DOI-10.53571/NJESR.2022.4.12.59-67

An Analysis Of Current Trends In Software Engineering

Subin Paul[*1]**Smt A.P.Revathi**[*2]

Head[*1&*2]

Department Of Computer Engineering

Government Polytechnic College

Vandiperiyar[*1] **Nedumangad**[*2]

(Received:10November2022/Revised:20November2022/Accepted:30November2022/Published:22December2022)

Abstract

Twenty years ago, Glass published the first report on evaluating institutions and scholars of software engineering and systems. Fund managers, young scholars, graduate students, and others can benefit from the annual, ongoing survey of publications in this field with valuable data for various purposes. However, due to a few flaws in the evaluation process, some critics have questioned the studies. On such an evaluation of academics and institutions, it is actually very difficult to reach a consensus that is widely accepted. In contrast to previous studies, this paper presents a module and automated method for software engineering assessment and trends analysis. We consider a greater number of high-quality publications, the rank of each publication analyzed, and the various roles of authors named on each paper in question to achieve a more reasonable evaluation result. The Matthew Effect is intriguing because the statistics of research subjects roughly follow power laws, as shown by the 7638 papers published in 36 publications between 2008 and 2013. The Top 20 scholars, institutions, and nations or regions are then selected using a new evaluation rule that is based on the one that is frequently used. Mark Harman of the University College London in the United Kingdom is the most highly regarded academic, followed by the University of California in the United States and the United States of America. In addition, on the basis of the EI classification system and user-defined uncontrolled keywords, we display two levels of trend changes and prominent researchers and institutions in a particular research area. We believe that young researchers and graduate students would benefit from our findings in their search for potential collaborators and comprehension of popular software engineering research topics.

Keywords: Systems And Software Engineering, Assessment, Trends Analysis, Research Publications, Power Law

Introduction

The primary means by which a field or field of study attempts to make progress is through scientific research. The analysis of research conducted within the discipline or field in question has been widely acknowledged as a reasonable and attainable method[1], primarily including assessment and trend analysis, in order to better comprehend where the discipline or field in question has been and where it may be headed. Through statistics on a large number of papers published in peer-reviewed journals, such a method presents the history and current status of a particular discipline or research field and predicts future directions. This provides various audiences with important references for various purposes. For instance, a quantitative and comprehensive assessment of scholars, institutions, and countries (or regions) is useful for evaluating the performance of research institutions and their scholars[2], and a trend analysis for a specific research field is important for newcomers looking for potential collaborative research opportunities and future research directions[3]. Software engineering is a relatively new branch of computer science research. From 1948 to the present day, its significance has been widely acknowledged by a growing number of computing scholars, making it an active and promising subfield of the field. Software engineering has long used assessment and trend analysis, as have other fields like cancer[4], agriculture, and geographic information systems. However, there are still a few issues that need to be addressed[2]. 1) The small sample size (i.e., the number of referred papers published in these journals) suggests that the results may be biased because there are only seven journals chosen as a result of a survey. (2) The keywords that were analyzed were gathered from the Top 15 scholars to best describe their research focus. As a result, they are likely to be subjective and biased, so they may not accurately reflect the software engineering trends and hot topics. (3) The proposed evaluation rule, which overlooks the leadership role of a few scholars among all authors of a multiple-authored paper, was used in the design of the scoring schemes for prominent institutions and scholars. Despite a few reports on subfields of software engineering like agile software development, the most recent paper in the annual survey of publications in systems and software engineering from 1994 has not yet been published to our knowledge. Accordingly, the primary objective of this paper is twofold: On the one hand, we will present a survey of this field's trend analysis over the past six years and a new assessment of software engineering scholars, institutions, and countries (or regions) from 2008 to 2013; In contrast, in order to carry out such a study with a greater number of publications than ever before, a more reasonable and general approach to assessment and trend analysis will be proposed. This approach will solve the aforementioned issues that have been present in previous studies. In addition, it is important to note that the research presented in this paper is actually based on empirical evidence, which means that the results may primarily depend on the analyzed data. We selected 24 prestigious journals and 12 well-known international conferences (research track) in systems and software engineering, obtained the author list, institution list, keyword list, and other information for each paper under discussion from the Elsevier EI (Engineering Village) Compendex database1, and ensured the repeatability of our findings. The following summarizes the main findings of this research: (1) Based on a large sample of 7638 research papers published in 36 distinct publications between 2008 and 2013, this paper presented two interesting results in addition to an assessment of software engineering scholars and institutions: notable scholars and institutions in a particular research field and popular research trends. (2) with regards to the various jobs of creators of a paper, this paper proposed another assessment rule for researchers, establishments and nations (or locales) in light of the one [7] much of the time utilized in earlier examinations; moreover, our strategy for evaluation and pattern examination was carried out by a product program, hence prompting computerized information handling instead of manual activity. (3) We discovered that the distributions of scholars, institutions, countries (or regions), and keywords roughly followed power laws in terms of their corresponding scores. Additionally, we theoretically demonstrated that our method's results are virtually unaffected by minor data errors, such as a small number of papers missing.

Cloud Computing

Cloud computing is a new area of software engineering research where more new techniques and models are introduced to benefit the industry and provide information with the goal of improving education and the software industry by lowering costs and enhancing existing technology[1]. According to previous research, it is difficult to determine the quality of cloud computing services because there are so many of them. This is especially true in the service industry, where the software industry and users do not see the benefits of cloud computing. As a result, Abdelmaboud et al.[2] has established five areas of research focus to enhance cloud computing services. Figure 1 depicts the areas of research. The applications that are included in systems that provide services to customers are the primary focus of SaaS research. The development of the platform resource for applications and system services is the primary focus of PaaS research.

Organizational virtualization resources and data stores are the primary subjects of IaaS research. CSP, on the other hand, focuses on the users of cloud computing services like software, software platform, and infrastructure services. Last but not least, CSC was concerned with the people and the organization that make use of cloud computing services like software and platforms. The Saad and Rana[1] research study's adoption of cloud computing for software engineering learning environments is another cloud computing-related conference paper. This paper focuses on issues related to the software and hardware tools used in a software engineering course's availability, maintenance, accessibility, scalability, compatibility, and resource utilization. One more issue featured is the readiness to adjust to this new innovation. Software engineering students face difficulties with software compatibility, availability, and licensing, and lectures are concerned about the availability of labs for scenarios involving large classes and unmanageable groups, according to the collective review of a previous survey conducted with three universities: the Asia Pacific University of Technology and Innovation Malaysia, University Technology Malaysia, and University Malaya Malaysia. In addition, the software engineering tool's cloud implementation is described in detail in Figure 2 of this paper.

Working on multiple computers and operating systems at the same time and from any location helps illustrate the advantages of cloud computing. Universities will also have top beneficiaries, allowing them to better utilize resources. In conclusion, cloud computing has been used in a variety of fields, including education and industry. It has helped users by improving quality and providing instructions on how to use cloud-based tools, which indirectly helps society.

Big Data

Operational Data (OD) is now referred to as "Big Data" because software systems use it for software design and maintenance[3]. In the software engineering field, operational support tools have long used both structured and unstructured data. Popular Big Data research topics[3] include proposing a systematic approach to the engineering field of operational database systems.

Models are made out of the mechanisms, which are used to input missing values, segment by context, and so on. Specific consideration should be taken while applying strategies in OD in light of the fact that the suspicions might be underestimated and methods may not be relevant for OD in that frame of mind for programming specifically. Implementing effective methods to identify data entry issues, cleaning data, enhancing or segmenting events, and developing robust methods for identifying subject identities are some of the areas of research that will be pursued in this area[4]. While another Big Data research paper[4] describes the environments and methods for utilizing clouds in Big Data applications, Proposed four areas of scientific and Enormous Information are information the executives, model turn of events and scoring, representation and client connection additionally plans of action. In conclusion, industries perceive Big Data as a challenge in the fight against rivals. Customers' demand will rise, revenue will rise, costs will decrease, and operations will expand in industries that are able to utilize Big Data to obtain information. Big Data still takes a long time, expensive software, large infrastructures, and a lot of effort, but cloud computing helps to increase demand at a cost proportional to that effort [4],[3].

Android Computing

Demands that are applicable to software testing methods have emerged as a result of the proliferation of Android devices and application services[5]. Unit and graphical user interface (GUI) testing of Android applications has been the focus of previous research. EvoDroid[6] is currently an evolutionary method for Android application system testing. Mahmood et al.'s [] system testing flaw is overcome by EvoDroid. 6] suggests combining two novel methods: an Android-specific program that identifies independent-searchable code segments and an evolutionary algorithm that provides information for such segments. Despite the fact that the approach[6] demonstrated the successful use of existing tools and methods for automated testing of Android applications, it may falter due to its inability to consistently reason about input conditions. The model and framework need to be expanded in this area in order to make full use of the search base algorithm. In addition, Android application can be considered as Occasion Driven Programming (EDS) that is driven by a few sorts of occasions[5]. Accessing testing methods for traditional EDS systems (like GUIs, Rich Internet Applications, embedded software, etc.) is a major issue with Android application testing. also available as a mobile application for Android[5]. Problem with automatic testing on Android The Google platform suggests methods for rapid application crash and regression testing[5],[1]. The proposed testing method is target finding runtime crashes or apparent deficiencies on changed adaptations of the application. In conclusion, Android computing research in software engineering is becoming increasingly tested for the appropriate strategy and model.

Network Security

Network security is alluring as it empowers to have direct estimation and thinks about the security level gave at various arrangements. The rank level of identified vulnerabilities, which can be quantified, is a common criticism of previous research, as is the fact that security cannot be quantified until the problem is fixed [7],[8]. According to research on a novel security metric called k-zero safety, metrics can count the number of network assets that are responsible for vulnerabilities. Network hardening and submissions are two methods for implementing K-zero safety. The k-zero networks are made more vulnerable to larger vulnerabilities by network hardening[8]. Increasing diversity, strengthening isolation, deactivating services, and firewall attacks are all examples of network hardening. By patching vulnerabilities that are associated with those services, sub metrics applies modeling and quantifies them. This presents a chance to select various network hardening solutions[8]. The proposed safety model is effective in selecting the most suitable value metric. To rank the k-zero day vulnerabilities in handling inputs that are known to have vulnerabilities[8] in an application service, further improvements and evaluations are required. However, measurements of existing networks have become more and more important in research on network security. Study on the single-broadcast Local Area Network (LAN) (Ethernet)'s security measures[7]. A progressive model was proposed in explaining the interruption location system in network security. Preparation phase, attack phase, and post phase are the three phases of an attack. During the preparation phase, the attacker acquires general network information. During the attack phase, the network is accessed from another machine and logged remotely. Last but not least, the post phase refers to the ongoing modification of the system following an Ethernet breach[7]. This model is advantageous for an open real-time environment. In conclusion, software engineering research into network security is moving away from the topology area and toward metrics or framework extensions regarding this technology.

Software Engineering Project Management

The goal of software engineering project management is to oversee the necessary set of activities and tasks for software projects[9]. These issues include software requirements, incomplete project planning, difficult-to-prepare software costs and schedules, and the criteria for selecting the best methodology for software project analysis, design, testing, and management[9]. Researchers don't know much about the strengths and weaknesses of a particular software type. Planning for software engineering projects helps ensure that they are completed on time by establishing objectives, goals, strategies, policies, course of action, and decision[9],[10]. In addition, recognizing team members' qualifications, technical skills, and experience is one way to increase success chances by putting the right people on the right project team[10].

Conclusion

As we probably are aware, the evaluation of logical examination is definitely not a basic work. For such a study, reaching a widely accepted evaluation method is extremely challenging. Prior research on the evaluation of academics and institutions has been reported, despite the fact that software engineering is a relatively new field. A software-aided approach to assessment and trend analysis is presented in this paper and can be utilized in software engineering as well as other computer science research fields. When compared to the method used in previous studies, the one proposed in this paper is automated and modular. In addition, it considers additional publications, including conference proceedings, the rank of each analyzed publication, and the various roles that authors play in the creation of a paper. This paper presents two levels of research trend changes and those notable scholars and institutions in a particular research field, in addition to the evaluation of scholars, institutions, and countries/regions, according to the unified data source of the EI Compendex database. As a result, we believe that the findings could be of use to new or upcoming researchers in the selection of suitable potential collaborators and popular software engineering research topics. Students studying software engineering can benefit from this article by learning about the most recent trends in the field of research, as well as how to fill in the research gaps and plan for the future as outlined in the reviewed papers.

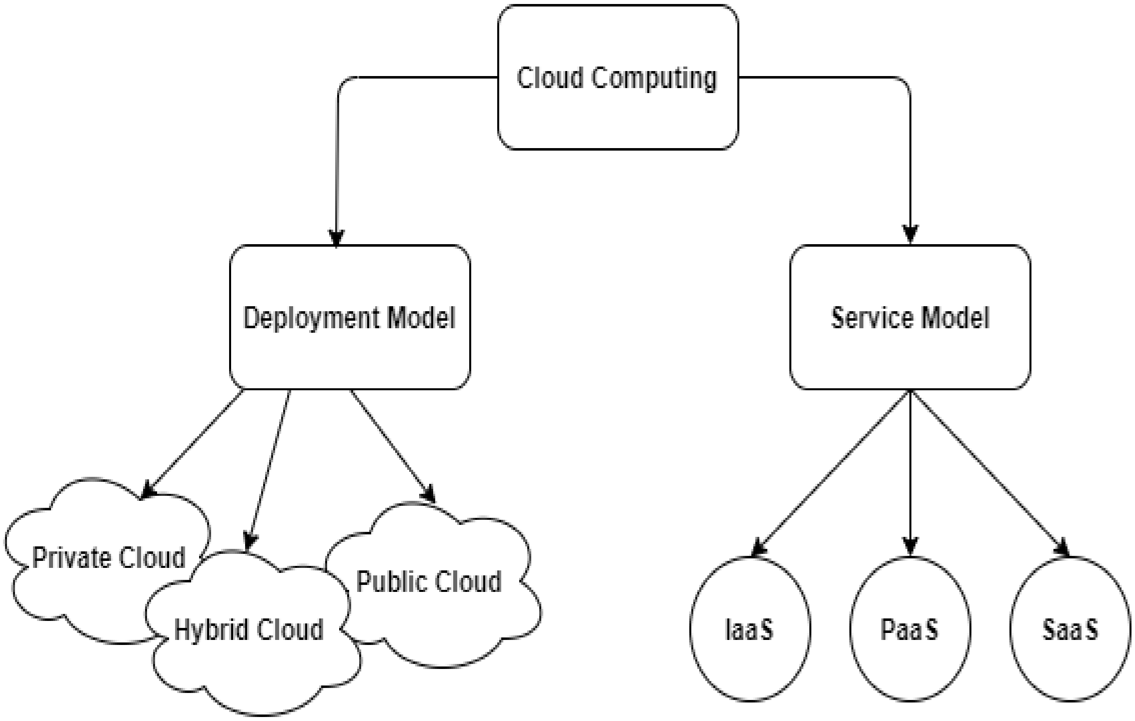

Figure-1. The Characteristics Of Quality Of Services In Cloud Computing [2].

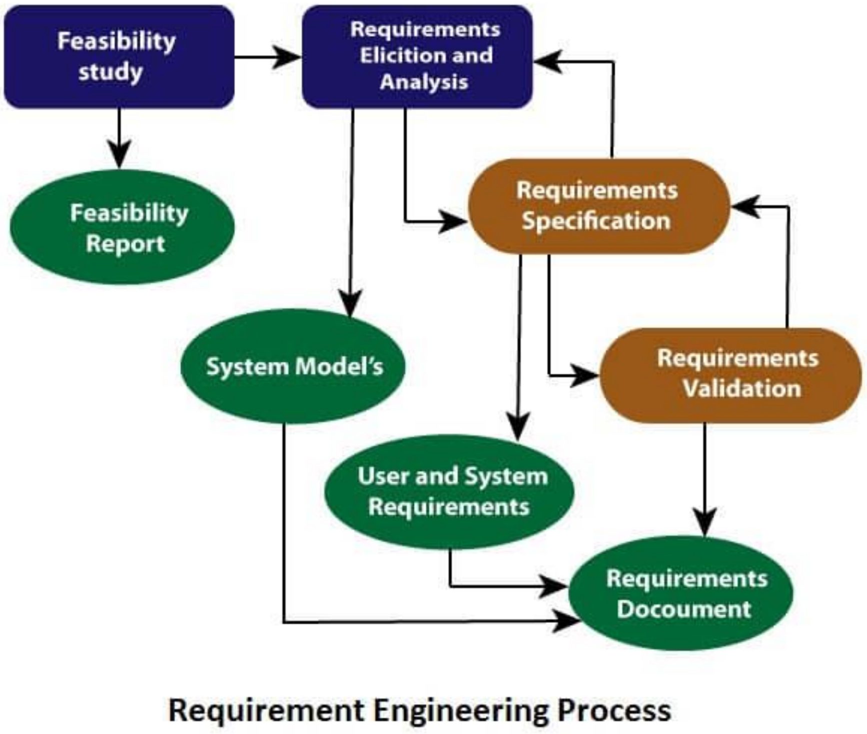
Feasibility study
Requirements Elicition and Analysis
Feasibility Report
Requirements Specification
System Model's
Requirements Validation
User and System Requirements
Requirements Docoument
Requirement Engineering Process

Figure-2. Guidelines In Implementing Software Engineering Tools On Cloud [1]

References

[1]. Gurcan, F., Dalveren, G. G. M., Cagiltay, N. E., & Soylu, A. (2022). Detecting latent topics and trends in software engineering research since 1980 using probabilistic topic modeling. IEEE Access, 10, 74638-74654.

[2]. Gurcan, F., Dalveren, G. G. M., Cagiltay, N. E., Roman, D., & Soylu, A. (2022). Evolution of Software Testing Strategies and Trends: Semantic Content Analysis of Software Research Corpus of the Last 40 Years. IEEE Access, 10, 106093-106109.

[3]. Yang, Y., Xia, X., Lo, D., & Grundy, J. (2022). A survey on deep learning for software engineering. ACM Computing Surveys (CSUR), 54(10s), 1-73.

[4]. Martínez-Fernández, S., Bogner, J., Franch, X., Oriol, M., Siebert, J., Trendowicz, A., ... & Wagner, S. (2022). Software engineering for AI-based systems: a survey. ACM Transactions on Software Engineering and Methodology (TOSEM), 31(2), 1-59.

[5]. Baham, C., & Hirschheim, R. (2022). Issues, challenges, and a proposed theoretical core of agile software development research. Information Systems Journal, 32(1), 103-129.

[6]. Agarwal, S., Sonbhadra, S. K., & Singh Punn, N. (2022, June). Software Testing and Quality Assurance for Data Intensive Applications. In Proceedings of the International Conference on Evaluation and Assessment in Software Engineering 2022 (pp. 461-462).

[7]. Obaidi, M., Nagel, L., Specht, A., & Klünder, J. (2022). Sentiment analysis tools in software engineering: A systematic mapping study. Information and Software Technology, 107018.

[8]. Baltes, S., & Ralph, P. (2022). Sampling in software engineering research: A critical review and guidelines. Empirical Software Engineering, 27(4), 1-31.

[9]. Laato, S., Mäntymäki, M., Islam, A. K. M., Hyrynsalmi, S., & Birkstedt, T. (2022). Trends and Trajectories in the Software Industry: implications for the future of work. Information Systems Frontiers, 1-16.

[10]. Ali, S., Baseer, S., Abbasi, I. A., Alouffi, B., Alosaimi, W., & Huang, J. (2022). Analyzing the interactions among factors affecting cloud adoption for software testing: a two-stage ISM-ANN approach. Soft Computing, 1-29.

[11]. Lu, Q., Zhu, L., Xu, X., Whittle, J., Douglas, D., & Sanderson, C. (2022, May). Software engineering for responsible AI: An empirical study and operationalised patterns. In Proceedings of the 44th International Conference on Software Engineering: Software Engineering in Practice (pp. 241-242).

[12]. de la Barrera, A. G., de Guzmán, I. G. R., Polo, M., & Cruz-Lemus, J. A. (2022). Quantum Software Testing: Current Trends and Emerging Proposals. In Quantum Software Engineering (pp. 167-191). Springer, Cham.

[13]. Faruk, M. J. H., Subramanian, S., Shahriar, H., Valero, M., Li, X., & Tasnim, M. (2022, May). Software engineering process and methodology in blockchain-oriented software development: A systematic study. In 2022 IEEE/ACIS 20th International Conference on Software Engineering Research, Management and Applications (SERA) (pp. 120-127). IEEE.

[14]. Peruma, A., Simmons, S., AlOmar, E. A., Newman, C. D., Mkaouer, M. W., & Ouni, A. (2022). How do i refactor this? An empirical study on refactoring trends and topics in Stack Overflow. Empirical Software Engineering, 27(1), 1-43.

[15]. Abdullin, A., Akhin, M., & Belyaev, M. (2022, May). Kex at the 2022 SBST Tool Competition. In 2022 IEEE/ACM 15th International Workshop on Search-Based Software Testing (SBST) (pp. 35-36). IEEE.