A Visual Investigation of the Stack Overflow Dataset

Master's Thesis submitted to the
Faculty of Informatics of the *Università della Svizzera Italiana*
in partial fulfillment of the requirements for the degree of
Master of Science in Informatics
Software Design

presented by
Nicolas Latorre

under the supervision of
Prof. Michele Lanza
co-supervised by
Luca Ponzanelli

September 2015

I certify that except where due acknowledgement has been given, the work presented in this thesis is that of the author alone; the work has not been submitted previously, in whole or in part, to qualify for any other academic award; and the content of the thesis is the result of work which has been carried out since the official commencement date of the approved research program.

Nicolas Latorre
Lugano, 11 September 2015

To my family, my uncle, and whoever believed in me

Remember the two benefits of failure. First, if you do fail, you learn what doesn't work; and second, the failure gives you the opportunity to try a new approach.

Roger von Oech

Abstract

Stack Overflow is a technical Question and Answer (Q&A) website for programmers, where users can ask and answer questions about programming problems or other topics related to computer science.

Stack Overflow contains a huge amount of discussions and it covers a wide range of arguments. Mining the Stack Overflow dataset is computationally expensive. Researches that aim to understand what the important topics for developers are or to understand the popularity trends in Stack Overflow require time.

Several studies have shown that visualization is an ideal way for getting acquainted with large datasets. Inspired by this point of view, we investigate how it is possible to visualize, in an effective way, the dataset provided by Stack Overflow.

In this thesis we present SODA, the *Stack Overflow Dataset Almanac*, an application that allows researchers to visually investigate Stack Overflow. The investigation is done through the analysis of single or co-occurrent tags in different time intervals and frames.

Acknowledgements

Contents

Figures

Tables

Chapter 1

Introduction

Question and Answer (Q&A) websites are means to share technical knowledge, to seek for advice, to compare opinions, or to satisfy one's curiosity [2]. One of the most popular Q&A websites is Yahoo Answers[1], a general-purpose Q&A website that allows users to ask questions related to any topic.

Online resources like technical Q&A websites are more and more leveraged by developers [38]. A prominent example is Stack Overflow[2], where users can ask and answer questions about programming problems or other topics related to computer science. Born in 2008, Stack Overflow has quickly become a fundamental online resource in the daily working life of developers [25], and it is the first and the largest communities of the Stack Exchange[3] network.

Stack Overflow contains a huge amount of discussions and it covers a wide range of arguments. According to the data dump of March 2015 provided by the Stack Exchange network[4], Stack Overflow stores 8.9 millions of questions, 15 millions of answers, and a community of 4 millions users. Furthermore, with more than six thousand questions asked every day [32, 31], and a median answer time of 11 minutes [25], Stack Overflow provides "archives with millions of entries that contribute to the body of knowledge in software development" [41]. Research in software engineering has started to take advantage of the enormous amount of information in Stack Overflow to a large extent. The information is exploited to understand how developers use APIs [6], to build recommender systems [30, 16], or to model and improve post quality [32, 31, 18].

Tagging can help to bridge the gap between technical and social aspects of software development [42], therefore the tagging system of Stack Overflow is another type of data leveraged by research, to provide a general categorization of tags and question types [41], and to store, pre-process, analyze and recommend tags [46]. Another topic of interest is related to popularity and trends, Barua *et al.* presented an analysis of topics and trends in Stack Overflow [7], while Allamanis *et al.* used a topic modeling analysis to categorize questions [4]. Due to its size, mining this dataset is a computationally expensive task. Research works that aim to define a structure from the unstructured contents of Stack Overflow discussions (*e.g.,* popularity trends analysis or topic modeling) require several tools and time.

According to Ware [45], visualization is the ideal way for getting acquainted with large

[1] https://answers.yahoo.com
[2] http://stackoverflow.com
[3] http://stackexchange.com
[4] https://archive.org/details/stackexchange

datasets. Inspired by this point of view, we investigate how it is possible to visualize, in an effective way, the dataset provided by Stack Overflow. Influenced by the work of Kuhn *et al*, which used software cartography to represent a program as a topic-map [23], we approached a solution that used *Latent Dirichlet Allocation* (LDA) [11] to extract topics from discussions, then *Multidimensional Scaling* (MDS) [12] to map them in a two dimensional (2D) space.

This solution achieves good results for a small number of discussions, *i.e.,* 50'000, however it does not scale to millions of discussions. A scalable alternative to map high dimensional data to a lower dimensional space is provided by *random projection* (RP) [19]. The visualization obtained following this approach suffers form a high loss of information, because the reduction from a high dimensional space to a 2D space is too aggressive. Due to these limitations, we decided to take advantage of the available tagging system of Stack Overflow to represent topics and produce a fast and meaningful visualization.

In this thesis we present SODA the *Stack Overflow Dataset Almanac*, a Scala application that allows researchers to visually investigate the Stack Overflow dataset. SODA relies on the available tagging system of Stack Overflow to support the analysis of topics and trends. The investigation is done through an interactive analysis of single or co-occurrent tags in different time intervals and frames. The tool provides an intuitive visualization that helps to understand, in a given period of time, how much a certain tag was used in discussions with respect to its peak occurrence. SODA also shows the evolution of tag occurrences, co-occurrences, and trends through charts.

To validate our approach we present and discuss six stories that have been discovered with SODA. These stories are evidence that SODA can be successfully used as a research tool to investigate the Stack Overflow dataset.

1.1 Contributions

The main contributions of this thesis can be summarized as follows:

- A meaningful and effective visualization of the Stack Overflow dataset based on the available tagging system.
- A tool that helps researchers to investigate Stack Overflow in a reasonable time.
- A collections of evidence that validate our approach.

1.2 Structure of the Document

The rest of the document is structured in different chapters, described in the following.

- **Chapter 2** introduces Stack Overflow and its mechanisms, then it discusses the related work concerning Stack Overflow.
- **Chapter 3** presents SODA, a Scala application that provides an interactive visualization of the Stack Overflow dataset. Specifically, we talk about the architecture of SODA, our visualization approach and the user interface of the tool.
- **Chapter 4** analyzes six stories that we were able to extrapolate with SODA in order to validate our approach.

- **Chapter 5** summarizes and concludes this thesis by going over the contributions of our work. It also presents possible future works to improve and extend SODA.

Chapter 2

Stack Overflow

The purpose of this chapter is to discuss the related work concerning this thesis. Section 2.1 presents an overview of Stack Overflow and its mechanisms. Section 2.2 describes the current state of the art of Stack Overflow researches.

2.1 What is Stack Overflow

Stack Overflow is one of the most popular technical Q&A website for programmers, where they can share programming knowledge to solve problems or discuss about other topics related to computer science. The system is written in c# and it was developed in 2008 by Jeff Atwood and Joel Spolsky.

At the time it was difficult to access technical knowledge online, because a lot of programming information were trapped in forums, buried in online help, or hidden away in books[1]. The goal of the authors was to unlock all that information, by creating a system where users could share technical knowledge about programming, ask a question to a community of experts and get an answer in a short period of time. As Jeff Atwood stated in his blog[2], "Stack Overflow is by programmers, for programmers, with the ultimate intent of collectively increasing the sum total of good programming knowledge in the world".

Stack Overflow was built following nine design decisions[3]: *voting, tags, editing, badges, karma, pre-search, Google is UI, performance, critical mass*. Six of these design decisions are the heart of Stack Overflow and they differentiate it from others Q&A services.

The voting mechanism allows users to approve or disprove questions and answers, therefore the quality level of a post is based on the knowledge of the community. Good answers voted by the community rise to the top of the stack of answers of the related question, thus they can be easily seen by future users of Stack Overflow. The author of a question can accept the answer that was more helpful for him and that answer becomes the first one that users see.

Tagging is a mechanism that provides a structure to the unstructured contents of Stack Overflow discussions, by sorting them into precise categories. A tag is a word which describes the technology or the topic addressed by a question, each question must contain at least one

[1] http://blog.codinghorror.com/introducing-stackoverflow-com
[2] http://blog.codinghorror.com
[3] https://www.youtube.com/watch?v=NWHfY_lvKIQ

tag to a maximum of five tags. Furthermore, experts can be connected with questions that they can answer through the tagging system.

Editing is a core function in Stack Overflow, users can edit a post in order to improve its quality, in this way discussions are not frozen, like it happens in forums. The authors were inspired by Wikipedia[4] and they wanted to create a similar system for programming questions, which means that the community can edit questions or answers in order to make them better and better. In this way a question can become a reference for future users of Stack Overflow to study a particular problem.

Badges come in three level: gold, silver, and bronze. They are like achievements in video games, encouraging users participation by rewarding positive behaviors. Badges are also a symbol of credibility in Stack Overflow, because they are evidence of users commitment.

Karma acts like reputation and it is represented by points. Users can earn points by answering questions, voting questions, editing posts, *etc*. Karma stimulates users contribution to the community, because when they reach specific levels of karma they unlock special privileges like voting, commenting posts, or editing posts of other users.

Pre-search aims to prevent duplicated questions at creation time. When a user writes a question, entries with similar titles are shown to him, such that he can solve his problem by consulting already asked questions.

2.2 Related Work

Several studies have revealed the importance of social media resources in software engineering. Begel *et al.* suggested that the use of social media in software development teams improves communication and coordination [9], while Storey *et al.* discussed how social media influence software development practicies [38]. Treude *et al.* found that basic tagging mechanisms can be effectively used to bridge the gap between technical and social aspects of software development (*e.g.,* to support informal processes) [42].

Due to the prominent role of Stack Overflow, the valuable source of data represented by its dataset has been exploited in a remarkable number of researches. The work of Treude *et al.* [41] is one of the first conceptual explorations of the Stack Overflow data, they identified general categories of tags used by the community of Stack Overflow: programming language, framework, environment, domain, non-functional, and homework.

Finding structures out of unstructured software repositories is an hot topic of research in software engineering, Thomas *et al.* proposed to use statistical topic models to accomplish this goal. It is not surprising that a considerable number of researches in Stack Overflow take advantage of topic modeling techniques (*e.g.,* LDA) to recover topics out of Stack Overflow discussions and analyze what developers are talking about. Allamanis *et al.* constructed a topic model by means of LDA to connect programming concept to question types, discovering that the types of questions asked in Stack Overflow do not vary across programming languages [4].

Barua *et al.* used LDA to automatically discover the main topics present in Stack Overflow discussions. They analyzed a total 3,474,987 posts and discovered that mobile application development is on the rise, faster than web development, in particular Android and iPhone development are predominant than Blackberry [7]. Campbell *et al.* combined Stack Overflow questions related to PHP and Python as well as the documentation of projects developed with

[4] https://www.wikipedia.org

both languages. Then, they applied LDA to reveal topics that were inadequately covered by project documentation [14].

Linares *et al.* used topic modeling techniques to extract topics from Stack Overflow questions related to mobile development. They detected that most of these questions include topics related to general questions and compatibility issues [24]. The work of Wang *et al.* analyzed contents of tens of thousands of Q&A by means of LDA, and they noticed that five main topic emerge from them: user interface, stack trace, large code snippets, web documents, and miscellaneous [43]. Riahi *et al.* and Chang *et al.* used topic model techniques to build a recommender systems which routes questions to experts [34] or to a group of experts [15].

There are several research works that exploited the data of Stack Overflow to build recommender system. For example there are projects [30, 16, 33] that aims to bridge the gap between IDEs (Integrated Development System) and online resources by means of recommender system, so that developers do not have to interrupt their programming tasks. Saha *et al.* proposed an automatic system to recommend the correct tag to users when they ask questions in Stack Overflow [35]. Two other works which focus on tag recommendation is [44], which proposed an automatic tag recommender based on historical tag assignments by means of LDA, and Xia *et al.* which aims to store, pre-process, analyze and recommend tags [46].

Another field of research worth of mention are those that aims to predict interesting events in Stack Overflow. Asaduzzaman *et al.* conducted a study about unanswered questions, they built a classifier that predicts how long a question will remain unanswered in Stack Overflow [5]. Correa *et al.* analyzed one million of closed Stack Overflow questions and created a model to predict them [17]. Stanley *et al.* built a tag prediction system to suggest suitable tags for Stack Overflow posts [37].

The Stack Overflow dataset can provide insights about project documentation and APIs usage. For example Kavaler *et al.* analyzed the use of Android API classes in over 100,000 applications, then they combined the corresponding Stack Overflow questions and found out that complex class are the most questioned in Stack Overflow [22]. Parnin *et al.* presented a study to measure the effectiveness and completeness of API documentation via social media. They demonstrated that a social media like Stack Overflow plays an important role in software documentation [28].

A different topic of research focuses on the analysis of code snippets present in Stack Overflow discussions. Nasehi *et al.* discovered that code examples and the accompanied explanation are two inseparable elements of good answers [27]. Subramaniana *et al.* presented two studies about code analysis in Stack Overflow posts. The first one analyzed code snippets of accepted answers related to Android API to identify API usage [39]. The second study aimed to link source code example to API documentation [40].

User interactions in Stack Overflow are another interesting topic of research. Bazelli *et al.* analyzed question and answers to identify the personality of the developers [8]. Bosu *et al.* presented an empirical study about building reputation and a guidance to earn high reputation scores quickly [13]. Grant *et al.* showed how badges influence user behavior [20]. Morrison *et al.* discovered a correlation between age and reputation in Stack Overflow [26]. There are also researches that aim to model and improve post quality [31], [32], [18].

Visualization research can reveal interesting aspect, like popularity and trends, of large dataset. For example Achananuparp *et al.* captured real-time information about trends and topics in microblogs (*e.g., Twitter*[5]), leveraging also visualization techniques [1]. In a different

[5] https://twitter.com

context, Bissyande *et al.* [10] studied the popularity of programming languages by considering around 100,000 open-source projects, and visualized its result by means of graphs.

In Stack Overflow there are several works that tried to visualize the dataset. The work of Parnin *et al.* studied how documentation can emerge from crowd knowledge, they also provided treemap in order to visualize the Android API classes discussed in Stack Overflow [29]. Akoglu *et al.* presented OPAVION, a system that allows to find patterns and anomalies in a community of users [3] and visualize them using charts and graphs. Huron *et al.* proposed a new design metaphor called *sedimentation* to visualize large dataset. They have applied this technique to the Stack Overflow dataset to visualize the most popular questions in the first 10 tags [21]. Schenk *et al.* analyzed Stack Overflow as an information market for software engineering knowledge in which the goods that are exchanged are answers to questions and the rewards are score points and badges that contribute to a users reputation. Then they produced a geolocation visualization in order to see how the region of the world contribute to the knowledge sharing in Stack Overflow [36].

Another interesting visualization of the Stack Overflow dataset is provided by Ekisto[6], a project developed by Alex Dragulescu in 2013. As the author said, Ekisto tries to imagine and map our online habitats using graph algorithms and the city as a metaphor. It aims to visualize the habitat of three online communities: Stack Overflow, Github and Friendfeed. In the case of Stack Overflow, Ekisto use cosine similarity between tags of posts contributed by users to place them in a 2D space, then it use the 3D dimension to show users reputation.

2.2.1 Stack Overflow Contest

In October 2012 the Stack Overflow staff organized a visualization contest to find interesting and informative ways of making sense of the mountains of interesting data in the Stack Overflow dataset. The winner of this competition was the aforementioned work of Parnin *et al.* [29]. Now we discuss submissions to the contest that are related to our work.

The first visualization has been submitted by Shubhanshu Mishra and is called "Stack Exchange Network Analysis with focus on growth of Stack Overflow"[7]. This submission contains several visualizations, but we considered only those related to tag visualization, which are two. The first is a D3[8] chart that depicts the top ten tags in Stack Overflow and their relationships. Tags are arranged in the circumference of circle, and their size is proportional to the importance of their relationships. The relationship between two tags is showed by means of undirected edges, where their thickness reveal the importance of the relationship. From this representation we can see that `c#`, `javascript`, and `jquery` have an high number of relationships. The second visualization is a plot that depicts the amount of discussions tagged with the top ten tags over time in Stack Overflow. The plot shows that that `java`, `android`, `c#`, `javascript`, `php`, `jquery` are the most important tags and their trends keep growing. Instead `c++`, `iphone`, `asp.net`, and `.net` after an initial growth, are stable

The third visualization has been submitted by Yin Zhu and is called "Tag Popularity Over Time"[9]. The goal of this visualization is to depicts the popularity of Stack Overflow tags over time. The visualization considers monthly time intervals and by means of a six shades of blue, it represents the rise and fall of 87 tags that have been in the top 50 at least once during a quarterly.

[6] http://ekisto.sq.ro
[7] https://www.kaggle.com/c/predict-closed-questions-on-stack-overflow/prospector#213
[8] http://d3js.org
[9] https://www.kaggle.com/c/predict-closed-questions-on-stack-overflow/prospector#188

The representation depicts that `c#` and `java` are always popular tags, while `javascript` and `android` are rising fast.

The fourth visualization has been submitted by Piotr Migdal and is called "256 Tags of Stack Overflow: Map and Top Tens"[10]. This visualization is a graph which represents the first 256 tags of Stack Overflow, where nodes are tags, edge between nodes represent co-occurrences between them, and colors represents clusters.

The fifth visualization has been submitted by Christian Jauvin is called "Tag Similarity Based on Word Use Patterns"[11]. This submission represents with a matrix the semantic similarity between words found in questions and the most frequent tags. Nearby columns in the matrix should correspond to similar, or related tags.

2.2.2 Graph Overflow

The Graph Overflow[12] website contains some data visualizations of Stack Overflow. Three of them visualize tags informations and then are related with our work. The first visualization[13] of Graph Overflow represents the first 60 tags of Stack Overflow arranged in circumference, where the size of tag is determined by the number and the importance of their relationships. A relationship between two tags is represented with a flow and its thickness reveals the importance of the relationship. The visualization provides also a list of tags sorted in descending order which depicts the number of discussions tagged with a particular tag. This representation shows that `c#` and `javascript` have more relations than `java`. Moreover `c#` is the most important tag, followed by `java` and `javascript`.

The second visualization[14] is an interactive heat map of questions asked in a specific year, against the 60 most relevant tags. They can be sorted in descending order respect to their number of questions asked in each year. The third visualization[15] is an animation which shows how tags have growth in Stack Overflow by means of growing bubbles. The size of bubble is determined by the number of questions tagged with the corresponding tag. The visualization shows that initially `c#` grows fast, then it is overtaken by `javascript` and `java`.

[10] https://www.kaggle.com/c/predict-closed-questions-on-stack-overflow/prospector#211
[11] https://www.kaggle.com/c/predict-closed-questions-on-stack-overflow/prospector#199
[12] http://graphoverflow.com/
[13] http://graphoverflow.com/graphs/stackoverflow-tag-relations.html
[14] http://graphoverflow.com/graphs/stackoverflow-tag-trending-yearly.html
[15] http://graphoverflow.com/graphs/stackoverflow-tag-growth-rate-monthly.html

Chapter 3

SODA

In this chapter we present SODA, the Stack Overflow Dataset Almanac. Section 3.1 discusses the architecture of SODA, section 3.2 explains how we used the data of Stack Overflow in SODA. Section 3.3 analyses the packing strategy adopted to visualize the dataset. Finally section 3.4 presents the User Interface (UI) of SODA.

3.1 The Architecture

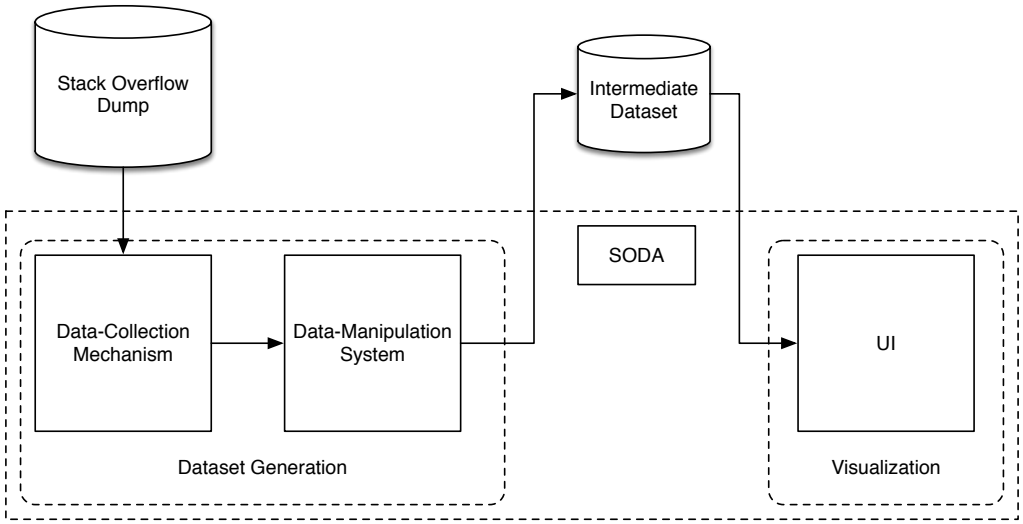

Figure 3.1. The architecture of Soda.

SODA is written in Scala and it is composed by three main components: a *data-collection mechanism*, a *data-manipulation system*, and an *user interface*. Figure 3.1 presents the architecture of SODA. Two main parts can be identified, the first part is called *dataset generation* and it is the most expensive one, and it is composed by the data-collection mechanism and the data-manipulation system.

The former allows SODA to interact with the database that stores the Stack Overflow dump in order to retrieve questions, the latter manages the data to create and store to disk an intermediate representation of the dataset. This representation must be generated only one time, for example when a new version of the Stack Overflow dump is available. Once the intermediate representation of the dataset is available, it can be used by SODA directly at every run. This leads to a remarkable speed up in the performance of SODA.

The second part is composed by the user interface, which implements a Model View Controller (MVC) design pattern to represent the data. The model harnesses the intermediate representation to generate a tree hierarchy of tags, which is then used as a base for the visualization of the data. The controller responds to the interactions of the user with GUI and the view is responsible to display the data.

3.2 The Data

We exploited the data from the Stack Overflow dump of March 2015, which stores 8.9 millions of questions, 15 millions of answers, and a community of 4 millions users. The main documents retrieved from this dump are Stack Overflow questions, and for each of them we considered the following information:

- **Id**. The unique identifier of a question.
- **Creation Date**. The creation date of a question in the format *year*, *month*, *day*.
- **Score**. The number of votes earned by a question.
- **View Count**. The number of views related to a question.
- **Owner Used Id**. The unique identifier of the author of a question.
- **Title**. The title of a question.
- **Tags**. The tags associated with a question.
- **Answer Count**. The number of answers connected to a question.

Tags are processed separately, SODA considers the first tag of every discussion of Stack Overflow, and for each of them it computes the total amount of discussions where they appear as the first tag. After that the tool considers the first two tags of every discussion and it repeats the computation. This process is repeated for three more times, in order to consider the first three tags, then the first four tags and finally five tags, which is the limit of Stack Overflow. For example SODA starts by computing the total amount of discussions where `java` appears as the first tag, then in a second time it takes into account discussions that start with "`java android`" or "`java swing`". For each set of tags SODA stores the id and the creation date of their corresponding discussions.

Then the information related to tags and to discussions is stored to disk. This is an intermediate representation of the dataset, composed by two set of Comma-Separated Values (CSV) files. The first set stores the information related to tags, and the second set stores the information related to discussions. The creation of this dataset increases the performance of SODA because it does not have to query directly the Stack Overflow dump at every run. Each CSV

file in both sets is named after a single tag computed in the beginning by SODA. For example the file named `jquery.csv` contains the information related to the single tag `jquery` and also about co-occurrences of tags that starts with `jquery` (*e.g.,* `jquery html`).

SODA exploits the intermediate representation of the dataset to generate a tree which depicts the hierarchy of tags in Stack Overflow. In order to build this tree, the tool takes into account the files related to tags in the dataset. Every single tag is a child of the root and their subtrees are built using the information provided by the corresponding file. The Listing 3.1 shows an abstract representation of a branch related to the single tag `java`. This tag is a child of the root, and its children are co-occurrences of two tags where the first is `java`. For example the co-occurrence of tags composed by `java android` is a child of `java`, and its children are co-occurrence of three tags where the first two are `java` and `android` respectively. It follows then that the first level of the tree is composed by single tags (*e.g.,* `java` or `python`), while the second level of the hierarchy is composed by co-occurrences of two tags (*e.g.,* "`java swing`" or "`php mysql`"). The tree can reach a maximum depth of five levels, and the last level represents co-occurrences of five tags, which is the greatest number of tags allowed in Stack Overflow to label a question.

```
java (803853)
  java android (104478)
    java android eclipse (7106)
      java android eclipse google-maps (118)
        java android eclipse google-maps geolocation (1)
        java android eclipse google-maps google-maps-marker (1)
        java android eclipse google-maps import (2)
        java android eclipse google-maps goolge-play-service (5)
        ...
      java android eclipse android-listview (38)
        ...
      ...
    ...
  ...
```

Listing 3.1. Java Hierarchy

Every node in the tree carries a value, which symbolizes the amount of discussions tagged with the corresponding set of tags. The root has value zero because it does not represent any tag and the amount of tagged discussions for a node depends also on the value its children. The co-occurrence of tags "`java android eclipse google-maps`" in the Listing 3.1 has a value of 118. Now suppose that the children of this node are those depicted in the listing, this would means that there are 109 discussions exactly tagged with "`java android eclipse google-maps`" and the remaining discussions is provided by the value of the children. This node contributes to the value of the node represented by "`java android eclipse`", which is 7106.

3.3 Data Visualization

To visualize tags, SODA uses a square packing strategy and the structure of the packing is depicted in Figure 3.2. The tool starts form the first level of the hierarchy tag tree and depicts the initial visualization which is composed by single tags. The main idea is to place the most

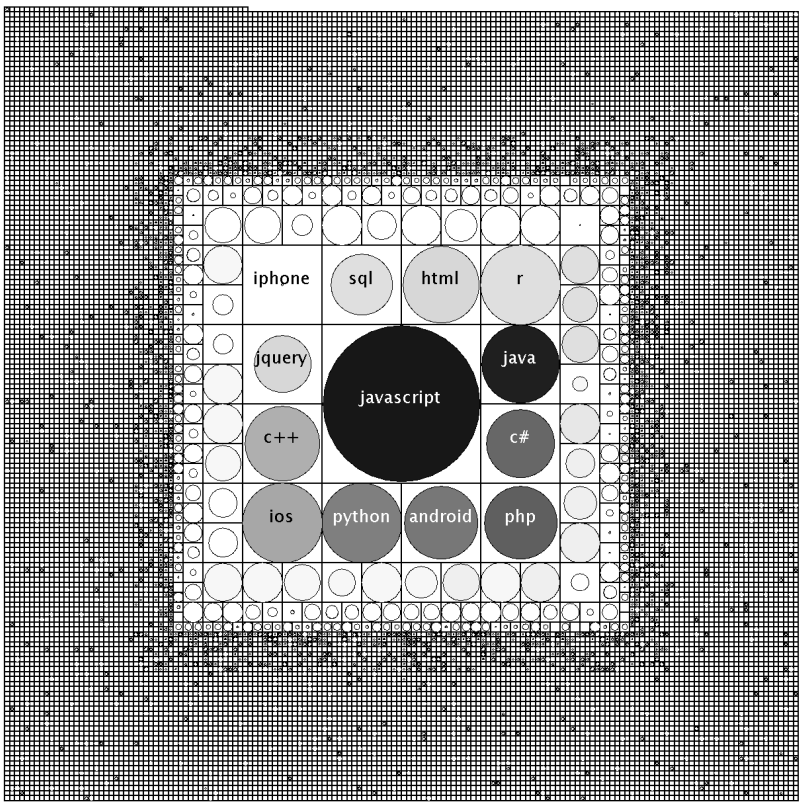

Figure 3.2. The main view of Soda with the structure of the square packing.

important and largest element in the center as a square of fixed size. Then a spiral develops in clockwise order from the right side of the central component, the size of the new elements is cut by half at every iteration of the spiral. For example in Figure 3.2 the `javascript` tag is the central member of the packing, which means that it is the most important element in the dataset. Then a spiral develops in clockwise order from the right side of `javascript`, and it begins with `java`, `c#`, `php`, *etc*. The size of these elements is cut by half respect to `javascript`. The first iteration ends with the `r` tag in the right top corner, after which a new spiral begins and the size of the new members is cut by half.

SODA represents the evolution of Stack Overflow in time intervals and to determine the order in which tags appear in the spiral, it computes the occurrences of tags for every time interval; the default time interval is one week. A peak occurrence is then determined, which represents the maximum amount of discussions tagged with a specific tag among all the time intervals. Tags are then sorted in descending order according to this peak and this determines the order in the spiral. In Figure 3.2 the `javascript` tag is the central element of the packing, meaning that it reached the peak as the most discussed tag per week ever, and there exists no other tag in Stack Overflow that in any week had more corresponding tagged discussions. In this way if a tag is popular it is located in the center or close to it, otherwise it is far away.

Every square in the packing contains a circle that represents the importance of a tag by means of size and color. The size of a circle is proportional to the number of discussions tagged with the corresponding tag in the current time interval. A circle perfectly fits its assigned square when the respective tag reaches its relative peak. For example in Figure 3.2 the circle related to the `python` tag completely fits its assigned square, which means that `python` in the corresponding time interval has reached its relative peak in the history of Stack Overflow. On the other hand, the circle related to the `sql` tag does not occupy entirely the associated square, because its relative peak occurs in another time interval.

The color of a circle is determined by a 30 level greyscale, computed in function of the ratio between the occurrence of discussions tagged with a specific tag and the peak of the central tag. The darker the color, the closer the number of discussions in a particular time interval is to the peak of the central tag. Figure 3.2 shows that `javascript` is the darkest element in the packing, and it is the only one that eventually will be completely black (*i.e.,* when it reaches its peak). Tags `ios` and `r` have reached their relative peak (their circles perfectly fits their associated squares), but their amount of discussions in the current time interval is lower than the peak represented by `javascript` (since their color is light grey). Many tags in the outer layers have circles that are perfectly inscribed in their assigned square, but they are essentially filled with white or very light shades of gray. This case corresponds to the fact that these tags are being discussed as much as their peak, but the respectively absolute amount of discussions is significantly lower than the most popular tag `javascript`.

3.4 User Interface

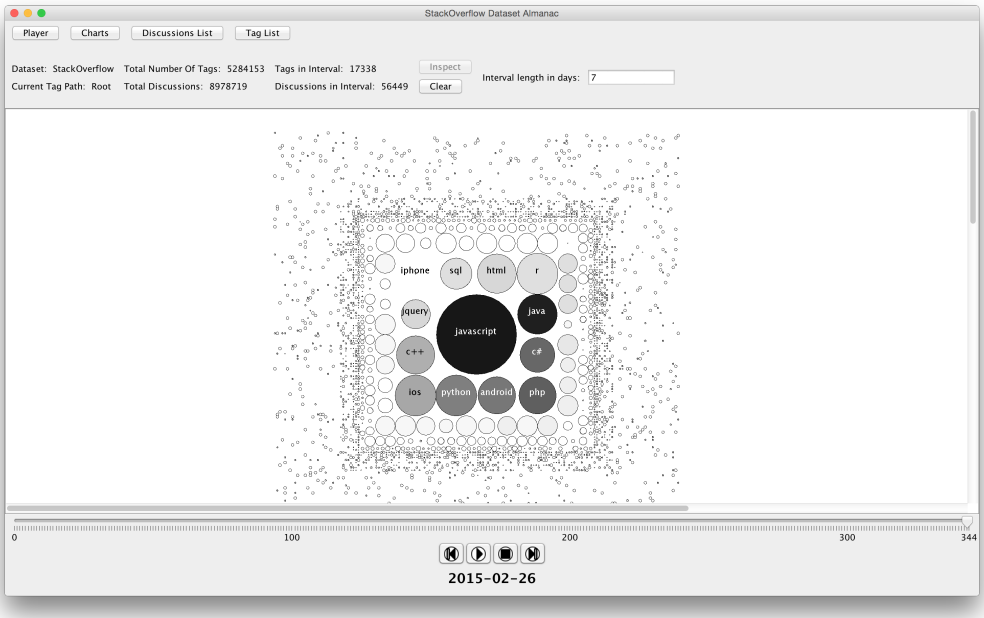

Figure 3.3. The main user interface of SODA.

Figure 3.3 shows the main user interface of SODA that is composed by three components: a *toolbar* on the top, a *tag view* in the middle, and a *timeline slider* with a *player* on the bottom. We now explain in detail the main functionalities of each component.

3.4.1 Toolbar

On the top of Figure 3.3 there is the toolbar of SODA. In the top left corner of the toolbar there are four buttons: *Player*, *Charts*, *Discussions List*, *Tag List*. The player button switches on/off the visibility of the player in the bottom, in order to gain space in the tag view; by default the player is visible. The charts button toggles a list of three buttons: *Line Chart*, *Bar Chart*, and *Area Chart*). These buttons allow users to plot the actual amount of discussions tagged with a particular set of tags over time by means of a line, bar, and stacked charts; by default they are not visible. The discussions list button shows or hides a table above the player, which contains the list of questions in the current time interval related to the corresponding tag visualization. Figure 3.4 depicts a partial list of questions related to `javascript` in the last week of Stack Overflow; by default the list is not visible and when the view is at the top of the tag hierarchy tree, as in Figure 3.3, no discussion is available. Discussions in the list are sorted in descending order according to their score and for each of them SODA provides the following information: id, title, creation date, number of answers, score, number of view, owner user id. When a user clicks on a particular discussion, SODA opens the corresponding Stack Overflow discussion in the browser.

ID	Title	Creation Date	Answers	Score	View	Owner
28655044	Why replacing constructor functi...	2015-02-22	2	4	41	1793574
28648090	Properties vs. Keys vs. Values in...	2015-02-21	2	4	54	4096007
28640123	Can you explain object and arra...	2015-02-20	1	4	56	2442514
28672871	Understanding javascript void	2015-02-23	1	3	38	2421990
28725112	Do getters and setters only work...	2015-02-25	1	2	23	38522
28687912	Why can't a reference an old obj...	2015-02-24	2	2	29	2653967
28689484	Automatic executing function in J...	2015-02-24	3	2	91	1421110
28704164	Check if an HTML file is online o...	2015-02-24	1	2	32	4136520
28693773	Call a Javascript function stored ...	2015-02-24	1	2	28	521554
28660868	Javascript animation does not b...	2015-02-22	2	2	46	4594233
28646574	Javascript function does not wor...	2015-02-21	4	2	26	4591330

Figure 3.4. A partial list of discussion for javascript.

The tag list button toggles a search box and a list of tags sorted in alphabetical order for the corresponding view. In Figure 3.2 we can see that SODA labels with a name only the more important tags, in order to maintain the view clean. This can be a potential problem when users need to search for a particular tag, therefore SODA provides a search function to address this problem. Figure 3.5 shows the list of tags and the search field, the list is update as soon as the user starts typing in the search box. From this list users can select or deselect a tag and their choices are reflected in the tag view.

Under the aforementioned buttons we can find some metrics relative to the Stack Overflow dump. First of all we have the name of the dataset (*i.e.,* Stack Overflow), then below it is reported the current tag path which represents the current position of the view in the tag hierarchy tree. Figure 3.3 represents the top of the hierarchy, and the current tag path is set to `root`, while in Figure 3.5 we can see that the current tag path is associated with the "`java swing`" view. The second column reveals the total number of tags and the total number of discussions in the Stack Overflow dump. The third column shows the number of tags in the current level of the tag hierarchy tree, while below it is reported the current number of discussions in the current tag view respect to the time interval.

Moving on in the toolbar we find two buttons: *Inspect* and *Clear*. These buttons are associated with the selection mechanism. The inspect button allows users to focus the view only on

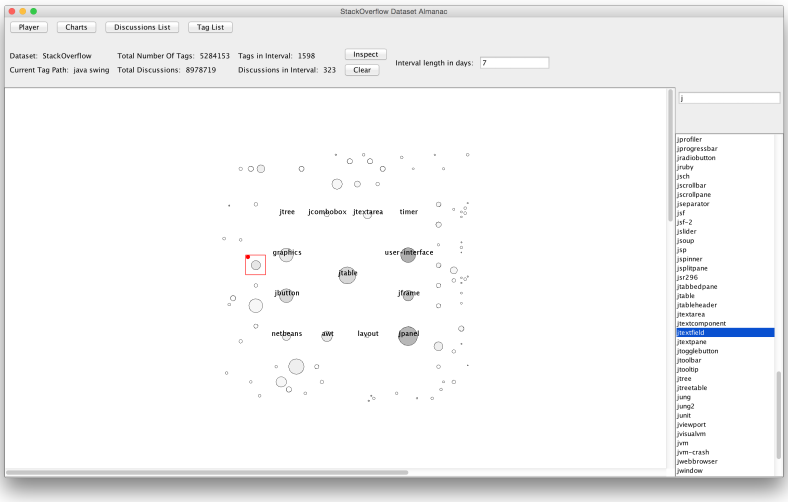

Figure 3.5. The list and search box where users can search for tags related to the view `java swing`.

the tags that they have selected, and it is enabled only if at least one tag is selected in the view. The square packing is maintained when users do inspections. The clear button has the purpose of clearing the selections that the users made, without the need of deselect each tag one at a time in the view; the same function is accomplished by pressing the `C` key.

Finally at the end of the toolbar there is the time interval field. The time interval in SODA is defined in days and the default value is seven, but users can change this value by specifying a positive integer. For example Figure 3.6 shows the first week of life of Stack Overflow while Figure 3.7 depicts its first three months of life. When a user changes the time interval, SODA recomputes the occurrences of tags in each period of time according to the new time interval, then it updates the tag view, the timeline slider, and the metrics with the new data.

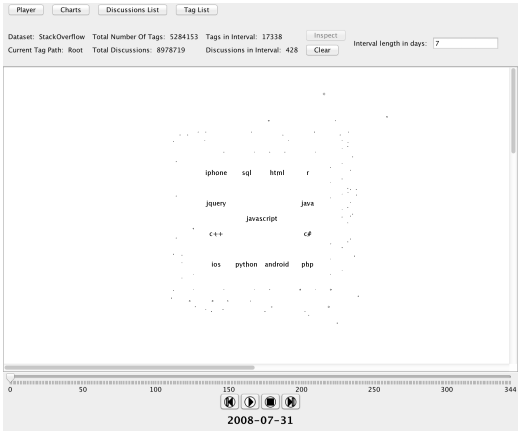

Figure 3.6. First Week.

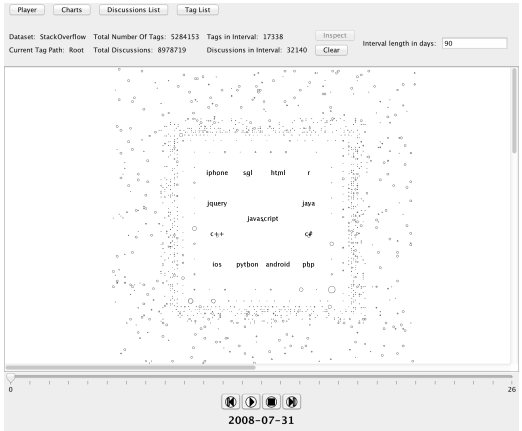

Figure 3.7. First Three Months.

3.4.2 Tag View

The visualization provided by SODA is not static, users can interact with the tag view in several ways. For example users can drag the square packing around the canvas and they can zoom in or zoom out. Users can toggle the structure of the packing at any moment by pressing the `M` key.

Double clicking on any tag allows users to move down of one level in the related tag hierarchy tree, which depth for every tag is at most five. If a user wants to go up of one level it has to press the `B` key. Figure 3.8 shows the tag view of SODA in the last week of Stack Overflow, this view corresponds to the the root of the tag hierarchy tree, and it displays the amount of discussions tagged with a single tag. If a user double clicks on the central tag `javascript`, SODA changes the main view accordingly. Figure 3.9 depicts the new level which represents discussions tagged with two tags, where the first is `javascript` and the second is another tag that co-occur with it (*e.g.,* `html`).

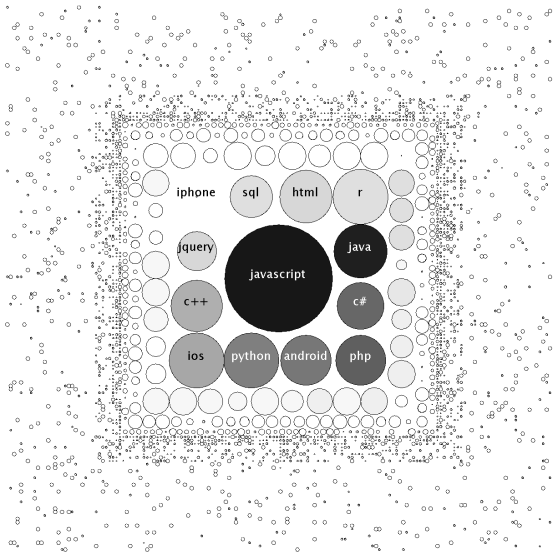

Figure 3.8. Root.

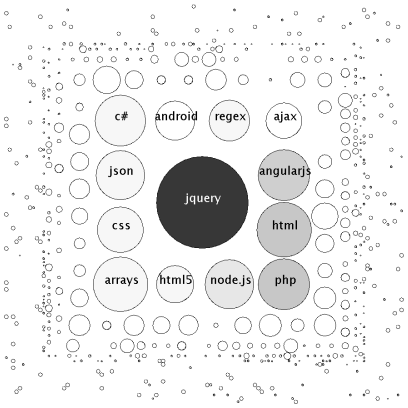

Figure 3.9. Javascript world.

When users leave the mouse over a tag for some seconds, metadata information about the tag is depicted to users. This information is composed by:

- The current tag path and the name of the tag.
- The current amount of discussion related with the corresponding tag.
- The total number of discussions related with the corresponding tag.
- The available number of discussions tagged exactly with the corresponding tag if we decide to move down in its level.

SODA provides the possibility to select tags, users can select a single tag or multiple tags. As you can see from Figure 3.5, a selection is highlighted in the view by a red square with a red

dot in the left top corner. The selection mechanism allows users to perform specific analysis on a subset of tags and the view of SODA can be changed to display only such subset. Users can select a tag and discover the tags that ever co-occur with it by pressing the A key. These tags are highlighted with a purple square, as depicted in Figure 3.10.

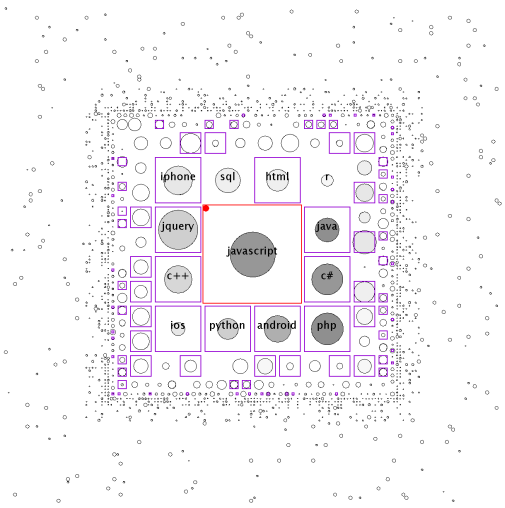

Figure 3.10. The relationship between javascript and the other tags.

3.4.3 Timeline Slider and Player

SODA allows users to observe the evolution of the Stack Overflow dataset, which starts from August 2008 until the first week of March 2015. The timeline slider provides visual clues about the temporal position in the dataset, and each thick represents a time interval in the dataset. Users can move the slider at any point of the evolution and start the animation from there. When users change the time interval value in the toolbar, the timeline slider is updated accordingly. In Figure 3.6 and Figure 3.7 we can see how the timeline slider changes from a weekly interval to a quarter interval. If we increase the interval size, the thicks in the timeline decrease, instead they increase if we reduce the interval size.

The player is composed by four buttons: *start*, *play*, *stop*, *end*. The start button resets the timeline at the beginning of the evolution. The play button initiates the animation, while the stop button arrests it. The end button forces the timeline to the end of the evolution. Furthermore under the player there is a date field which shows the initial date of the current time interval; this field is updated in real time during the animation or when a user drags the timeline slider. Figures 3.11, 3.12, 3.13, 3.14, 3.15 present in chronological order five snapshots of the single tags view taken at different period of time.

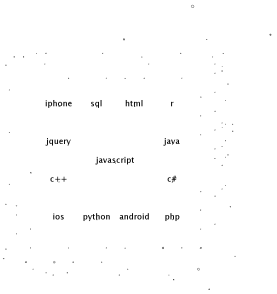

Figure 3.11. Stack Overflow dataset at the beginning.

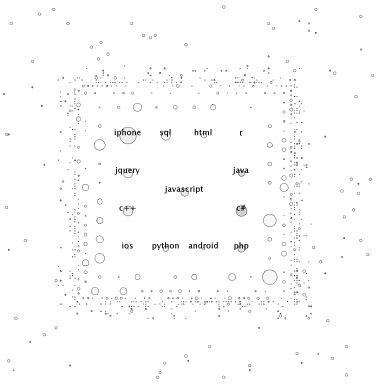

Figure 3.12. Stack Overflow dataset at 25% of life.

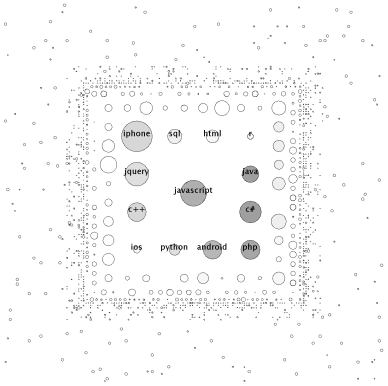

Figure 3.13. Stack Overflow dataset at 50% of life.

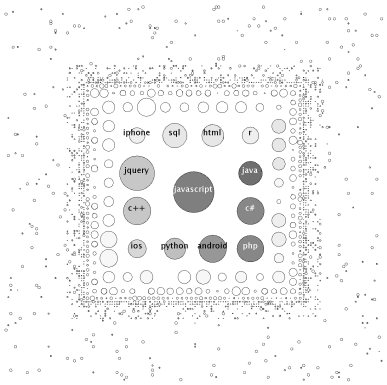

Figure 3.14. Stack Overflow dataset at 75% of life.

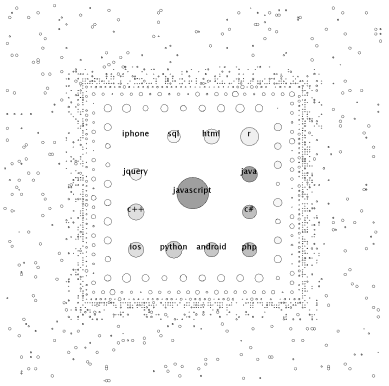

Figure 3.15. Stack Overflow dataset at the end.

Chapter 4

Stories

In this chapter we present a set of stories that we discovered using SODA. Section 4.1 shows how SODA is able to detect and visualize abandoned tags. Section 4.2 depicts developers migration in "iOS" operating system versions, while section 4.3 presents trends of database platforms in Stack Overflow. Section 4.4 compares javascript web frameworks. Section 4.5 presents trends in programming languages and section 4.6 shows version control software trends in Stack Overflow.

4.1 iOS and iPhone

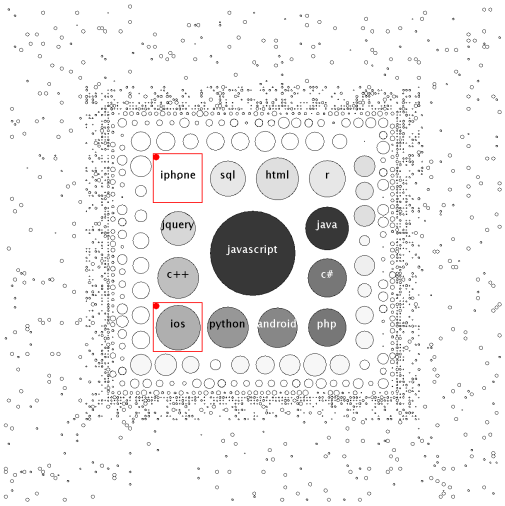

Figure 4.1. The abandoned `iphone` Tag

The first story that we report is about the `iphone` tag and the `ios` tag. Figure 4.1 depicts these tags in the last week of the Stack Overflow dump. Almost every tag around the center is still active, except the one in the upper left corner, *i.e.,* `iphone`. If a tag has ever been popular,

meaning that it reached a significant (weekly, in this case) peak in the history of Stack Overflow, it occupies a significant square in the visualization. If for some reason it became less popular, that fact would result (in more recent times) in almost-empty locations. As we can see from Figure 4.1, this is the case of the `iphone` tag. In June 2010 Apple introduced the name "iOS" for its iPhone operating system, then developers abandoned the `iphone` tag and started using the new tag `ios`. In the lower left corner of Figure 4.1 we can see how the `ios` tag is still active.

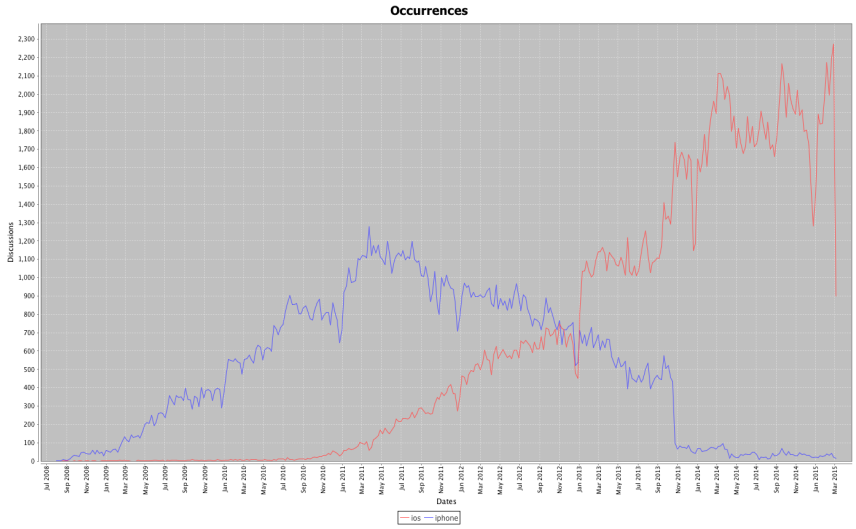

Figure 4.2. Line Chart that compares `iphone` tag and `ios` tag.

The line chart in Figure 4.2 represents the number of discussions, per week, tagged with `iphone` or `ios`. The plot reveals that the growth of `iphone` is linear until April 2011, after which it starts to decrease. On the other hand `ios` begins growing from June 2010. This discovery is a demonstration of how the community of Stack Overflow adapted to the change introduced by Apple. The growth of `ios` has a remarkable peak in September 2013, while `iphone` goes throught a dramatic drop and it is almost abandoned. We suppose that this event is related to the release of "iOS7" by Apple in September 2013.

Figure 4.3 depicts the same information provided by Figure 4.2 as a stacked chart. We can see from this chart that initially `iphone` has a growing number of discussions, however when `ios` starts rising, `iphone` suffers a decline, until it becomes almost inexistent after September 2013.

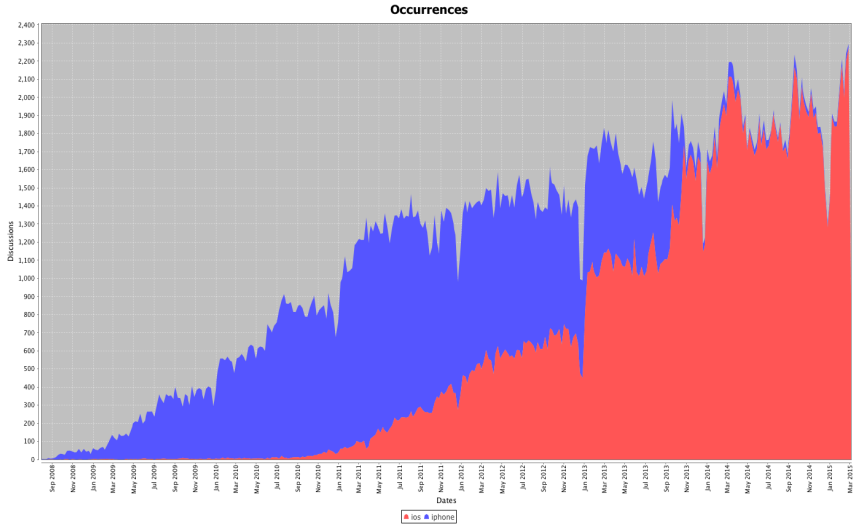

Figure 4.3. Stacked Chart that compares `iphone` tag and `ios` tag.

4.2 iOS versions

The second story that we present is related to the `ios` version tags that co-occur with the `ios` tag, starting from `ios4` until `ios8`. Figure 4.4 represents these tags in last week of Stack Overflow. We can see that `ios4`, `ios5`, and `ios6` are almost abandoned and their popularity is not as high as those of `ios7`, and `ios8`. It seems that also `ios7` is going to be abandoned, while `ios8` is still active.

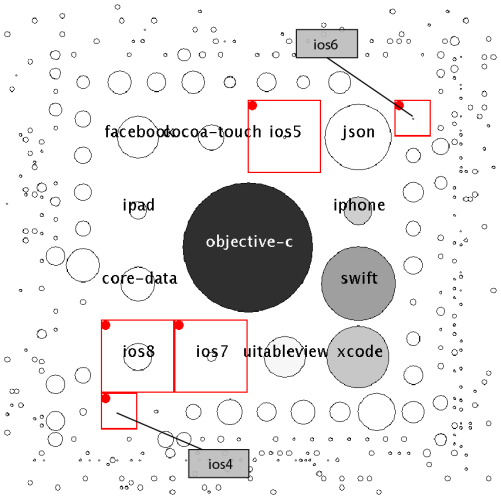

Figure 4.4. `ios` version tags that co-occur with the `ios` tag.

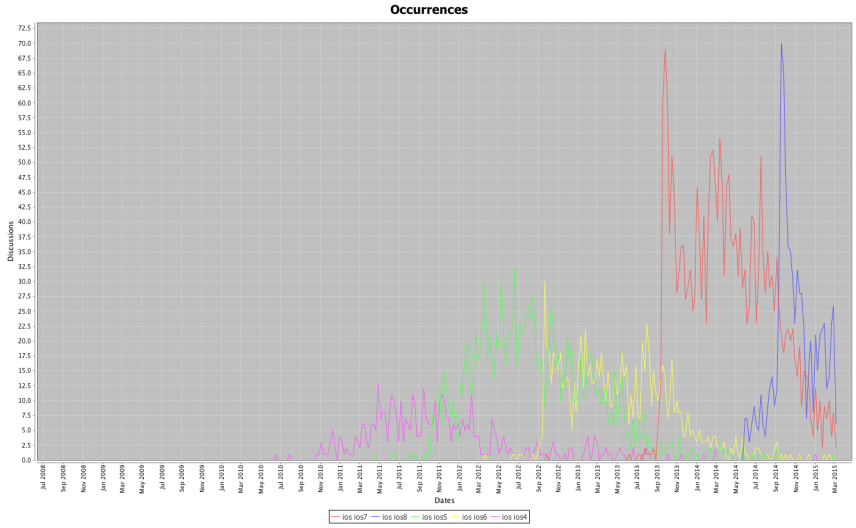

Figure 4.5. Line Chart of `ios` version tags that co-occur with the `ios` tag.

Figure 4.5 depicts a line chart that represents the number of discussions, per week, tagged with `ios4`, `ios5`, `ios6`, `ios7`, or `ios8`. The chart shows that `ios4` did not reached a high peak of popularity in Stack Overflow. After its release in June 2010, it grows slowly until May 2011, after which it stabilizes until October 2011, when it starts decreasing. This date corresponds to the release of `ios5` that is twice as popular respect to `ios4`. `ios5` grows until September 2012, then it begins to decrease due to the release of `ios6`. The latter after an initial peak takes a constant trends until September 2013, when it starts to decline, since the release of `ios7`. It appears that `ios6` is less popular than `ios5`, in fact we can see in Figure 4.4 that the former is more distant from the center respect to the latter. This could be due to the fact that `ios6` replaced the Google-Map application with its own version, which had a lot of problems. Apple received a great number of critics about this issue, then maybe for this reason developers decided to wait before migrating to the new operating system.

`ios7` starts with an high peak, after which it stabilizes and starts to decrease around September 2014, when `ios8` is released. `ios6` and `ios7` present a similar pattern, there is an initial peak followed by a drop and then another rise that tends to stabilize. The peak could be the result of the initial learning curve faced by developers with a new operating system. The drop occurs around December, so it could be attributed to the Christmas holidays, thereafter there is a new growth that tends to stabilize. We can observe that `ios8` follow a similar pattern, so we can assume that after the second growth, it will stabilize.

Figure 4.6 shows the same information provided by the line chart using stacked chart. We can clearly see from this chart how the amount of discussions related to a particular operating system decreases when the new version is released and hits Stack Overflow. We can recognize a pattern from both charts, developers migrate to newer version of the iOS operating system as soon as their are available, and the old version is eventually abandoned. The remarkable difference in popularity between `ios7` and the previous versions could be linked to the fact that this version introduced a new interface and a great number of features, but it could also attest the increasing importance of Stack Overflow as online resource.

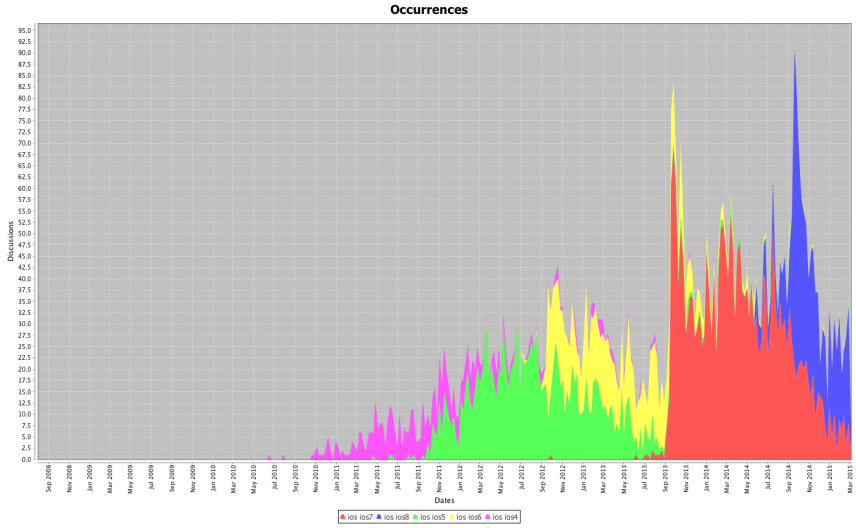

Figure 4.6. Stacked Char fof `ios` version tags that co-occur with the `ios` tag.

4.3 Database Platforms

The third story that we discuss regards the trends of database platforms in Stack Overflow. We considered the following tags: `mysql`, `sql-server`, `mongodb`, `oracle`, `postgresql`, `sqlite`, `db2`. Figure 4.7 shows them in the last week of the Stack Overflow dump. We can see that tags related to database platform are not in the top ten most popular tags. `mysql` is the most important platform, followed by `sql-server` and `mongodb`. Then we find `oracle` and `postgresql`, which are not so popular as the previous platform. Finally there are `sqlite` and `db2` which are not much discussed by Stack Overflow users.

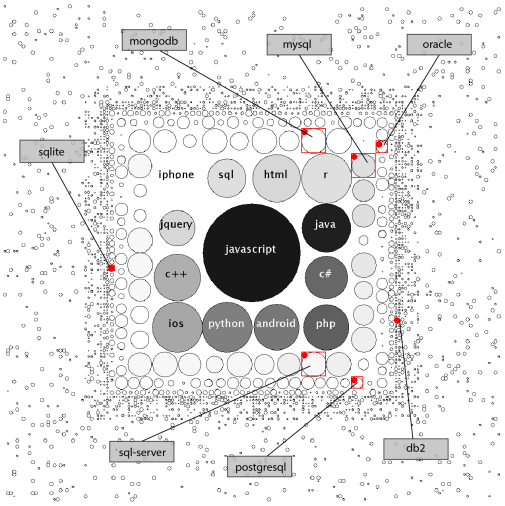

Figure 4.7. Positions of database related tags.

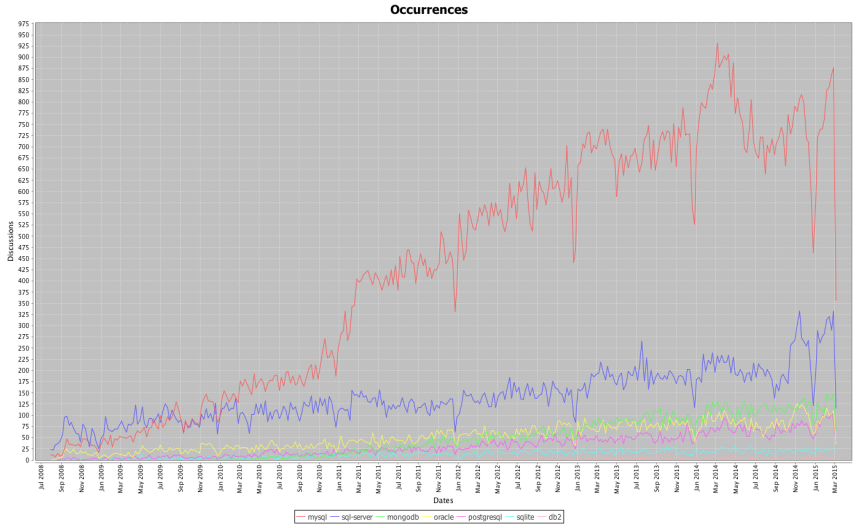

Figure 4.8. Line Chart for database platforms trends.

Figure 4.8 depicts a line chart that represents the number of discussions, per week, tagged with `mysql`, `sql-server`, `mongodb`, `oracle`, `postgresql`, `sqlite`, or `db2`. The plot shows that `sql-server` and `mysql` are initially the most popular database platform in Stack Overflow. Then around September 2009 `mysql` overtakes `sql-server` and starts to increase. In particular around November 2010 `mysql` grows quickly, this could be due to the release of "MySQL 5.5", and after that it clearly becomes the dominant database platform in Stack Overflow. Instead `sql-server` stays constant. The popularity of `mysql` can be traced to the fact that it is free, it is adopted by a large number of project, and its syntax is easy to learn respect to other alternative. Instead the less diffusion of `sql-server` can be linked to the fact that it is a commercial product.

Looking at the other database platforms, `oracle` grows slowly and it does not reach high levels of popularity. In 2009 `mongodb` is released as an open source project, in fact we can see that it starts growing from September 2009. Its growth rate is similar to `oracle`, however around February 2013 `mongodb` overtakes `oracle` and now it is the third database platform in Stack Overflow. `postgresql` grows with a slow rate and it never overtakes `oracle` or `mongodb`. `sqlite` and `db2` share a low constant trend, however `db2` is almost absent.

Figure 4.9 shows the same information provided by the line chart using stacked chart. We can notice that the dominant database platform is indeed `mysql`, followed by `sql-server`. `mongodb` and `oracle` have more or less the same amount of discussions, while `postgresql` and `sqlite` have less discussions but are still visible in the chart. Instead `db2` is almost invisible in the chart. The reason because some database platforms are less popular respect to `mysql`, could be due to the fact that they are commercial product, like `oracle` and `db2`, or although their are free, `mysql` is the preferred choice over them.

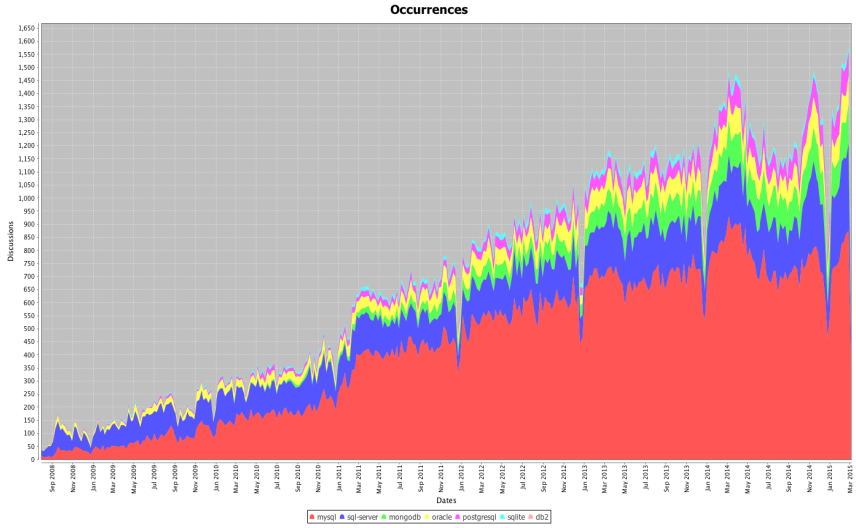

Figure 4.9. Stacked for database platforms trends.

4.4 Javascript Web Frameworks

The fourth story that we analyze is about javascript web framework tags that co-occur with the `javascript` tag. We compared four of them: `angular.js`, `backbone.js`, `ember.js`, `knockout.js`. These tags are the first four tags related to web frameworks in Figure 4.10 , which shows them in the last week of Stack Overflow. We can see that the most important framework is `angular.js`, because it is close to the center. The other three frameworks are discreetly popular.

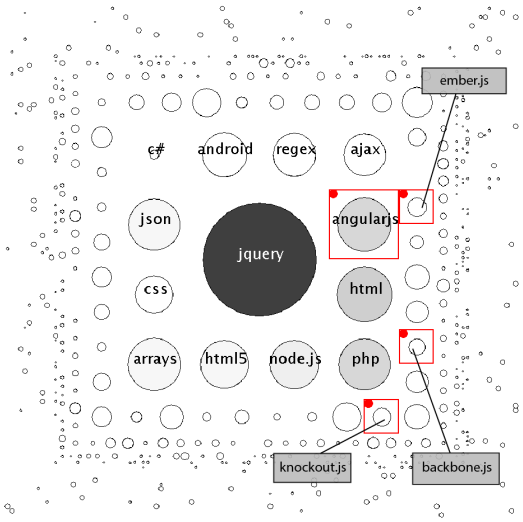

Figure 4.10. Positions of the tags related to javascript in its world.

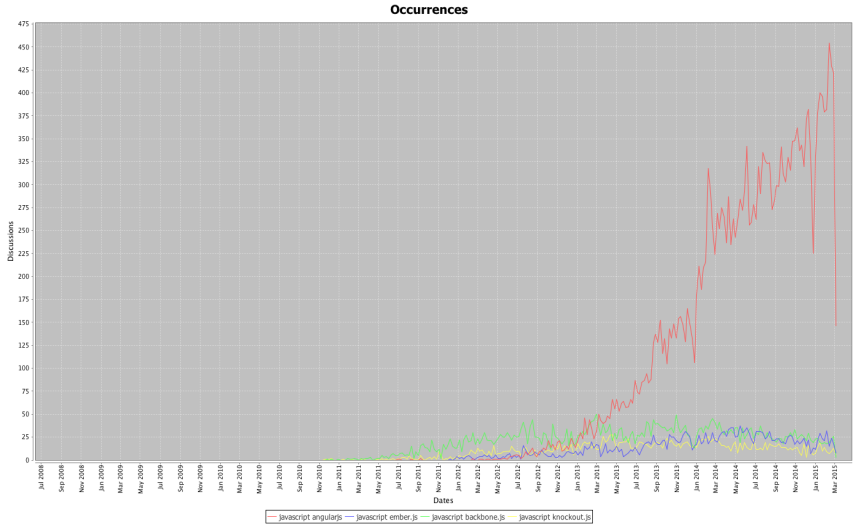

Figure 4.11. Line Chart for javascript frameworks trends.

Figure 4.11 depicts a line chart that represents the number of discussions, per week, tagged with `angular.js`, `ember.js`, `backbone.js`, or `knockout.js`. The chart reveals that `angular.js` is indeed the most popular framework that co-occur with `javascript`. `angular.js` starts growing quickly around June 2012, this date corresponds to the release of version 1.0.0 of the framework. At the end of 2013, the version 1.2.0 of `angular.js` is released and we can see a substantial growth around that date in Figure 4.11.

The other three frameworks have definitely another growth rate. `ember.js` have been released in December 2011, and we can see from the chart that it starts growing slowly around that date. `backbone.js` has been released in October 2010, but it is only at November 2010 that it starts rising in Stack Overflow, maybe this could be due to an initial period in beta. However after two year `backbone.js` stabilized and does not grow anymore. `knockout.js` was released in July 2010, but similarly to `backbone.js`, it emerges in Stack Overflow around November 2010. `knockout.js` grows slowly for two years, the it stabilizes.

Figure 4.12 shows the same information provided by the line chart using stacked chart. We can see that `backbone.js` and `knockout.js` are the first to appear in Stack Overflow, and the former seems to have more discussions. After them it comes `ember.js`, but its popularity is low. However the explosion of `angular.js` eclipses the other three frameworks. The popularity of `angularjs` could be due to the fact that is developed and supported by Google[1], but could also be a symptom of a hard learning curve of the framework.

[1] https://www.google.com

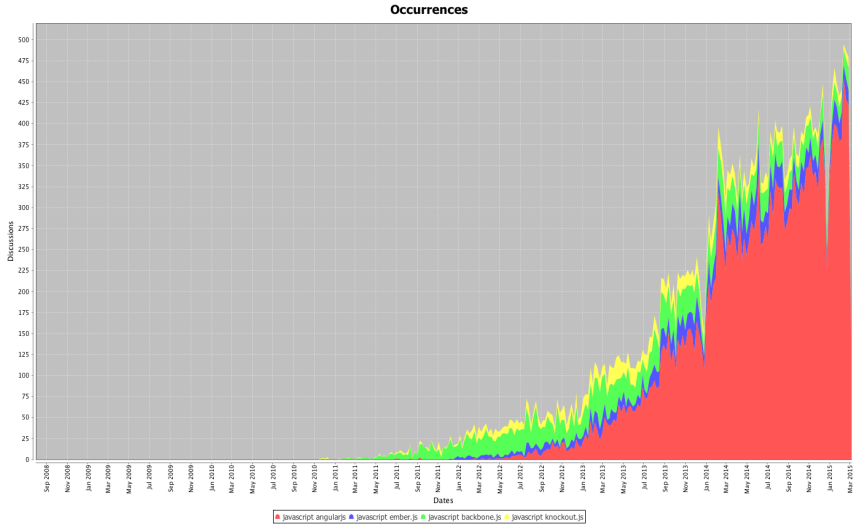

Figure 4.12. Stacked Chart for javascript frameworks trends.

4.5 Programming Languages

The fifth story that we report shows the programming languages trends in Stack Overflow. Figure 4.13 depicts the programming languages tags in the last week of Stack Overflow, and we can observe that most of the popular programming languages tags are located close to the center in the main visualization of SODA.

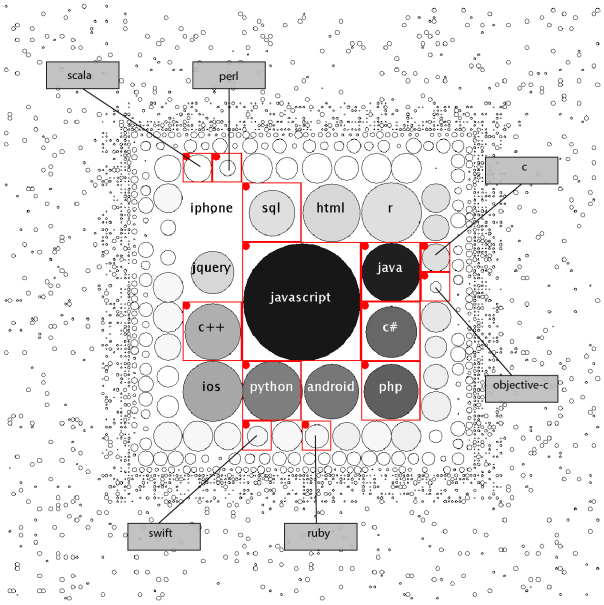

Figure 4.13. Programming Languages trends.

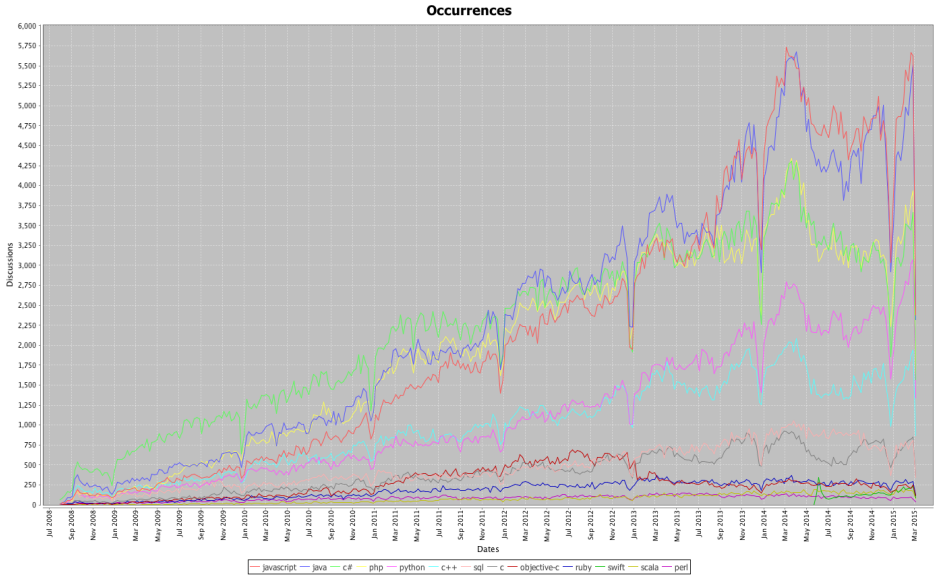

Figure 4.14. Line Chart of programming languages trends.

Figure 4.11 depicts a line chart that represents the number of discussions, per week, tagged with `javascript`, `java`, `c#`, `php`, `python`, `c++`, `sql`, `c`, `objective-c`, `ruby`, `swift`, `scala`, or `perl`. The chart reveals that most of the tags are constantly growing during the whole history of Stack Overflow, which is a simple consequence of the increasing importance of this online resource. The popularity rank between languages is also pretty much stable, with a couple of exceptions.

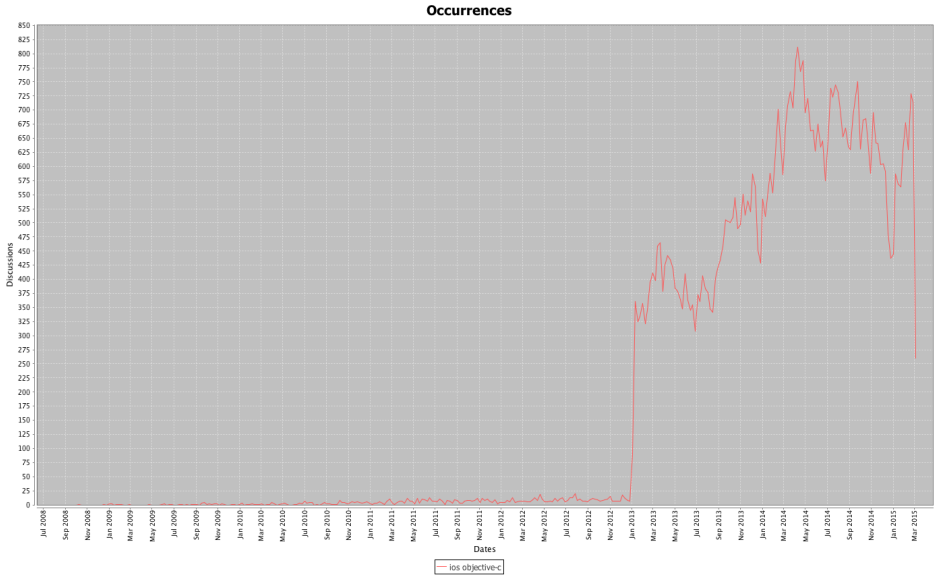

Figure 4.15. Line chart of discussions tagged with `ios` and `objective-c`.

Around May 2012, `python` starts to become more popular than `c++`, a fact that is more evident after 2014. `c#` is initially the most popular language in Stack Overflow, but it starts a relative decline in favor of `java` and `javascript`, which nowadays are the most popular discussed languages. `php` initially follows the same rate of growth as `java`, except that it starts to decrease at the same time with `c#`. Around January 2013, `objective-c` has a relative drop in popularity, this is due to the fact that after January 2013, discussions related to `objective-c` are also tagged with `ios`. Figure 4.15 shows how the co-occurrence of tags composed by `ios` and `objective-c` starts rising in January 2013. Recently, `javascript` is becoming extremely popular, even more than `java`. Another interesting fact that is pretty much evident is the birth of `swift`, with a considerable peak during the first months after its introduction and a slow increase of popularity.

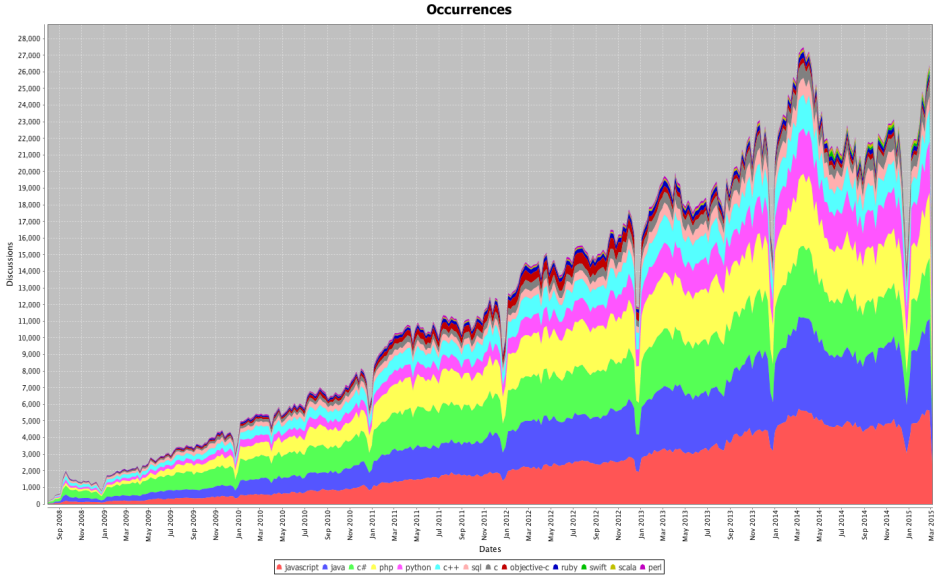

Figure 4.16. Stacked Chart of programming languages trends.

Figure 4.12 shows the same information provided by the line chart using stacked chart. We can observe that the amount of discussions tagged with `javascript`, `java`, `c#`, or `php` is pretty much the same until December 2013, after which the first two tags augment their number of discussions. `python` and `c++`, have more or less the same amount of discussions while the other languages have clearly less discussions.

4.6 Version Control Software

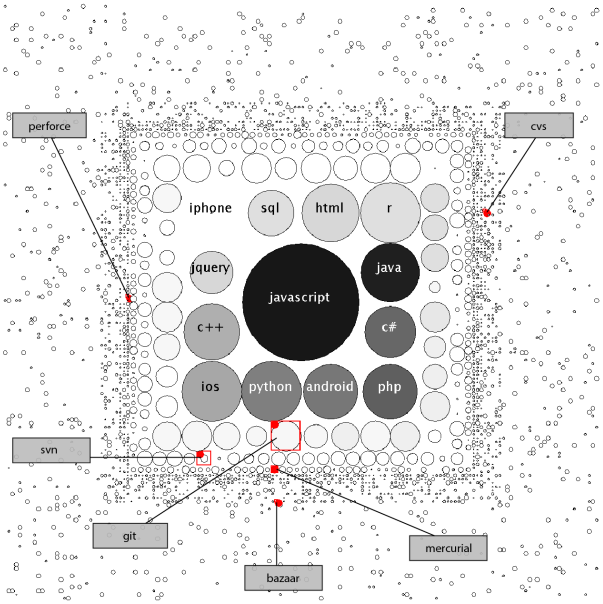

Figure 4.17. Version Control Software Trends.

The sixth story that we present is about Version Control Software (VCS). These systems are very important during the development process of a project, so it is natural to see how their trends evolve in a community of programmers like Stack Overflow. In this analysis we considered six versioning software: `git`, `svn`, `mercurial`, `perforce`, `csv`, `bazaar`. Figure 4.17 depicts their position in the last week of the Stack Overflow dump.

Figure 4.18 depicts a line chart that represents the number of discussions, per week, tagged with `git`, `svn`, `mercurial`, `perforce`, `csv`, or `bazaar`. We can see that `git` is one of the main standard used in the community of Stack Overflow, with a linear growth over the entire life cycle of Stack Overflow. `svn` is the second most used standard, but as we can see from the charts, it is not as popular as `git` and its trend is slowly decreasing. This is probably due to the fact that `git` provides services that `svn` does not have, like Github[2], a web repository hosting system based on `git`. `mercurial` was released in 2005, like `git`, but its trend is completely different, after a small peak in April 2010, it starts to decrease. The remaining system are not very popular in Stack Overflow, `perforce` is a commercial versioning system, used by companies like NVIDIA[3] and Ubisoft[4], the it is natural that we do not find a lot of discussion about it in Stack Overflow. `bazaar` is a versioning control software sponsored by Canonical[5], that aims to be more simple and efficient respect to `git`, but this is not reflected in Stack Overflow. Finally `csv` is an old standard of versioning software that is almost abandoned, in fact its trend is quite low.

[2] https://github.com
[3] http://www.nvidia.com
[4] https://www.ubisoft.com
[5] http://www.canonical.com

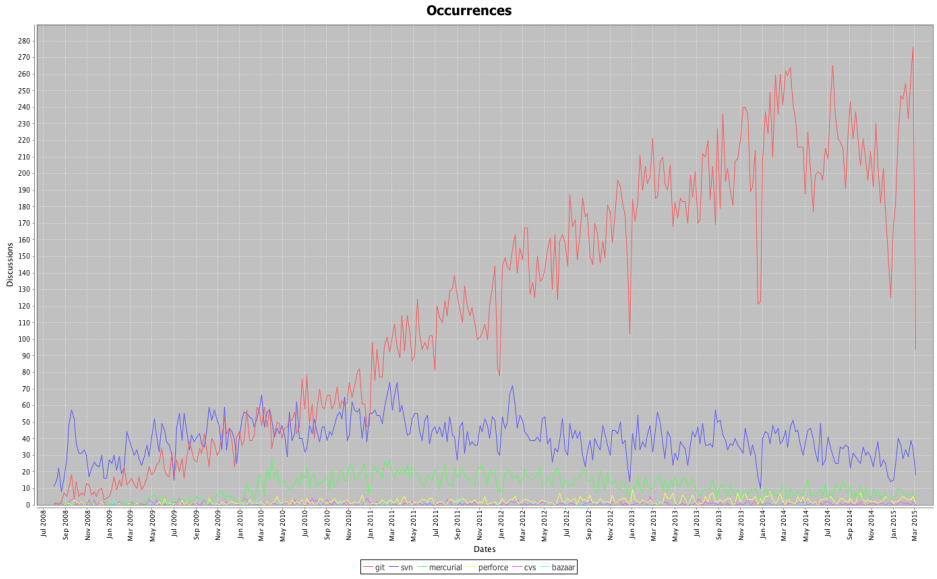

Figure 4.18. Line Chart of version control software.

Figure 4.19 shows the same information provided by the line chart using stacked chart. We can notice that the amount of discussions related to `git` is enormous compared to the other. `svn` has a discrete amount of discussions, but it is losing popularity. `mercurial` has a small number of discussions compared to the first two, and this number is quite low in the end. `perforce` and `bazaar` are almost invisible.

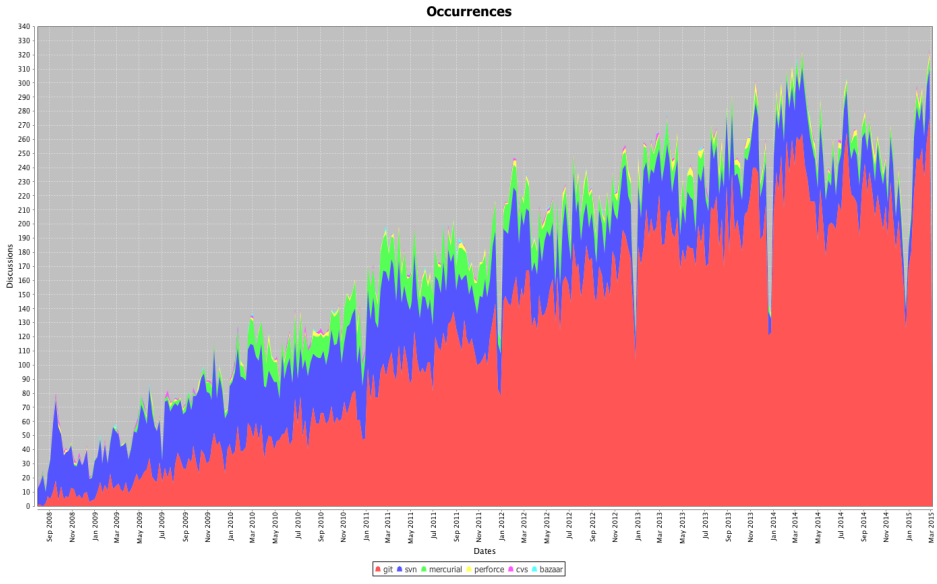

Figure 4.19. Stacked Chart of version control software.

Chapter 5

Conclusion

This chapter concludes this thesis by reviewing the aspects that has been discussed so far. We first give a review of our approach and then we present the future work.

5.1 Summary

Mining the Stack Overflow dataset is not a trivial task, it is a heavy process that requires the use of several tools and time. The main goal of our work is to provide a straightforward process to analyze and visualize the Stack Overflow dataset, in order to get insights about topics popularity and trends. We now summarize our three contributions:

- **A meaningful and effective visualization of the Stack Overflow dataset based on the available tagging system.** In Chapter 3 we presented a square packing strategy to effectively visualize the large amount of tags present in Stack Overflow. The strategy locates the most important tag at the center of the screen and the less popular tags around it. To maintain a reasonable view, we decided to keep the size of the square packing structure fixed and highlights the differences between tags by means of an internal circle, which can vary in size and color. In this way users are able to immediately identify popular tags and spot the differences about their importance.

- **A tool that helps researchers to investigate Stack Overflow in a reasonable time.** In Chapter 3 we presented SODA, a tool to visualize the evolution of the trends in the Stack Overflow dataset based on tag. SODA provides the possibility to analyze single or co-occurrent tags in different time intervals and frames. This tool can be used by researches to analyze the popularity and trends in the dataset in a reasonable time, avoiding to compute them directly every time.

- **A collections of evidence that validate our approach.** In Chapter 4 we used SODA to explore the discussion tags in the history of Stack Overflow, and we reported a collection of interesting stories related to development topics that we found out by means of SODA.

5.2 Future Work

Currently SODA relies on the assumption that discussion tags can be considered as topics of a discussion. This is true with a certain level of approximation. However, in the future we plan to reconsider statistical topic modeling methods like LDA to model discussion topics in a more precise way. There are a number of possible improvements and extensions that can be done to SODA:

- **Different visualizations**. In order to extrapolate different aspect of the data. Right now SODA uses the data of Stack Overflow questions, in the future we plan to integrate information related to answers, comments, and users. For example it would be interesting to show users contributions in topics, to visualize demographics informations related to developers, and to provide regional information about programmers. Another interesting visualization would be to depict the evolution of the trends in Stack Overflow based on comments or answers, or both. We could also provide a visualization to show what the most commented or answered topics are. An additional visualization could depict discussion contents like the presence of different structured fragments like code.
- **Filters.** We plan to consider potential filters in SODA, because they could allow users to include or exclude a particular class of questions from the analysis. For example with a filter we could select or remove questions without answers, questions with one answer, questions that have a number of answers under or above a particular threshold, or questions that have been closed. If we analyze users, filters can be used to highlight only some regions of the world which contribute to Stack Overflow, or they can be used to study users in a particular range of age.
- **Web Interface**. We have the intention to build a web interface based on SODA, for example using the D3 javascript library. In this way users do not have to download SODA, but they could simply work with the web interface to investigate the Stack Overflow dataset.

Bibliography

[1] P. Achananuparp, I. Lubis, Y. Tian, D. Lo, and E.-P. Lim. Observatory of trends in software related microblogs. In *Proceedings of ASE 2012 (27th IEEE/ACM International Conference on Automated Software Engineering)*, pages 334–337, 2012.

[2] L. A. Adamic, J. Zhang, E. Bakshy, and M. S. Ackerman. Knowledge sharing and yahoo answers: everyone knows something. In *Proceedings of the 17th international conference on World Wide Web*, pages 665–674. ACM, 2008.

[3] L. Akoglu, D. H. Chau, U. Kang, D. Koutra, and C. Faloutsos. Opavion: Mining and visualization in large graphs. In *Proceedings of the 2012 ACM SIGMOD International Conference on Management of Data*, pages 717–720. ACM, 2012.

[4] M. Allamanis and C. Sutton. Why, when, and what: analyzing stack overflow questions by topic, type, and code. In *Proceedings of the 10th Working Conference on Mining Software Repositories*, pages 53–56. IEEE Press, 2013.

[5] M. Asaduzzaman, A. S. Mashiyat, C. K. Roy, and K. A. Schneider. Answering questions about unanswered questions of stack overflow. In *Proceedings of the 10th Working Conference on Mining Software Repositories*, pages 97–100. IEEE Press, 2013.

[6] K. Bajaj, K. Pattabiraman, and A. Mesbah. Mining questions asked by web developers. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, pages 112–121. ACM, 2014.

[7] A. Barua, S. W. Thomas, and A. E. Hassan. {What are developers talking about? An analysis of topics and trends in Stack Overflow}. *Empirical Software Engineering*, 19:619–654, 2012.

[8] B. Bazelli, A. Hindle, and E. Stroulia. On the personality traits of stackoverflow users. In *Software Maintenance (ICSM), 2013 29th IEEE International Conference on*, pages 460–463. IEEE, 2013.

[9] A. Begel, R. DeLine, and T. Zimmermann. Social media for software engineering. In *Proceedings of FOSER 2010 (FSE/SDP Workshop on Future of Software Engineering Research)*, pages 33–38, 2010.

[10] T. Bissyande, F. Thung, D. Lo, L. Jiang, and L. Reveillere. Popularity, interoperability, and impact of programming languages in 100,000 open source projects. In *Proceedings of COMPSAC 2013 (37th IEEE Annual Computer Software and Applications Conference)*, pages 303–312, 2013.

[11] D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. *the Journal of machine Learning research*, 3:993–1022, 2003.

[12] I. Borg and P. J. Groenen. *Modern multidimensional scaling: Theory and applications*. Springer Science & Business Media, 2005.

[13] A. Bosu, C. S. Corley, D. Heaton, D. Chatterji, J. C. Carver, and N. A. Kraft. Building reputation in stackoverflow: an empirical investigation. In *Proceedings of the 10th Working Conference on Mining Software Repositories*, pages 89–92. IEEE Press, 2013.

[14] J. C. Campbell, C. Zhang, Z. Xu, A. Hindle, and J. Miller. Deficient documentation detection: A methodology to locate deficient project documentation using topic analysis. In *Proceedings of the 10th International Working Conference on Mining Software Repositories*, pages 57–60. IEEE, 2013.

[15] S. Chang and A. Pal. Routing questions for collaborative answering in community question answering. In *Advances in Social Networks Analysis and Mining (ASONAM), 2013 IEEE/ACM International Conference on*, pages 494–501. IEEE, 2013.

[16] J. Cordeiro, B. Antunes, and P. Gomes. Context-based recommendation to support problem solving in software development. In *Recommendation Systems for Software Engineering (RSSE), 2012 Third International Workshop on*, pages 85–89. IEEE, 2012.

[17] D. Correa and A. Sureka. Fit or unfit: Analysis and prediction of 'closed questions' on Stack Overflow. In *Proceedings of the first ACM Conference on Online Social Networks*, pages 201–212. ACM, 2013.

[18] D. Correa and A. Sureka. Chaff from the wheat: Characterization and modeling of deleted questions on stack overflow. In *Proceedings of the 23rd international conference on World wide web*, pages 631–642. ACM, 2014.

[19] X. Z. Fern and C. E. Brodley. Random projection for high dimensional data clustering: A cluster ensemble approach. In *ICML*, volume 3, pages 186–193, 2003.

[20] S. Grant and B. Betts. Encouraging user behaviour with achievements: an empirical study. In *Mining Software Repositories (MSR), 2013 10th IEEE Working Conference on*, pages 65–68. IEEE, 2013.

[21] S. Huron, R. Vuillemot, and J.-D. Fekete. Visual sedimentation. *Visualization and Computer Graphics, IEEE Transactions on*, 19(12):2446–2455, 2013.

[22] D. Kavaler, D. Posnett, C. Gibler, H. Chen, P. Devanbu, and V. Filkov. Using and asking: Apis used in the android market and asked about in stackoverflow. In *Social Informatics*, pages 405–418. Springer, 2013.

[23] A. Kuhn, D. Erni, P. Loretan, and O. Nierstrasz. Software cartography: Thematic software visualization with consistent layout. *Journal of Software Maintenance and Evolution: Research and Practice*, 22(3):191–210, 2010.

[24] M. Linares-Vásquez, B. Dit, and D. Poshyvanyk. An exploratory analysis of mobile development issues using stack overflow. In *Proceedings of the 10th Working Conference on Mining Software Repositories*, pages 93–96. IEEE Press, 2013.

[25] L. Mamykina, B. Manoim, M. Mittal, G. Hripcsak, and B. Hartmann. Design lessons from the fastest q&a site in the west. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 2857–2866. ACM, 2011.

[26] P. Morrison and E. Murphy-Hill. Is programming knowledge related to age? an exploration of stack overflow. In *Mining Software Repositories (MSR), 2013 10th IEEE Working Conference on*, pages 69–72. IEEE, 2013.

[27] S. M. Nasehi, J. Sillito, F. Maurer, and C. Burns. What makes a good code example?: A study of programming q&a in stackoverflow. In *Software Maintenance (ICSM), 2012 28th IEEE International Conference on*, pages 25–34. IEEE, 2012.

[28] C. Parnin and C. Treude. Measuring api documentation on the web. In *Proceedings of the 2nd international workshop on Web 2.0 for software engineering*, pages 25–30. ACM, 2011.

[29] C. Parnin, C. Treude, and L. Grammel. Crowd documentation: Exploring the coverage and the dynamics of api discussions on stack overflow,âĂİ georgia institute of technology. 2012.

[30] L. Ponzanelli, G. Bavota, M. Di Penta, R. Oliveto, and M. Lanza. Mining stackoverflow to turn the ide into a self-confident programming prompter. In *Proceedings of the 11th Working Conference on Mining Software Repositories*, pages 102–111. ACM, 2014.

[31] L. Ponzanelli, A. Mocci, A. Bacchelli, and M. Lanza. Understanding and classifying the quality of technical forum questions. In *Quality Software (QSIC), 2014 14th International Conference on*, pages 343–352. IEEE, 2014.

[32] L. Ponzanelli, A. Mocci, A. Bacchelli, M. Lanza, and D. Fullerton. Improving low quality stack overflow post detection. In *Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on*, pages 541–544. IEEE, 2014.

[33] M. M. Rahman, S. Yeasmin, and C. K. Roy. An ide-based context-aware meta search engine. In *Reverse Engineering (WCRE), 2013 20th Working Conference on*, pages 467–471. IEEE, 2013.

[34] F. Riahi, Z. Zolaktaf, M. Shafiei, and E. Milios. Finding expert users in community question answering. In *Proceedings of the 21st international conference companion on World Wide Web*, pages 791–798. ACM, 2012.

[35] A. K. Saha, R. K. Saha, and K. A. Schneider. A discriminative model approach for suggesting tags automatically for stack overflow questions. In *Proceedings of the 10th Working Conference on Mining Software Repositories*, pages 73–76. IEEE Press, 2013.

[36] D. Schenk and M. Lungu. Geo-locating the knowledge transfer in StackOverflow. In *Proceedings of the 2013 International Workshop on Social Software Engineering*, pages 21–24. ACM, 2013.

[37] C. Stanley and M. D. Byrne. Predicting tags for stackoverflow posts. In *Proceedings of ICCM*, volume 2013, 2013.

[38] M.-A. Storey, C. Treude, A. van Deursen, and L.-T. Cheng. The impact of social media on software engineering practices and tools. In *Proceedings of the FSE/SDP workshop on Future of software engineering research*, pages 359–364. ACM, 2010.

[39] S. Subramanian and R. Holmes. Making sense of online code snippets. In *Proceedings of the 10th Working Conference on Mining Software Repositories*, pages 85–88. IEEE Press, 2013.

[40] S. Subramanian, L. Inozemtseva, and R. Holmes. Live api documentation. In *Proceedings of the 36th International Conference on Software Engineering*, pages 643–652. ACM, 2014.

[41] C. Treude, O. Barzilay, and M.-A. Storey. How do programmers ask and answer questions on the web?: Nier track. In *Software Engineering (ICSE), 2011 33rd International Conference on*, pages 804–807. IEEE, 2011.

[42] C. Treude and M.-A. Storey. How tagging helps bridge the gap between social and technical aspects in software development. In *Proceedings of ICSE 2009 (31st ACM/IEEE International Conference on Software Engineering)*, pages 12–22, 2009.

[43] S. Wang, D. Lo, and L. Jiang. An empirical study on developer interactions in StackOverflow. In *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, pages 1019–1024. ACM, 2013.

[44] S. Wang, D. Lo, B. Vasilescu, and A. Serebrenik. Entagrec: an enhanced tag recommendation system for software information sites. In *Software Maintenance and Evolution (ICSME), 2014 IEEE International Conference on*, pages 291–300. IEEE, 2014.

[45] C. Ware. *Information visualization: perception for design*. Elsevier, 2012.

[46] X. Xia, D. Lo, X. Wang, and B. Zhou. Tag recommendation in software information sites. In *Proceedings of MSR 2013 (10th Working Conference on Mining Software Repositories)*, pages 287–296, 2013.