market. A web application has been developed to display the results to the public. It provides visualizations such as charts showing trends in programming language popularity. This information can help students and educators make informed decisions about which technologies to learn. Feedback from employees at VIATeC about the project have been very positive. They indicated they would like to support future development by providing an API to query for data instead of parsing the RSS feed. They may also be able to provide further historical data for trend analysis. The two years of job posting data currently available is too short to see significant changes in the programming language skill sets currently being monitored. However, analyzing popular programming libraries, which will be added soon, is more likely to show trends over shorter time periods because new libraries appear regularly.

Bibliography

[1] R. Agrawal and R. Srikant. Fast algorithms for mining association rules. pages 487–499, Santiago, Chile, 1994. In Proceedings of the 20th International Conference on Very Large Databases (VLDB-94).

[2] M. E. Califf and R. J. Mooney. Relational learning of pattern-match rules for information extraction. pages 328–334, Orlando, FL, 1999. In Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99).

[3] Canadian Coalition For Tomorrow's ICT Skills (CCICT). The growing IT skills gap: How organizations are adapting. `http://ccict.ca/news/the-growing-it-skills-gap`, 2011. Accessed: 2013-10-30.

[4] W. W. Cohen. Fast effective rule induction. pages 115–123, San Francisco, CA, 1995. In Proceedings of the Twelfth International Conference on Machine Learning (ICML-95).

[5] D. Freitag and N. Kushmerick. Boosted wrapper induction. pages 577–583, Austin, TX, 2000. In Proceedings of the Seventeenth National Conference on Artificial Intelligence (AAAI-2000).

[6] Jiawei Han and Micheline Kamber. *Data mining: concepts and techniques*. Elsevier, Burlington, MA, 2011.

[7] N. Kanya and S. Geetha. Information extraction - a text mining approach. pages 1111–1118. In Proceedings of the Sixteenth National Conference on Artificial Intelligence (AAAI-99), 2007.

[8] Qingxia Kong, Yang Cai, and Quanyin Zhu. The case study for the basic information service of job post resource based on web mining. pages 498–501. IEEE, 2012.

[9] T. N. Layton. The integration of skill sets used in industry into the bioengineering curriculum at the University of Illinois at Chicago. volume 3, pages 2600–2601 vol.3, 2002.

[10] Harry J.W. Percival. *Test-Driven Development with Python*. O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA, 2014.

[11] David Rusk and Yvonne Coady. Location-based analysis of developers and technologies on github. pages 681–685. The 2014 International Symposium on Mining and Web, 2014.

[12] C. Terblanche. Meeting employers' and students' expectations through the use of employer demand ontology in curriculum development. pages 80–85. IEEE, 2011.

[13] Mantz Yorke. *Employability in higher education: what it is, what it is not*. Learning and employability series 1. Higher Education Academy, 2006.

[14] Shilin Zhang and Mei Gu. Job opportunity mining by text categorization. pages 1–4. IEEE, 2010.