Global Journal of Information Technology

Volume 06, Issue 1, (2016) 107-116

http://sproc.org/ojs/index.php/gjit

Big data in software engineering: A systematic literature review

Selami Bagriyanik *, Turkcell Technology R&D, Maltepe, Istanbul, Turkey
Adem Karahoca, Software Engineering Department, Besiktas, Istanbul, Turkey

Suggested Citation:
Bagriyanik, S. & Karahoca, A. (2016). Big data in software engineering: A systematic literature review. *Global Journal of Information Technology*. *6*(1), 107-116

Received 08 January, 2016; revised 04 February, 2016; accepted 17 March, 2015.
Selection and peer review under responsibility of Prof. Dr. Adem Karahoca, Bahcesehir University, Turkey

Abstract

Purpose of Study: We investigate the big data studies using batch and/or streaming data generated in the process of software development lifecycle. All phases of application development phases are in our scope including but not limited to elicitation, requirements analysis, design, software implementation, version control management, unit / functional / regression / automated / performance / stress test, release management, application log monitoring, application usage monitoring, user complaint management, security and compliance management and software problem management.

Methods: We use a systematic literature review methodology used in Software Engineering studies to find and analyse the related studies published from January 2010 to October 2015. We synthesize the quantitative and qualitative outputs of selected papers and report the results.

Findings and Results: In general, there are scarce studies in the literature. However there are relatively more papers regarding some areas such as Software Quality, Development, Project Management and Human Computer Interaction. However research in some fields such as Deployment, Requirements Engineering, Release Management and Mobile Applications were relatively less.

Conclusions & Recommendations: More studies are required to identify the use cases, data attributes, measurements, platform requirements especially in the fields which are identified as having lack of study. A holistic big data perspective is needed to support software engineering ecosystems in large and complex enterprises.

Keywords: Big Data, Software Engineering, Software Analytics, Data Mining, Software Development, Operational Intelligence, Software Archaeology

*ADDRESS FOR CORRESPONDENCE: **Selami Bagriyanik**, Turkcell Technology R&D, Maltepe, Istanbul, Turkey.
E-mail address: selami.bagriyanik@turkcell.com.tr

1. Introduction

Knowledge discovery from big data seems to have a huge potential for businesses, scientific studies, governments and so on. It presents lots of new opportunities and new research avenues [1]. Big Data also enables synergistic inter- disciplinary studies [2]. The 5Vs (Volume, Velocity, Variety, Veracity and Value) of big data [3] have become valid for the data generated within software engineering ecosystem or in the process of software development life cycle from elicitation to deployment and monitoring in the field until phasing out of the software. For example software source code is a basic artefact in the software engineering domain and Google declared that it had 2 billion lines of code [4]. This huge quantity gives an idea about how much software related big data a large enterprise sit over. Another example is the artefacts, changes and process data produced during the life cycle of software projects conducted in a large scale enterprise. In a recent case study, the number of total yearly finished projects (Small and mid-size) in a large telecommunications company is given as 5350 [5]. Considering some basic use cases that generates data in the context of the projects, for instance people assignments or timesheet data may yield an order of magnitude increase in the number. Code changes may introduce an increase of two orders of magnitude. Logs, transactions, usage or incidents data generated brings us to three, four or more orders of magnitude of project numbers. Therefore, software engineering practitioners have already entered the era of big data. We observe this phenomenon in the organisation of leading technology companies such as Microsoft and Google as well. Microsoft has a research team conducting empirical software engineering research and Google employs at least 100 engineers in developing its tools using data mining technics [6].

In this study, we conducted a systematic literature review (SLR) covering the intersection of Big data and concepts around software engineering discipline. The rest of the paper is organized as follows: research method details are given in the second section. The third section discusses the results obtained from extracted data and is followed by the conclusions.

2. Research Method

We conducted the SLR following Kitchenham and Charters' de facto review guideline for software engineering [7]. This methodology has been used in more than one thousand six hundred studies (Google Scholar's citation count) in last eight years. The original idea for employing systematic literature review practice is coming from evidence-based medicine. Kitchenham and Charters customised the method for Software Engineering domain [7]. There are three main stages of the method: planning, conducting and documenting the review. The review steps are as follows: definition of the research questions, design of the search, conducting the search, selecting the studies, assessing the quality and synthesizing the data at hand.

2.1. Research Questions

In this SLR we intend to find answers for the following questions:

Research Question 1: In which software engineering areas Big Data and Software Engineering are interacting and to what extent? By Big Data we mean related keywords such as data mining, analytics, streaming data, complex event processing, knowledge discovery, operational intelligence etc. This researh question aims to find the areas (requirements engineering, performance testing etc.) that benefits from the Big Data research. State of practice for the Software Engineering practitioner community and research opportunities for researchers will also be identified.

Research Question 2: Which software engineering artefacts are used for Big Data processing? What are the most frequently used artefacts? We want to discover the types of data used in Software Engineering Big Data research and whether there is a lack of holistic data usage or not.

2.2. Search Strategy

Having defined the research questions in previous section, we designed a search string based on the questions. To cover all relevant studies, keywords and terms regarding Software Engineering and Big Data are consolidated to define the search string. Alternative terms are connected using OR Boolean operator to get a wide coverage. There are mainly two segments of the search statement. The first sub-segment is the union of all basic Big Data related terms, second sub-segment is the union of Software Engineering keywords. The intersection of the first and second sub-segments constitutes the first output for Big Data in Software Engineering research. The second segment addresses interdisciplinary terms. Consequently, the union of these two segments are applied an OR Boolean operator to get the union of the results. As a result, we generated the following search string:

[("Data Mining" **OR** "Big Data" OR "Streaming data" **OR** "complex event processing" **OR** "CEP" **OR** "Statistical Methods" **OR** "Anomaly Detection" **OR** "Knowledge Discovery") **AND (**"Software Engineering" **OR** "SE" **OR** "SD" **OR** "Software Development" **OR** "Software Implementation" **OR** "SDLC" **OR** "Software Development Life Cycle" **OR** "Requirements Engineering" **OR** "Software Design" **OR** "Software Architecture" **OR** "DevOps" **OR** "Continuous Delivery" **OR** "Continuous Integration" **OR** "Project Management" **OR** "Application Monitoring" **OR** "Software Measurement" **OR** "Software Size" **OR** "Software Metric" **OR** "Release Management" **OR** "Change Management" **OR** "Version Control" **OR** "Usability" **OR** "Software Usage" **OR** "Appplication Usage Monitoring" **OR** "HCI" **OR** "Human Computer Interaction" **OR** "Software Testing" **OR** "Test Automation" **OR** "Automated Test" **OR** "Unit Test" **OR** "Performance Test" **OR** "Stress Test" **OR** "Software Quality" **OR** "Incident Management" **OR** "Complaint Management" **OR** "Software Defect Prediction" **OR** "Software Log Mining" **OR** "Software Fault Detection" **OR** "Software Security" **OR** "Software Fraud detection" **OR** "Transaction mining" **OR** "Software Integration" **OR** "Static Code Analysis" **OR** "Application Development Life Cycle Management" **OR** "ADLM"**)] OR (**"Operational Intelligence" OR "Operational Analytics" OR "Software Analytics" OR "Software Archaeology" OR "Digital Archaeology "**)**

2.3. Literature Resources

We used Google Scholar as the primary resource for three reasons. First, English published study coverage of Google Scholar is very high (87 %) [8]. Second, the subject of the study is interdisciplinary and Google Scholar is a convenient platform to find the related research under study. Third, there’s an important disadvantage of other electronic databases. The search strings needed to be adapted to suit the specific requirements of the different databases. This may be a very time consuming task for the researchers. Google Scholar has some important issues as well [9]. Google Scholar has a 256 character limitation for the search string. If the length of the search string is above 256, it silently truncates the string without warning [9]. To overcome this limitation we constituted 17 shorter subqueries from the original search string.

Our search covers the time frame from January 2010 to November 2015. We aimed to cover relevant papers in the recent past. We also added another filter on the content search. We conducted the search by using “allintitle” keyword to limit the keyword search within paper titles. In this manner we aimed to increase relevancy.

2.4. Study Selection Process

We obtained 326 studies by executing our 17 search strings. In the first filtration phase, we made a quick scan of the abstracts of all the resulting papers and made elimination based on the following inclusion and exclusion criteria:

Inclusion Criteria:

- Paper must contain big data studies in software engineering domain
- Studies reviewed in peer reviewed workshop OR conference OR journal OR are reported in a technical report OR Msc/Phd thesis

Exclusion Criteria:

- Studies not in English
- Study is a book chapter

After the first filtration, 112 papers remained for the second phase. In the second phase remaining 112 papers' full content were scanned and assessed according to the quality criterias given in the next section. 32 papers with highest quality assessment scores were selected. These papers are given in reference section in sequence [10–41].

2.5. Study Quality Assessment

We specified following quality assessment criteria in order to determine the final output of the survey which are the 32 papers cited in section 2.4. Each criteria is 5 points. Thus the possible maximum score is 20 and minimum score is 0.

- **Criteria 1**: Study contribution is clearly described.
- **Criteria 2**: Artefacts and methods used in the study are clearly described.
- **Criteria 3**: Empirical validation is performed.
- **Criteria 4**: The results and applications are described and discussed thorougly.

Each candidate paper was given a score using the assesment. The highest score was 17 and the lowest score was 5. All primary studies scored above 12 points were selected.

2.6. Data Extraction and Data Synthesis

To reach the data needed to answer our research questions and constitute some additional statistical data, we extracted following data from the papers: Title, Quality Criteria 1 Score, Quality Criteria 2 Score, Quality Criteria 3 Score, Quality Criteria 4 Score, Overall Quality Score, Year of Publication, Type, Country, SE Sub Domain, Artefacts, Objective, Data Processing Algorithms, Batch/Streaming, Tool/Technology. Next, extracted data is synthesized using graphics and tables which are presented in the following section.

3. Data Results

In this section, answers for the two research questions defined in section 2.1 will be discussed. Some other statistical data extracted from the papers remained after first filtration and second filtration (selected final primary studies) will be presented as well.

3.1. Research Question 1

The question was "**In which software engineering areas Big Data and Software Engineering are interacting and to what extent?"** To answer this question, we classified the papers based on the software engineering phases or keywords they focus in the data extraction phase. In Figure 2, the numbers of studies for each phase are shown. For the 112 papers remained after first filtration (Blue bars), the most popular six areas in which big data research is active are Software Quality, Development, Test, Project Management, Human Computer Interaction and Operational Intelligence. The picture for the selected primary studies (Red bars) is partially different. Software Quality, Development, Project Management and Human Computer Interaction are still in the top six list. However two new domains appear in the list: Software Evolution and Software Visualisation. Primary studies for Operational Intelligence and Test areas have a sharp decline. Deployment, Requirements Engineering, Release Management and Mobile Applications are the domains that have nearly no studies in both paper sets.

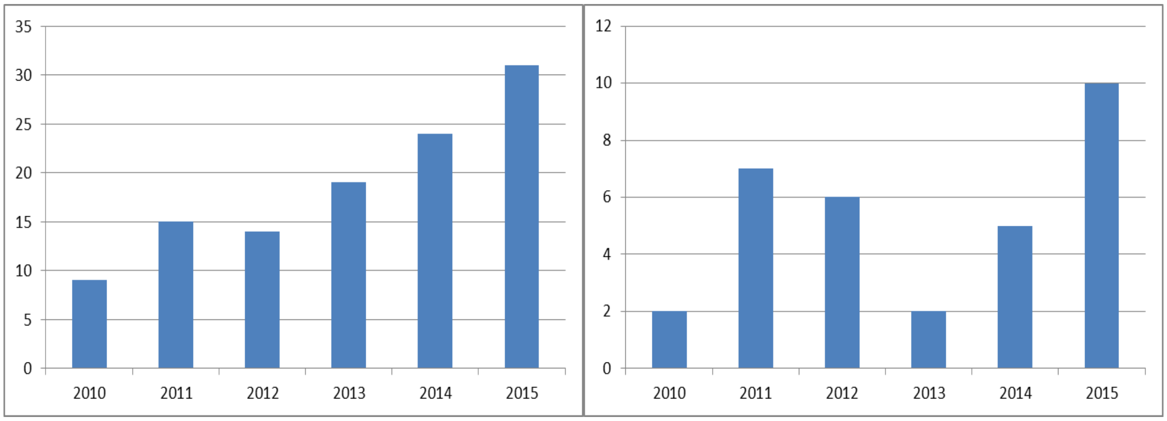

Figure 1 Distribution of Studies per Year for First (Left) and Second (Right) Filtration

3.2. Research Question 2

The question was "**Which software engineering artefacts are used for Big Data processing? What are the most frequently used artefacts?**" To answer this question, we also classified the papers based on the software engineering artefacts they use. In Figure 3, the numbers of studies for each artefact are shown. Source code and source code changes, bug related data and operational data are the most used artefacts in both papers set. The usage of all the other artefact types are not significant. Average Artefact number per paper is 1.16 in the paper set after first filtration. The average is 1.06 for the second paper set. This implies that majority of the papers focus on the problems using a single artefact. This finding is also consistent with the Figure 7. That is, majority of the papers lack a holistic perspective. More studies are required to correlate several software engineering artefacts to support high level decision making.

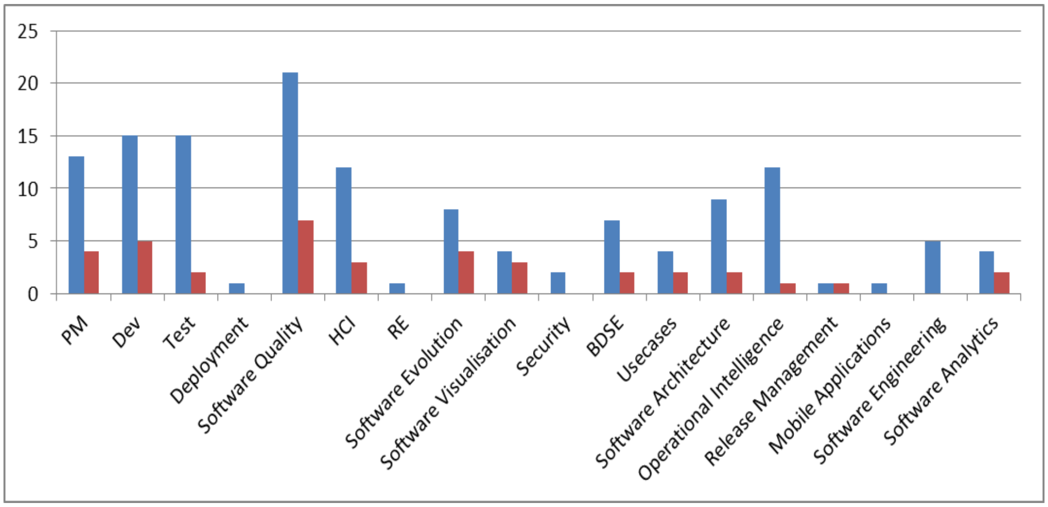

Figure 2 Distribution of Studies per Keywords for First (Blue) and Second (Red) Filtration

3.3. Additional Statistics

The trend of the number of the papers in the last six years is shown in Figure 1. For the first paper set, there is a regular increase beginning from 2012. However there is no regular increase pattern for the selected primary studies even though 2015 has the maximum number of studies. Figure 4 presents the paper type distribution. Conference and journal papers are the majority of the publications and conference papers are slightly more than journal studies. In Figure 5 and Figure 6, paper numbers by

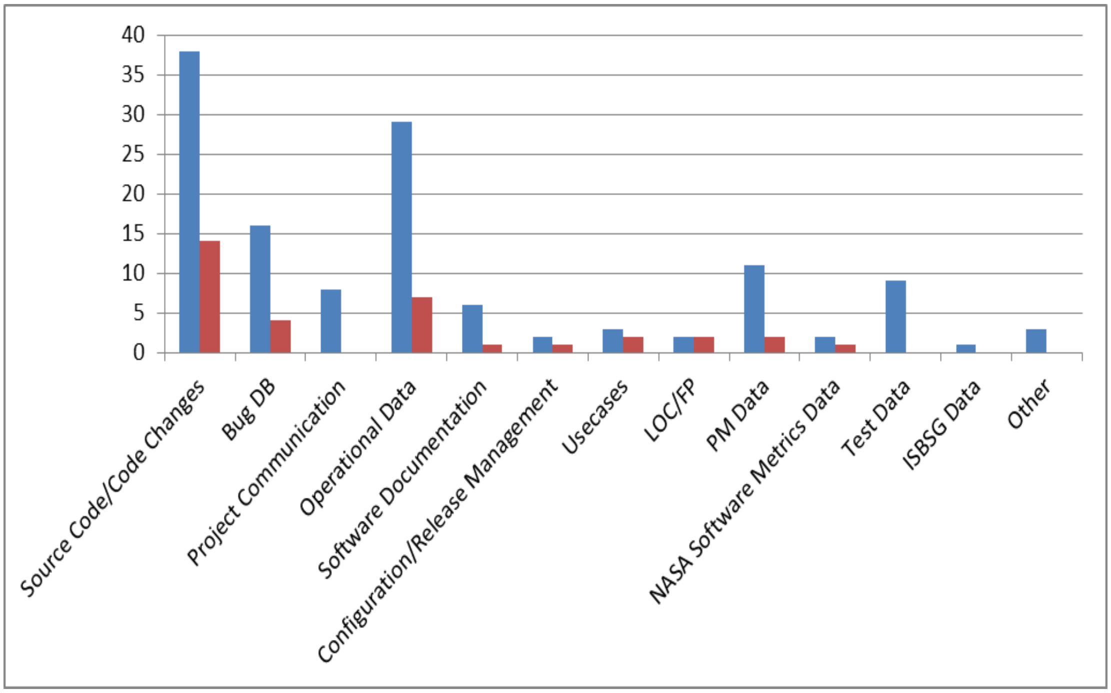

Figure 3 Distribution of Studies per Artefacts for First (Blue) and Second (Red) Filtration

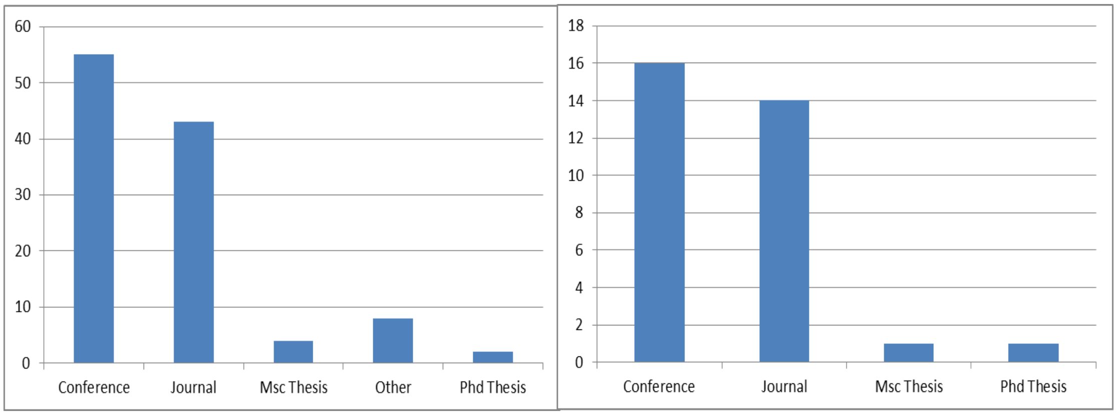

Figure 4 Distribution of Studies per Type for First (Left) and Second (Right) Filtration

Countries are given. USA, India and China seem to dominate the publications for the first set. However for the selected studies, USA is leading the way by itself.

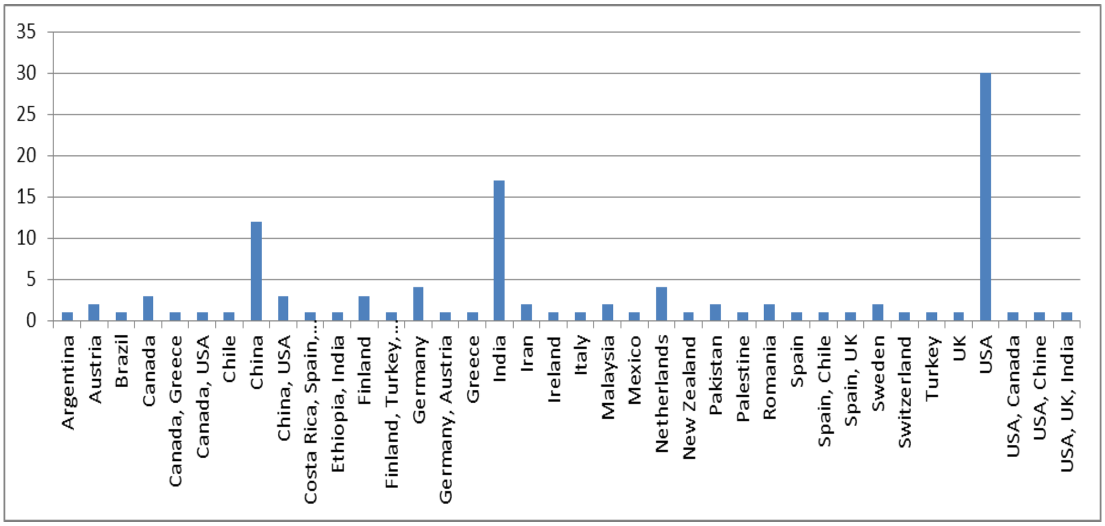

Figure 5 Distribution of Studies by Country for the First Filtration

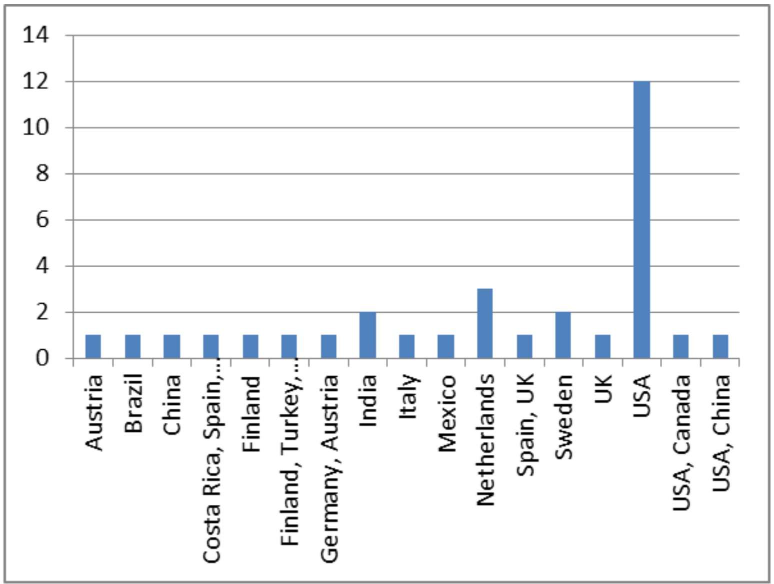

Figure 6 Distribution of Studies by Country for the 2nd Filtration

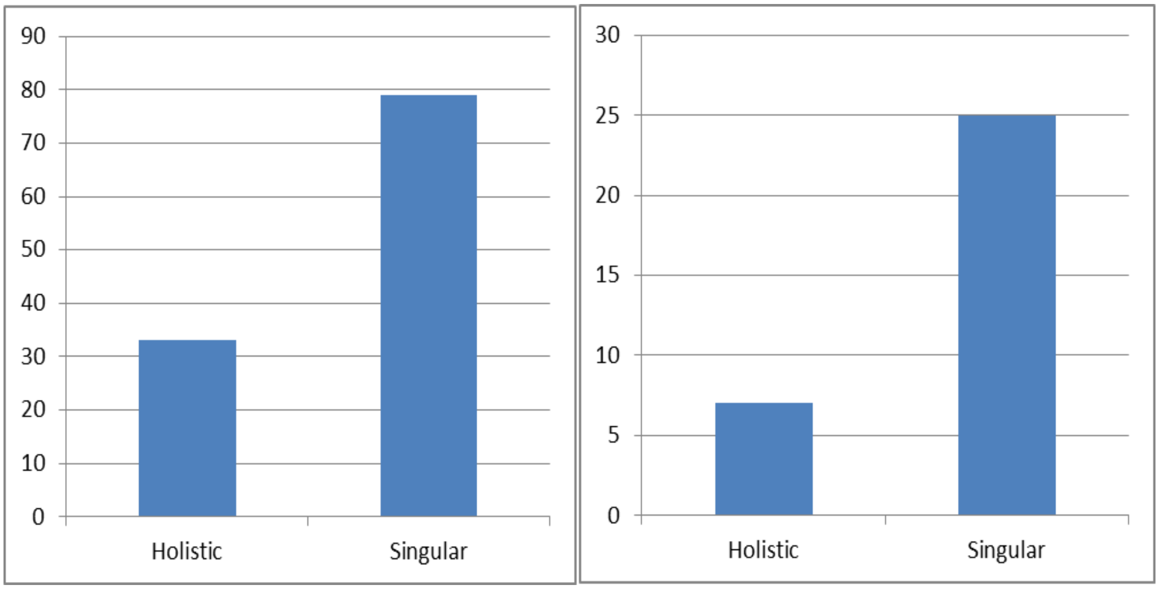

Figure 7 Distribution of Studies per Type for First (Left) and Second (Right) Filtration

4. Conclusions

In this study, we investigated the current state of the research in Big Data and Software Engineering by using the systematic literature review methodology. We selected the primary studies extracting 326 relevant studies published in last six years (2010-2015). In the first filtration, we eliminated about % 66 of the extracted studies using inclusion and exclusion criterias. 32 primary studies with highest quality assessment scores were selected out of 112 papers.

The conducted primary studies in the literature are scarce. However some areas are studied relatively more. Software Quality, Development, Project Management, Human Computer Interaction, Software Evolution and Software Visualisation are the most active research topics in software engineering big data studies. Source code and source code changes, bug related data and operational data are the most used artefacts in the studies. Deployment, Requirements Engineering, Release Management and Mobile Applications are the areas that have less studies. Studies lack a holistic perspective in terms of used artefacts. More studies are required to correlate several software engineering artefacts to support efficient decision making in large and complex enterprises.

References

[1] C.L. Philip Chen, C.-Y. Zhang, Data-intensive applications, challenges, techniques and technologies: A survey on Big Data, Inf. Sci. (Ny). 275 (2014) 314–347. doi:10.1016/j.ins.2014.01.015.

[2] M. Chen, S. Mao, Y. Liu, Big data: A survey, Mob. Networks Appl. 19 (2014) 171–209. doi:10.1007/s11036-013-0489-0.

[3] S. Yin, O. Kaynak, Big Data for Modern Industry : Challenges and Trends, Proc. IEEE. 103 (2015) 143–146. doi:10.1109/JPROC.2015.2388958.

[4] Wired.com, Google Is 2 Billion Lines of Code—And It's All in One Place, (2015). http://www.wired.com/2015/09/google-2-billion-lines-codeand-one-place/ (accessed December 12, 2015).

[5] M. Salmanoğlu, K. Öztürk, S. Bağrıyanık, E. Ungan, Benefits and Challenges of Measuring Software Size : Early Results in a Large Organization, in: IWSM Mensura, 2015.

[6] R. Robbes, R. Vidal, M.C. Bastarrica, Are Software Analytics Efforts Worthwhile for Small Companies ? The Case of Amisoft, IEEE Softw. SEPTEMBER/ (2013) 46–53.

[7] B. Kitchenham, S. Charters, Guidelines for performing Systematic Literature Reviews in Software Engineering, Tech. Rep. (2007).

[8] M. Khabsa, C.L. Giles, The Number of Scholarly Documents on the Public Web, (2014). doi:10.1371/journal.pone.0093949.

[9] Aa. Tay, 8 surprising things I learnt about Google Scholar, (2014). http://musingsaboutlibrarianship.blogspot.com.tr/2014/06/8-surprising-things-i-learnt-about.html#.VoeMGvmqqN3 (accessed November 30, 2015).

[10] R. Malhotra, A. Jain, Fault Prediction Using Statistical and Machine Learning Methods for Improving Software Quality, J. Inf. Process. Syst. 8 (2012) 241–262.

[11] D.H. (Polo) Chau, Data Mining Meets HCI: Making Sense of Large Graphs, Carnegie Mellon University, 2012.

[12] A. Telea, L. Voinea, Visual software analytics for the build optimization of large-scale software systems, Comput. Stat. 26 (2011) 635–654. doi:10.1007/s00180-011-0248-2.

[13] R.P.L. Buse, T. Zimmermann, Information Needs for Software Development Analytics, in: 34th Int. Conf. Softw. Eng., 2012: pp. 987–996.

[14] A. González-torres, F.J. García-peñalvo, R. Therón-sánchez, R. Colomo-palacios, Science of Computer Programming Knowledge discovery in software teams by means of evolutionary visual software analytics, Sci. Comput. Program. 1 (2015) 1–20. doi:10.1016/j.scico.2015.09.005.

[15] J. Kätevä, P. Laurinen, T. Rautio, J. Suutala, L. Tuovinen, DBSA - A Device-Based Software Architecture for Data Mining, in: 2010 ACM Symp. Appl. Comput., 2010: pp. 2273–2280.

[16] M. Wermelinger, Y. Yu, Some Issues in the “ Archaeology ” of Software Evolution, Gener. Transform. Tech. Softw. Eng. (2011) 426–445.

[17] F. Fotrousi, Analytics-based Software Product Planning, Blekinge Institute of Technology, 2013.

[18] A. Begel, T. Zimmermann, Analyze this! 145 questions for data scientists in software engineering, Proc. 36th Int. Conf. Softw. Eng. - ICSE 2014. (2014) 12–23. doi:10.1145/2568225.2568233.

[19] K.M. Anderson, Embrace the Challenges : Software Engineering in a Big Data World, in: First Int. Work. BIG Data Softw. Eng., IEEE Press, 2015: pp. 19–25. doi:10.1109/BIGDSE.2015.12.

[20] X. Fern, C. Komireddy, V. Grigoreanu, Mining Problem-Solving Strategies from HCI Data, ACM Trans. Comput. -Human Interact. 17 (2010). doi:10.1145/1721831.1721834.

[21] E.A. El-sebakhy, Expert Systems with Applications Functional networks as a novel data mining paradigm in forecasting software development efforts, Expert Syst. Appl. 38 (2011) 2187–2194. doi:10.1016/j.eswa.2010.08.005.

[22] R. Hewett, Mining software defect data to support software testing, Appl. Intell. 34 (2011) 245–257. doi:10.1007/s10489-009-0193-8.

[23] H. Tribus, I. Morrigl, S. Axelsson, Using Data Mining for Static Code Analysis of C, Adv. Data Min. Appl. (2012) 603–614.

[24] M. Bruntink, Science of Computer Programming Towards base rates in software analytics Early results and challenges from studying Ohloh, Sci. Comput. Program. 97 (2015) 135–142. doi:10.1016/j.scico.2013.11.023.

[25] C. Gupta, K. Viswanathan, L. Choudur, R. Vennelakanti, P. Helm, A. Dev, et al., Better Drilling Through Sensor Analytics : A Case Study in Live Operational Intelligence, in: Fifth Int. Work. Knowl. Discov. from Sens. Data, ACM, 2011: pp. 8–15.

[26] R. Souza, C. Chavez, R.A. Bittencourt, Rapid Releases and Patch Backouts : A Software Analytics Approach Code Integration at Mozilla about Rapid Releases at Mozilla, IEEE Softw. 32 (2015) 89–96.

[27] T. Cerqueus, E.C. De Almeida, S. Scherzinger, Safely Managing Data Variety in Big Data Software Development, in: BIGDSE 2015, IEEE/ACM, 2015. doi:10.1109/BIGDSE.2015.9.

[28] R. Heimgärtner, H. Kindermann, Revealing Cultural Influences in Human Computer Interaction by Analyzing Big Data in Interactions, Act. Media Technol. (2012) 572–583.

[29] F.A. Batarseh, A.J. Gonzalez, Predicting failures in agile software development through data analytics, Softw. Qual. J. (2015). doi:10.1007/s11219-015-9285-3.

[30] M. Beller, G. Gousios, A. Zaidman, How (Much) Do Developers Test?, in: 2015 IEEE/ACM 37th IEEE Int. Conf. Softw. Eng., 2015: pp. 559–562. doi:10.1109/ICSE.2015.193.

[31] S. Banerjee, B. Cukic, On the cost of mining very large open source repositories, in: First Int. Work.

BIG Data Softw. Eng., IEEE Press, 2015: pp. 37–43. doi:10.1109/BIGDSE.2015.16.

[32] W.D. Sunindyo, T. Moser, T. Wien, D. Dhungana, Improving Open Source Software Process Quality Based on Defect Data Mining, in: SWQD, 2012: pp. 84–102. doi:10.1007/978-3-642-27213-4.

[33] M. Gayathri, A. Sudha, Software Defect Prediction System using Multilayer Perceptron Neural Network with Data Mining, Int. J. Recent Technol. Eng. 3 (2014) 54–59.

[34] A. Gonzalez-Torrez, R. Theron, F.J. Garcia-Penalvo, M. Wermellinger, Y. Yu, Maleku : an evolutionary visual software analytics tool for providing insights into software evolution Conference Item Maleku : an evolutionary visual software analytics tool for providing insights into software evolution, in: Softw. Maint. (ICSM), 2011 27th IEEE Int. Conf., IEEE, 2011.

[35] C. Rosen, B. Grawi, E. Shibab, Commit Guru : Analytics and Risk Prediction of Software Commits, in: 10th Jt. Meet. Found. Softw. Eng., ACM, 2015: pp. 966–969.

[36] C. Li, L. Huang, L. Chen, Breeze graph grammar : a graph grammar approach for modeling the software architecture of big data-oriented software systems, Softw. Pract. Exp. (2015) 1023–1050. doi:10.1002/spe.

[37] R. Liu, Q. Li, L. Mei, J. Lee, Big Data Architecture for IT Incident Management, in: 2014 IEEE Int. Conf. Serv. Oper. Logist. Informatics (SOLI), IEEE, 2014: pp. 424–429.

[38] A. Bovenzi, F. Brancati, S. Russo, A. Bondavalli, A Statistical Anomaly-Based Algorithm for On-line Fault Detection in Complex Software Critical Systems, Comput. Safety, Reliab. Secur. (2011) 128–142.

[39] A.T. Misirli, B. Caglayan, A. Bener, B. Turhan, A Retrospective Study of Software Analytics Projects : In-Depth Interviews with Practitioners, IEEE Softw. 30 (2013) 54–61.

[40] C. Lopez-martin, A. Chavoya, M.E. Meda-campaña, A Machine Learning Technique for Predicting the Productivity of Practitioners From Individually Developed 6 Software Projects, in: 2014 15th IEEE/ACIS Int. Conf. Softw. Eng. Artif. Intell. Netw. Parallel/Distributed Comput., IEEE, 2014: pp. 1–6.

[41] H. Chen, R. Kazman, S. Haziyev, O. Hrytsay, Big Data System Development : An Embedded Case Study with a Global Outsourcing Firm, in: First Int. Work. BIG Data Softw. Eng., IEEE Press, 2015: pp. 44–50. doi:10.1109/BIGDSE.2015.15.