MSc Computer Science
Department of Computer Science and Information Systems
Birkbeck, University of London

Project Proposal

Web based job flow organization for self-employed couriers

Raitis Cerkasovs

This proposal is substantially the result of my own work, expressed in my own words, except where explicitly indicated in the text. I give my permission for it to be submitted to the JISC Plagiarism Detection Service.

April 2014

Abstract

In this proposal, I outline the analysis, design and development of the prototype system that will be used to build an online job flow for self-employed couriers. A self-employed courier is a person who is not connected to any organization. The proposed system will consist of web and mobile application. The purpose for it (system) is to digitalize and resolve issues in self-employed courier job cycle. The courier job cycle includes an indication of his (courier) availability, reference of parcel collection, delivery status change, online tracking and visibility. The issues in self-employed courier job cycle are trust of customers and verifiable previous job history, the organization of his accounts and equipment. Technology must satisfy the considered scale of the project. Evaluation will be made by third party persons simulating various activities of courier job flow.

Table of contents

1. Introduction

In this chapter, I will provide research of existing situation and scale of couriers industry. I will define the role of self-employed couriers and challenges within it. Then I will formulate the aims and objectives of the project.

1.1. Aims and objectives

I propose to write online based prototype software which would provide individual accounts for couriers and customers

My main objectives are:

- To create a web application for providing couriers and customers with a registration option. Therefore, couriers will be able to keep accounts on previous jobs and customers - to book jobs in required area and trace the progress of delivery. In addition customers will also have the option to leave feedback for the services. That will resolve the issue of the courier's trust. Furthermore, customers would be able to communicate with the courier directly without using a third party (courier companies). Hence courier service costs will be reduced.

- To create a mobile application for couriers to use with smartphones for tracing, receiving jobs and providing an option for proof of delivery. In this way, it would resolve the issue to avoid the necessity for extra equipment.

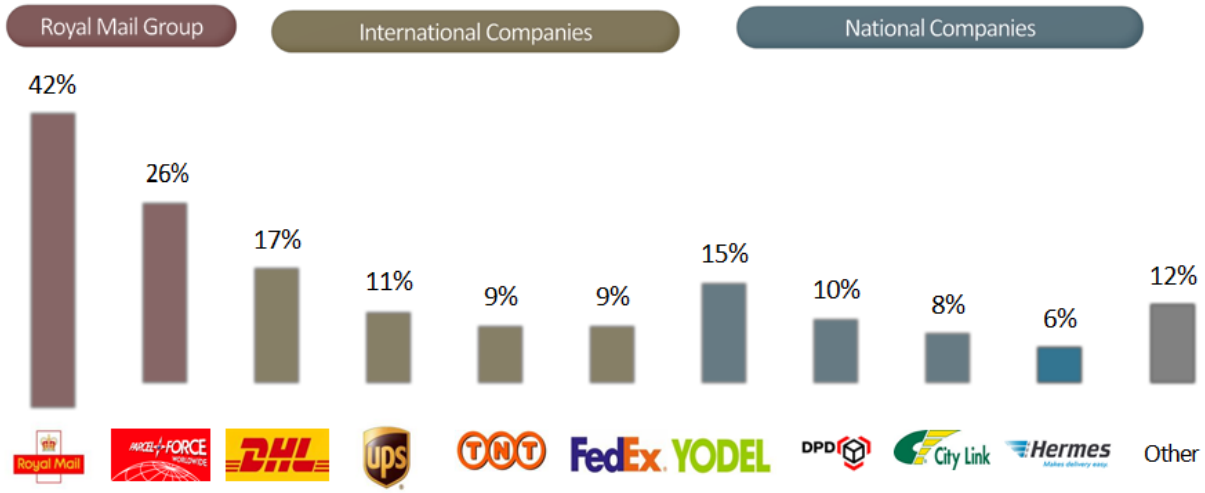
- To write a prototype software for much higher scalability as initially needed. If the software is going to gain the popularity among self-employed couriers and it is going to be connected with main outside courier networks such as Royal Mail, DHL and UPS, etc., it has to be designed in such a way to be adequate to cope with larger scale of data (courier jobs).
  Therefore, it could become like a single instance, adapter for all self-employed courier job offers online and could resolve other issues of couriers being connected with only one of courier's networks.

1.2 The Courier Industry Background and Scope

Within UK postal and courier activities in the industry are operated by licensed and non-licensed companies as well as self-employed couriers, to pick up, sort, transport and deliver letters and parcels. Almost a billion parcels are estimated to have been delivered in the UK, in 2013 by an expanding logistics workforce of around 1.8 million people, supporting a domestic parcel

industry worth around £4.4bn [4]. Due to online shopping continues to grow; industry revenue is expected to expand over the five years at a compound annual rate of 8.4%. In the current year, revenue is projected to increase by 4.6% [4].

The leading operators are Royal Mail [23], DHL [24], TNT [25], YODEL [26] and others [2]. Logo of each company holds reference to its website.



1.3. Role of Self-employed Couriers in Industry

Self-employed drivers now aggregate anything between 5-8% [4] (approximately 100 000 – 130 000 persons) of UK logistics workers, with one of the rapidest expansions in the group known in the industry. Most of the big parcel industry players are outsourcing local delivery jobs and are using self-employed couriers to help them cope. Another area is hand to hand deliveries. Parcels are coming from small companies or individuals rather than major online shops and retails, for A to B deliveries.

1.4. Self-employed Couriers Job Flow

In general terms couriers job flow means:

- Courier receives collection and delivery address in a reasonable area. Courier location must be known
- Courier is able to show collection of parcel
- Courier is traceable during delivery time
- Courier is able to indicate the delivery and collect, store, and provide, in case of request, the proof of delivery

1.5. Issues and problems of Self-employed Couriers

The courier industry is still thriving with most of the market being dominated by the big companies [3]. If a self-employed person wants to have access to their job flow, he must be registered to company's inner network, therefore, losing his independence and spirit of self-employed nature. Within the network couriers must have a tool designed for communication with an operator, to receive jobs and produce proof of delivery.

In early days, couriers had a list of addresses written on the paper with ail for signature as proof of delivery. Until now, all parcels for delivery purposes were collected from one location. Nowadays to improve efficiency courier must have PDA device [15] to organize his job flow and make collections from any place in the area he is covering. It [PDA device] is a personal digital assistant, also known as a palmtop computer or personal data assistant and is a mobile device. It functions as a personal information manager, connected to the network that a courier currently is serving. Such device could be expensive and qualified as essential extra equipment. It can be configured for work flow with only one network. Therefore if a courier, for example, wants to work for TNT and DHL simultaneously, then he would need two devices connected to each.

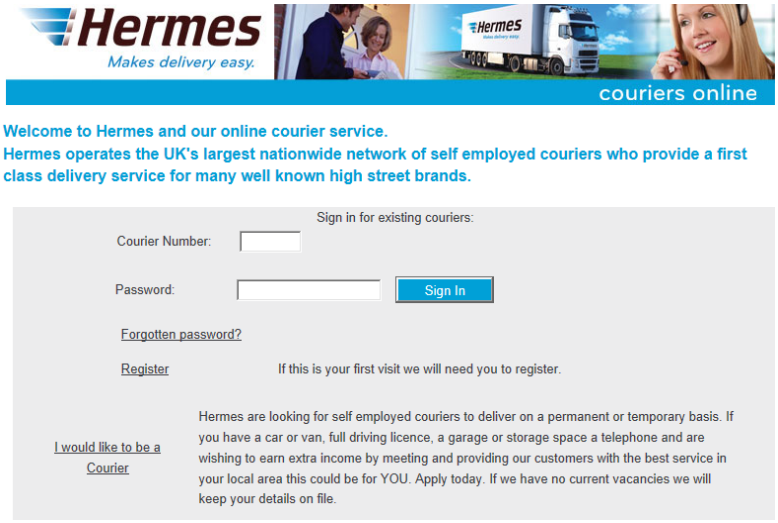
If the courier has decided not to cooperate with any large courier company and to do private jobs instead, he would still need a way to organize accounts, receive jobs, store proof of delivery and represent his availability. Trust of courier plays an extensive role in this industry.

Accordingly the issues are as it follows: link to only one network, expensive equipment for work flow in solid collaboration with larger companies, trust of customers, tracing possibility and accounts for private job life cycle.

1.6. Existing Solutions

In the market today exists variety of free or commercial software suitable for large, medium or small size courier job flow operators to organize their fleet of couriers, such as **E-courier** or **Courier Systems** [holds reference to its website]. Software examples would be **FieldAware**, **FleetWIzard**, **Key Courier Systems,** etc. Unfortunately, none of them complies with the needs of self-employed courier. Furthermore, there is an online solution for job sharing among couriers, such as **Courier Exchange**. Members can post available loads, journey information and real time vehicle availability status on exchange. However it does not organize job flow, only gives access to jobs.

Almost every major operator has online way how to work with self-employed couriers and outsource jobs. As example **Hermes** provides online registration where courier must complete the form and then join inner distribution network of Hermes.



Unfortunately, it is not possible to analyze how this particular system works, because you have to be approved as a courier. However it is obvious that the connection here is established only with one, Hermes network.

In conclusion, there isn't software suitable for courier doing person to person private jobs, being appropriate only for one person (not fleet) tracing and job flow organization. Couriers in such cases operate using phones to communicate with customer and if necessary use paper and pen for proof of delivery. There isn't such a way for customers to track the courier, and for courier to have proof of being trustable.

2. Project Design

In this chapter, I intend to summarize and establish design to complete the proposed project. I will also introduce to the evaluation of the design and its implementation stages as well as a third party assessment of conformance.

2.1. Blueprint

Program will be created as web rather than standalone application. Programming will be done on a single computer with all necessary tools installed, such as Eclipse, Database, Web server etc. Final result will be hosted on a public server.
Web application will provide standard *home* page interface with general information about the services and its direction. Main page will have an interactive map (potentially Google) with current locations of all available couriers, specified area that every courier is covering, vehicle type (for example, an icon on the map that could represent a bike or a car) and their feedbacks (could be a mouse-over option). It will be grouped in two types of registrations - one for couriers and another for customers. The webpage will have an integrated payment option for courier jobs. For this scale of the project, I am proposing to use only PayPall. Payment choices could be in full, in advance or into installments. Job bookings could be made by clicking on appropriate courier profile where all his information, feedbacks, mobile phone number, etc. would be available for clients.
Moreover, courier's account will contain all his previous job routes, customer's information, payment statuses, etc. Couriers won't be permitted to edit feedbacks from clients in order to avoid fabricated information. Going back to the registration of a courier, after its completion he will be able to access his account to edit personal information, download and install a mobile app. Also, in the interest of couriers to avoid fake jobs, some history of customer's past actions (job bookings) will be visible.
Discussing a customer account in more detail, it will hold complete information about all previously booked jobs and trace the current ones. There will be an interactive map with current location of a courier from the moment of collection until completion of the job with proof of delivery.
Mobile app will store couriers previous (one day) jobs, receive new jobs, send courier's current location to the server, change his availability status and make proof of delivery that is going to be a captured receiver's signature or / and a photo with a delivered object in the location. For example, parcel at the doorway of a delivery address. Therefore, it would also be considered as controllable condition of the delivered parcel.

2.2. Data Storage

Database initially will be categorized into minimum of following entities:

1. Courier
2. Client
3. Sender
4. Receiver
5. Job
6. Proof of delivery

Database access will be designed in such a way where it would be straightforward to upgrade in the case of its growth. I propose to use Hibernate ORM (Object/Relational Mapping) [17] as it enables developers to write applications more accessible whose data outlives the application process.

2.3. Security

Security access will be divided into four groups: Guest, Client, Courier and Administrator.

- Guest access would be for unregistered users. It would allow access to all currently available couriers along with their feedbacks on an interactive map. There will also be available information of courier's vehicle type and distance he is covering.
- Client credentials would allow to make bookings, trace current jobs, explore couriers job history, leave feedbacks, access proof of delivery and make payments.
- Courier credentials would give access to all previous job accounts, receive new jobs on smartphone and change availability status.
- Admin would have super user credentials to access any level of data. It will be in his duty to delete inappropriate feedbacks, fake jobs etc.

2.4. Capture Signature

As being a web project, Graphical User Interface (GUI) will be achieved by HTML, CSS and Javascript. Some additional elements may be used from Javascript JQuery library. Styling may also include some open source premade templates.

2.5. Unit testing

Testing will be done for both syntax and logic errors. Test-cases will be written to check individual sections of code (usually a class or a method). The unit tests will be done prior to or alongside the code to check it throughout the development.

2.6. Evaluation

It is critical to evaluate accessibility throughout the design and implementation stages of the system to manage its conformance. However, it is important to be aware that evaluations carried out while these development stages can quickly become obsolete by implementing even minor changes. For a self-assessment of conformance during the development process, particular attention will be made to use cases, design analysis, technical specifications and testing resources. Developed parts will be tested in real life environments. For example, code unit for android smartphone sending location updates. Is it quick enough for certain update frequency? Such evaluations must be carried out during the development process.

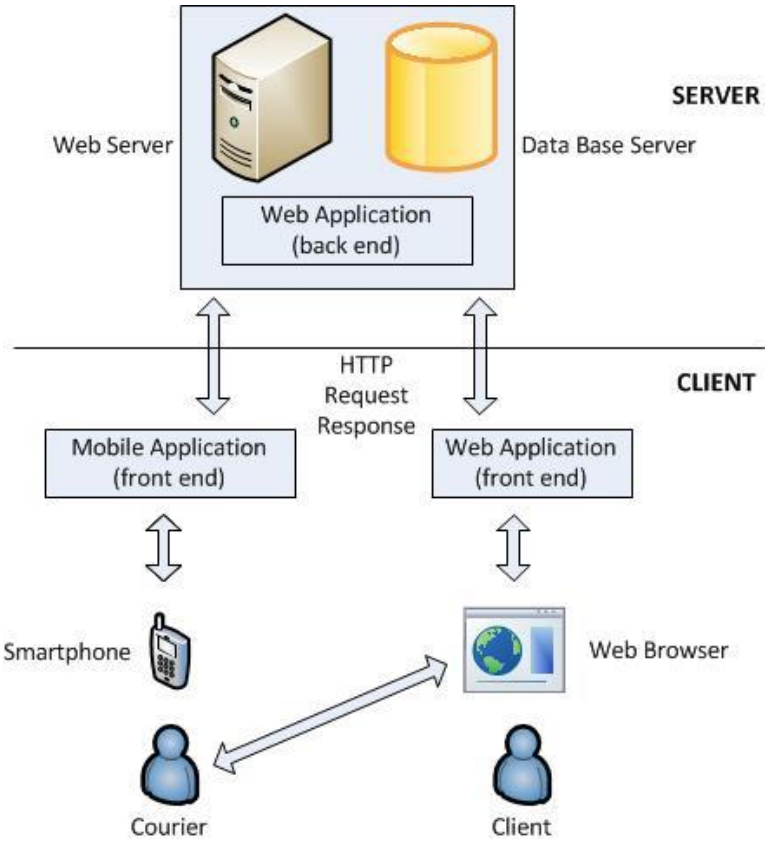
At the final stage of the project, the third party assessment of conformance will take place. Five persons will be simulating courier activities from the registration part until the commitment of three independent courier job flow cycles each. That will include realistic movement around the city, their devices (smartphones) tracing, collection and delivery of testing parcel. Results will be analyzed in the project report.

3. Technology and Analysis

In this chapter, I will outline technologies used by different web development platforms and will analyze the best possible tools and practices to develop an online system appropriate to use in the project.

3.1. Technology Choice

To gain popularity from potential user's software, it must be accessible, platform independent and avoid any extra requirements. Therefore I choose to write web based instead of a standalone application. It must utilize server-side scripting (backend), client-side scripting (frontend), mobile application and databases to store a range of data.



Web projects as such, are accessed by users using a 2-tier architecture referred as the client-server model. The server has one or more programs running on a machine holding stored information such as a database. It [server] usually runs on a machine that is permanently switched on and linked to the internet network and waits to receive requests for information. The client's access will be implemented via web browsers and mobile devices, simultaneously sending requests via the network to access the remote-run processes and data.

3.2. Choose of Web Development Platform

There are a lot of different approaches and appropriate technologies available to develop web based systems.The most popular and widespread platform formats today are:

1. Java EE [30]
2. Linux, Apache, MySql, PHP, or LAMP [31]
3. Microsoft, Dot Net [32]
4. Ruby on Rails [33]
5. Python Django [34]
   Others

Each of them is based on different programming language and philosophy. Because Ruby on Rails is considered with slower performance, and additionally my lack of knowledge of it, I do not consider it as potential development platform. Django is not appropriate for the scale of this project, therefore, I will be comparing in details only JEE, PHP and Dot Net [6].

Area	LAMP	ASP.NET	JEE
Licensing cost	No licensing cost	Expensive licensing cost	No licensing cost
Support options and cost	Free support via community Paid support options available	Free support via community Paid support options available	Free support via community Paid support options available
Platform(s)	Multiple	Windows only	Multiple
Staffing	Somewhat difficult to find qualified people	Reasonably easy to find qualified people	Reasonably easy to find qualified people
External Hosting	Widely available and inexpensive	Widely available, but more expensive	Not widely available
Security	Very good	Historically very bad, but improved recently	Good
Performance	Very good	Often requires more expensive hardware to perform well	Often requires substantial configuration
Scalability	Scales very well	Can be difficult to scale	Scales well when configured properly
Administration	Difficult: Often requires reading documentation and editing text files	Easy: Often can be done through point and click interface	Moderate: Sometimes can be done visually
Configuration ease of use	Can be difficult to configure properly	Easy to configure	Moderately difficult to configure

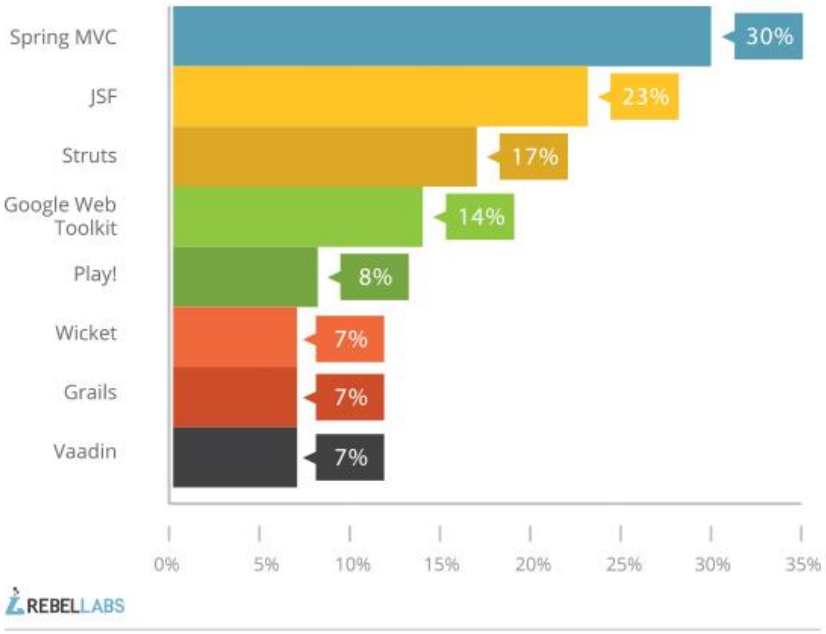
C# (dot net) has a high level of available documentation and for the scale would be appropriate, but I am not choosing it for this project purely because of involved costs. This is an academic project and must be completely for free, whereas dot NET is surrounded by costs of licenses, Microsoft server, buying plugins etc.

Acording to polls of most programmers JEE is better for enterprise applications. One of the reasons being, that Java is a more powerful language, since the developer/programmer can call on many other existing Java API's to perform complex tasks. To achieve the same in PHP, one has to look for existing plug-ins and download them. Consequently, although PHP might be easier to learn, JEE is easier to use. Besides, good idiomatic code examples in PHP are hard to find and for this reason its community known for poor code quality, lack of tests.

Scalability is always important when deciding on a language for a particular application. In PHP potential real time messaging can be an issue. For the initial stage of the project where few hundred couriers are doing about fifty deliveries a day, it would lead to few thousand transactions a day. However the target also includes the project being able to connect to other networks, such as DHL or UPS. Then, according to statistics, amount of couriers may increase to 130 000. Java EE is mostly used for large scale applications, while PHP is designed for smaller projects. In the project where the scale is 130 000 of potential users carrying 40 - 50 deliveries a day within only UK, I propose to use Java EE as a development platform.

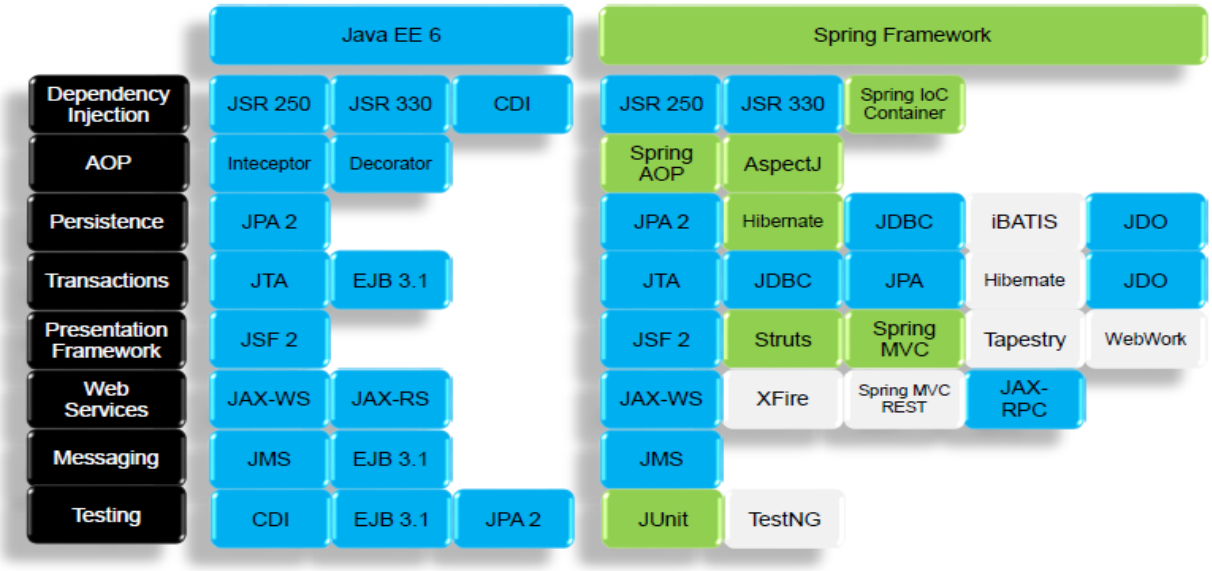
3.3. Development Framework

Present-day is not clear a web framework winner. Most marketable and widely used at the moment is Spring MVC. According to research by “Rebellabs” [10], where 1800 developers took part. It concluded that 30% of projects have been written by Spring MVC framework.



Spring MVC does not require a heavy application server, it can deploy an application to web container, such as Tomcat [36]. Spring MVC allows for accessible dependency injection

management and aspect oriented programming while providing ease for plugging in to other technologies, such as Hibernate [37], Java Message Service, etc. [35]. Spring MVC has excellent integration with Java EE [22]. I have provided with a picture from Derrick Kittler, Mauricio Maltron Leal and Vamsi Chemitiganti presentation where it is shown how variety of features from Java EE is wired into Spring.



Another aspect is the popularity of Spring MVC. It would be most appropriate to use this framework for the project that in the future is likely going to expand as there would be simple to find qualified people.

According to returns from **http://builtwith.com/** that check websites from database of 6000 web technologies and most likely is testing headers, it returned that Spring framework has been used for such big players as UPS, DPD and Hermes. It is not a 100% trustable source, but, probably, the only possible way how to find out which technologies are used by major players. Because of a quite ambitious potential scale of the project data, necessary flexibility of reconfiguration of components and its possible popularity among related projects, I propose to use MVC Spring framework for this project.

3.4. Data Storage

According to research from “DB-Engines” [12], three best ranked databases are Oracle, MySql and Microsoft SQL Servers. Ranking considers:

- Number of mentions of the system on websites. Measured through search engine results.
- General interest in the system. Measured through Google Trends.
- Frequency of technical discussions about the system. Measured through Stack Overflow and similar.
- Number of job offers, in which the system is mentioned. Measured through “Indeed” and similar.Number of profiles in professional networks, in which the system is mentioned. Measured through LinkedIn.

194 systems in ranking, October 2013

Rank	Last Month	DBMS	Database Model	Score	Changes
1.	1.	Oracle	Relational DBMS	1583.84	+54.23
2.	3.	MySQL	Relational DBMS	1331.34	+25.58
3.	2.	Microsoft SQL Server	Relational DBMS	1207.00	-106.78
4.	4.	PostgreSQL	Relational DBMS	177.01	-5.22
5.	5.	DB2	Relational DBMS	175.83	+3.58
6.	6.	MongoDB	Document store	149.48	-2.71
7.	7.	Microsoft Access	Relational DBMS	142.49	-4.21
8.	8.	SQLite	Relational DBMS	77.88	-4.90
9.	9.	Sybase	Relational DBMS	73.66	-1.68
10.	11.	Teradata	Relational DBMS	54.41	+3.32

The substantial presence today has been taken by NoSQL databases. They may be divided in following categories:
1. Key-values Stores. The main idea here is to use a hash table where it holds a unique key and a pointer to a particular item of data.
2. Column Family Stores. These were created to store and process very large amounts of data distributed over many machines.
3. Document Databases. These were inspired by Lotus Notes and are similar to key-value stores. The model is basically versioned documents that are selections of other key-value collections. The semi-structured documents are stored in formats like JSON.
4. Graph Databases. Instead of tables of rows and columns and the rigid structure of SQL a flexible graph model is used which, again, can scale across multiple machines.

NoSQL databases do not provide a high-level declarative query language like SQL to avoid overtime in processing. Rather, querying these databases is data-model specific [27].In my project data is related: jobs belong to couriers and clients, and deliveries to jobs etc.

Therefore, I am not considering NoSQL databases, because none of them would be appropriate for the project. NoSQL is used in cases where data is not structured or volume of data wouldn't fit in one machine.

For the chosen platform the most suitable from the list above are relational MySQL or Oracle databases. MySQL is relatively light-weight and can be extremely fast when applications leverage architecture. Lots of features stay free as the database servers grow such as replication and partitioning. Oracle offers lots of features/functionality for solving complex problems. It supports large OLTP environments as well as VLDBs [11].

Features/Functionality	MySQL	Oracle
Strengths	Price/Performance Great performance when applications leverage architecture.	Aircraft carrier database capable of running large OLTP and VLDBs.
Database Products	Enterprise (money) – supported, more stable. Community (free) – more leading edge.	Enterprise (money) Express (free) – up to 4GB
Application Perspective	Web applications often don't leverage database server functionality. Web apps more concerned with fast reads.	More you do in the database the more you will love Oracle with compiled PL/SQL, XML, APEX, Java, etc.
Administration	Can be trivial to get it setup and running. Large and advanced configurations can get complex.	Requires lots of in-depth knowledge and skill to manage large environments. Can get extremely complex but also very powerful.
Database Server (Instance)	Database Instance stores global memory in MySql background process. User sessions are managed through threads.	Database instance has numerous background processes dependent on configuration. System Global Area is shared memory for SMON, PMON, DBWR, LGWR, ARCH, RECO, etc. Sessions are managed through server processes.
Tables	Tables use storage engines. Each storage engine provides different characteristics and behavior.	A few tables with tons of features.
Partitioning	Free, basic features	Money, with lots of options

Oracle has many features from XML, user-defined types and lots of database management tools. It minimizes the need for a third party software where the advanced data management is needed. MySQL excels when high speed reads can be used for web, gaming and small/medium data warehouses and OLTP systems [11]. It can also be used by startup companies with simple usability at a low cost. MySql does not have a fraction of features of Oracle, but for the companies using MySQL, most of the functionality is built in the middle-tier and do not need lots of capabilities in the database.

In my project, in the job flow of courier, all data is relational (jobs are related to couriers and customers, deliveries to jobs etc.). I propose to use MySQL as a database. It is free, fast in the web environment and easy to use. At the beginning stage of the project there is no need for advanced data management, however, MySQL can also be highly scaled by using a methodology of replication or sharing [21]. Good example here is web social networking giant Facebook, which still uses MySql. Due to the probable growth of the project with possible connections to the third party networks, the need for advanced management and scale might be required. Design of the project will use ORM to allow in the case of necessity to swap easily MySQL to Oracle or any other suitable database.

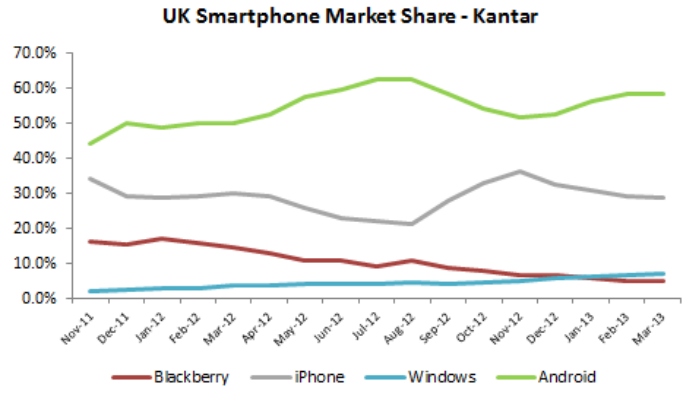
3.5. Mobile Application

In this section I will be comparing the popularity of different mobile platforms and will be choosing the most appropriate one to develop a mobile application. It will include analysis of three most popular ones: Android, IPhone, Windows and consideration of alternative use of X-platform tool. It will be followed by overview of the most applicable platform version and its compatibility.

3.5.1. Platform choice

In this project, a mobile application will provide couriers location tracking, job status and proof of delivery. Mobile device will not be extra equipment instead it will be widely used smartphone which courier already has. Here I am going to compare 4 major smartphone platforms which can be used by couriers.

According to UK market share research, the Android platform has extended its lead to nearly 60% as iPhone sales weakened after the iPhone 5 launch. Meanwhile, Windows Phone has seen steady growth reaching ~ 7% of smartphone sales [13].



As stated in the chart, Android platform, led by Samsung, is the most used one and it will maintain its volume advantage over Apple for 2014/15 [20]. Unfortunately we do not have specific usage statistic for couriers, therefore it is presumably equal as general statistics.

Developing for all of these platforms is a challenge. However, today cross platform development environments exist, for example, Appcelerator, PhoneGap, Rhomobile Corona that have the capability of software or hardware run identically on different platforms i.e. users can switch from one platform to another without converting their data to a new format. For example, Appcelerator is a free and open source application development platform, which lets you to create native mobile, tablet and desktop application experiences, using existing web skills as JavaScript, HTML, CSS, Python, Ruby, and PHP. It comes within built language-OS bridge and a runtime shell that compiles and packages your applications for cross -platform distribution.

Considering the scale of MSc project, I am not going to use a cross platform tool, because it is hard to fix glitches that may appear in relatively complicated units such as signature capturing. Instead I am going to choose Android as a prototype platform because it is the most used one [13], it presumably will remain in the leading position for the year 2014/15 [20] and likely is the most popular one among couriers. Android is the Java based development which allows the most flexibility in the development process.

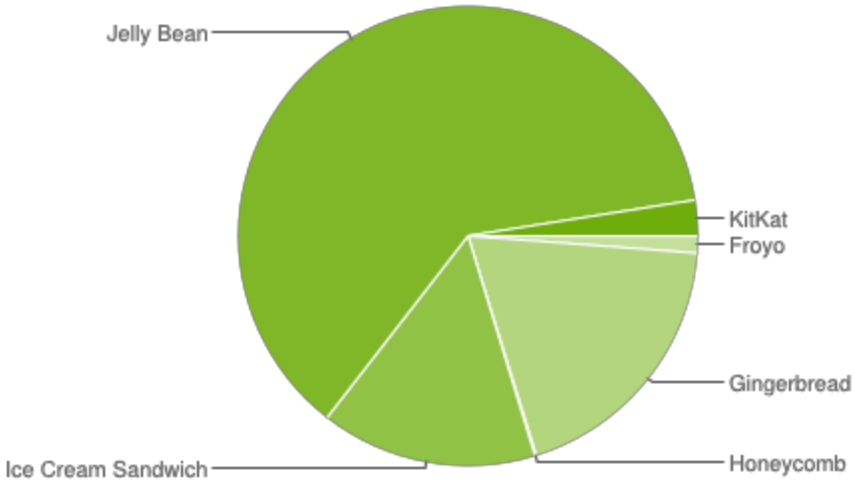
3.5.2. Android version compatibility

Proliferation of diverging variants of the **Android platform** is a result in the inability of some devices to properly run designed apps. With a large number of custom versions of the Android platform emerging, the concern is that interoperability will be weakened as a result of the potential for applications built specifically for one variant or device not being able to work with others. Variety of Android phones produced by different vendors, which often come with custom software, suffers from hardware-based and software based fragmentation. The term hardware-based fragmentation refers to the fact that at any moment, devices based on the same Android operating system run on different processors, graphics cards, and screen sizes. It causes some feature sin the same Android version to present different issues on the diverse devices. The term software-based fragmentation refers to three related aspects:

1 Parallel deployed versions of the Android operating system.
2 Vendors offer customized device-specific Android versions.
3 Carriers also offer software customizations.

Fragmentation suggests an opportunity for personalization and increased usability, since it enables users to choose the device and software that best meets their needs. It also implies that, due to insufficient cross-platform testing Android applications may not behave consistently across devices and versions of Android. [28].

To resolve such issues mobile app must be written in a simple manner and must be a compromise between technology and relative number of devices running on a given platform. Currently newest version is Jelly Bean 4.x [14]. As a chart shows it is divided in subversions 4.1 (35%) 4.2 (17%), 4.3 (10%) and 4.4 (2,5%).



To indicate which set of APIs are available, each platform version specifies an **API level**. For example, Android 1.0 is API level 1 and Android 4.4 is API level 19 [29]. If app uses APIs added in more recent platform version, but does not require them for its primary functionality, it is possible to check the API level at runtime or degrade the corresponding features when the API level is too low. I am choosing to start with Android version Jelly Bean v4.1, furthermore downgrade it if possible to rice compatibility.

4. Conclusion

In this document, I have explained the motivations behind the development of web based prototype software for self-employed courier job flow organization. As it was discussed in the introduction chapter, there are the considerable amount of people who are not working for any company, and due to lack of appropriate software in the market, they do not benefit from the potential they may have.

My developed software would be a prototype solution which would satisfy such a need for self-employed couriers. It would be capable of resolving all necessary courier job flow steps, discussed in chapter 2.3. The program will be written in a way to cope with the scale of data growth generated by self-employed courier activities. Therefore, it (scale) has to be sufficient with a significant increase if connection is made to outside courier companies, such as DHL, UPS, YODEL or other.

My previous experience covers only PHP approach to web development. During the work on the project, I am going to learn technologies such as Java Platform Enterprise Edition, Spring MVC framework and mobile programming for Android.

5. Proposed Timetable

Time required	Project Tasks
1 week	Learn Spring MVC from various tutorials
0.5 week	Setting up JEE development tools, Eclipse STS and Android SDK
0.5 week	Set up database, generate scaffolding for tables
0.5 week	Develop simple android app that returns GPS coordinates
2 weeks	Find and implement the best approach for android coordinate tracing in server
1 week	Develop web interface for web app
1 week	Develop interface for android app
1 week	Implement image file upload from android device to server
1 week	Implement signature capturing mechanism
1 week	Test all system in action
1,5 weeks	Project report writing
1 week	Project report rewriting

6. References

[1] Postal & Courier Activities in the UK: Market Research Report
http://www.ibisworld.co.uk/market-research/postal-courier-activities.html

[2] UK parcel delivery research by “Network Research 2013”
http://courier-direct.co.uk/news/wp-content/uploads/2013/03/UK-Parcel-Delivery-Research-May-2013.pdf

[3] Small Business Pro, starting a courier business
http://www.smallbusinesspro.co.uk/start-business/courier-business.html

[4] The irresistible rise of the UK's 'lifestyle couriers', By Harry Kretchmer 27 January 2013
http://www.bbc.co.uk/news/business-21219819

[5] Matt Aimonetti, author at O’Rayly “What Technology Should My Startup Use”
http://matt.aimonetti.net/posts/2013/08/27/what-technology-should-my-startup-use/

[6] Jason M. Hanley, B.Math, MBA, PMP President and Senior Consultant, Syllogistic Software Inc.
http://www.syllogisticsoftware.com/papers/Web_Development_Technology_Comparison.html

[7] Wikipedia, LAMP (software bundle)
http://en.wikipedia.org/wiki/LAMP_(software_bundle)

[8] Wikipedia, .NET Framework
http://en.wikipedia.org/wiki/.NET_Framework

[9] Wikipedia, Java Platform, Enterprise Edition
http://en.wikipedia.org/wiki/Java_Platform,_Enterprise_Edition

[10] The 2014 Decision Maker’s Guide to Java Web Frameworks, by Rebellabs
http://zeroturnaround.com/rebellabs/the-2014-decision-makers-guide-to-java-web-frameworks/

[11] **A DBA's Journey in the Sun**, George Trujillo Blog, ORACLE
https://blogs.oracle.com/GeorgeTrujillo/entry/mysql_versus_oracle_features_functionality

[12] DB-engines, provided by “**Solid IT**”
http://db-engines.com/en/ranking

[13] Global Smartphone Market Share Trends - Q1 2013: Android Extends Lead Over iPhone, Windows Phone Performance Mixed by the “The-thoughts”.
http://www.tech-thoughts.net/2013/05/global-smartphone-market-share-trends-android-iphone-windows-phone.html#.UyGjA1LuPIU

[14] Android platform versions, by “Android Studio” developers
https://developer.android.com/about/dashboards/index.html

[15] Wikipedia, Personal Digital Assistant.
http://en.wikipedia.org/wiki/Personal_digital_assistant

[16] CloudBees
http://www.cloudbees.com/

[17] Wikipedia, Object-relational mapping (ORM, O/RM, and O/R mapping)
http://en.wikipedia.org/wiki/Object-relational_mapping

[18] Android capture signature using Canvas and save in png format
http://www.mysamplecode.com/2011/11/android-capture-signature-using-canvas.html

[19] LetsGoMo apps, X-platform Apps
http://letsgomo.com/cross-platform-mobile-apps/

[20] IDC Predicts for year 2014
http://www.idc.com/getdoc.jsp?containerId=prUS24472713

[21] Facebook Debuts Web-Scale Variant Of MySQL
http://www.informationweek.com/big-data/software-platforms/facebook-debuts-web-scale-variant-of-mysql-/d/d-id/1141521

[22] Java EE and Spring by Derrick Kittler, Mauricio “Maltron” Leal, Vamsi Chemitiganti, 2013
http://rhsummit.files.wordpress.com/2013/06/leal_f_1100_java_spring_lovers_quarrel.pdf

[23] Royal Mail homepage
http://www.royalmail.com/

[24] DHL express, express delivery
http://parcel.dhl.co.uk/dhl-service-point/about

[25] TNT delivery services
http://www.tnt.com/express/en_gb/site/home.html

[26] YODEL delivery services
http://www.yodel.co.uk/

[27] The for categories of NoSql databases, by “Rebelling Interactive Consultancy”, May 2013
http://rebelic.nl/2011/05/28/the-four-categories-of-nosql-databases/

[28] Understanding Android Fragmentation with TopicAnalysis of Vendor-Specific Bugs, by Dan Han, Chenlei Zhang, Xiaochao Fan, Abram Hindle, Kenny Wong and Eleni Stroulia
http://softwareprocess.es/a/fragmentation.pdf

[29] Android device and version compatibility froon android.com
http://developer.android.com/guide/practices/compatibility.html

[30] Oracle Java EE technologies
http://www.oracle.com/technetwork/java/javaee/tech/index.html

[31] Linux, Apache, MySql, PHP software bundle
http://en.wikipedia.org/wiki/LAMP_(software_bundle)

[32] Microsoft dot Net Build 2014
http://www.microsoft.com/net

[33] Web development with Ruby on Rails
http://rubyonrails.org/

[34] Django web site
https://www.djangoproject.com/

[35] Spring MVC framework from Spring documentation
http://docs.spring.io/spring/docs/3.2.x/spring-framework-reference/html/mvc.html

[36] Apache Tomcat
http://tomcat.apache.org/

[37] Hibernate website
http://hibernate.org/