Journal of Positive School Psychology http://journalppw.com
2022, Vol. 6, No. 8, 1051-1056

Potential Challenges For Software Engineering Applications And Practical Solutions

Nguyen Tan Danh

Faculty of IT, FPT University, Vietnam, Email: DanhNT16@fe.edu.vn

Abstract

Practical studies show that, in order for users to use smart devices more, use smart applications more often for different purposes, applications must meet the requirements of use. their application, must be easy to use, stable and reliable. The era of information technology development with countless advancements and economic development. This is the same thing that requires an increasing human capacity to process and store information based on two factors: efficiency and accuracy. However, in the process of using smart applications, there are many problems that have been encountered related to the quality of the software. In the article, some challenges facing Software Engineering industry are mentioned through comparison of data and some solutions mentioned are most practical to solve the problem.

Keywords: Software engineering, software, applications, challenges.

1. Introduction

Recent years have seen the rapid growth of malware both in number and type. Today, this number has increased to more than 800 million malicious code samples by the end of 2018. In the face of the rapid development and heavy destruction of malware, there have been many studies on the world. methods for detecting and removing malware, but generally these studies revolve around two main methods of anomaly or anomaly and signature-based detection. The concept of data mining for malware detection is mentioned. There are three different static properties for malware classification: Portable executable, strings, and byte sequences [1]. A method of visualizing and classifying malware using image processing techniques, this method visualizes malicious files as grayscale images in question. Use K nearest neighbor with Euclidean distance for malware classification. It has been found that the classification using this method is faster, more scalable and comparable to dynamic analysis in terms of accuracy [2]. A framework for automatic malware classification based on the structural information of the malware was also detected [3]. The most important problem to properly detect malicious code is to have a really effective feature extraction method, a new malware detection method using n-grams byte signature, Abou-Assaleh's experiments almost absolute accuracy has been achieved [2].

In recent years, Vietnam's software industry has made remarkable leaps and bounds. In 2020, the revenue of the software and digital content industry will reach over 6 billion USD, 2 times higher than the revenue in 2015 (3 billion USD). In just the first 6 months of 2022, the ICT industry has brought a total revenue of more than 72 billion USD. But the bigger and higher goal is to affirm the position of Vietnam's software technology on the world IT map. Software does not require complex machines to develop, it can be created on personal computers that are accessible to most people in society today [4]. This creates the illusion that anyone who just needs to learn some simple programming languages can develop software. At the same time, the invisibility of software makes users tend to invest and pay attention to hardware devices rather than to the software itself. Sometimes, some modules are created by experts in the field themselves, rather than software engineers, especially for applications in science[2].

Looking back, we can see that the lack of expertise or the lack of stages in the software development process caused software errors. This will have serious consequences or if the software is to be repaired there will be a fraction of the costs incurred [3]. In recent years, the development of technology has led to more and more automation in stages such as testing, deployment, and management of new software,

while allowing researchers and practitioners to identify new approaches to creating and operating software and services. On the other hand, software and interactive devices sometimes change due to various reasons, such as interface and implementation errors, changing requirements, etc. To build and manage software efficiently, we need to strengthen the management methods, improve the quality of software and applications [5].

2. Explosion of apps and providers

Digital transformation has seen us witness an explosion of these applications, software and application vendors for businesses. Think about how many apps and programs you're using to work in a day [4]. The number of marketing tools has increased extremely fast in the past few years. Any part of the business now relies on certain types of work-related applications. According to Netskope, the average business has more than 800 cloud applications.

Most cloud-based software is available for anyone to use, free or paid. With the growth rate of the marketing tools above, you can imagine the number of applications must be a huge number. The ease of access, coupled with the variety of new tools, creates a renewed need for users, even though the chances are high that the tools they need already exist.

Using too many new apps or apps every week poses a serious threat to IT department protocols and business security. Dealing with the sprawl of applications becomes a real headache if the business is not prepared for this [6].

3. Current status of software projects

In which, the export of hardware and electronics continues to be the mainstay with export turnover reaching 57 billion USD, up 16.4% compared to the first 6 months of 2021. In which, computer exports reached 29.1 billion USD increased by 21.8% and phone exports reached 27.9 billion USD, up 11.2% over the same period [7].

Comments from other independent entities such as HSBC also show that exports are the main driving force of the ICT industry. Accordingly, Vietnam is currently the world's No. 2 smartphone producer, accounting for 13% of the total number of smartphones being sold worldwide, just behind China with 50%. Not only that, Vietnam is also one of the countries that provide microprocessors for assembled electronic devices around the world [5].

Current software development trends are customer-centric, the goal is to provide customers with a great experience. In recent years, the software industry has developed strongly in the following areas: cloud computing; big data analysis and processing, intelligent application development, multi-sensory, multi-channel experience. Along with design and development trends, software projects have been developing new software engineering approaches, transitioning from full applications to applications developed based on the application of software engineering. microservices model and the need for new approaches that facilitate service/application composition. The above-oriented development has encountered technical difficulties in implementing software projects as follows.

The transition from application development approach to application component approach is considered very important. In this direction, supporting software reuse through software development based on microservices combined with software component frameworks must be implemented. In fact, the transition from all-code-heavy applications to smaller, self-contained services has already been made [3]. Applications with a microservices architecture consist of a set of independently deployable services. Such services can be combined to create a service chain graph and support advanced functions. The implemented microservices must be validated against a uniform representation model regarding their interface characteristics and interfaces. Therefore, such services must be software developed by design, by applying reusable design patterns, separation of concerns and high-level modeling, supporting the exploration and understand complex software systems and refactor them accordingly [6].

Within software design challenges also include challenges for the development of systems that are self-adapting, responsive, fault-tolerant, self-healing, developed software components, etc. development must respond by design to changes in the operating environment. During

software design, it is necessary to take into account the variability of highly distributed applications in heterogeneous environments [5].

Among the middleware deployment and orchestration challenges are deploying applications on a programmable infrastructure in a way that is optimal in terms of software performance as well as service provider policies. high. Therefore, the software delivered must be modeled in a way that provides flexibility to the user (or automated deployment tools) to deploy it optimally. This mostly applies to applications with concepts. Furthermore, in this direction, data location, data volatility, and serious timing issues drive the definition of a set of requirements that must be met. It's critical to be able to manage ready-to-run workloads designed for physical, virtual, and cloud environments using single templates for all public, private cloud environments or mixture. From the challenges mentioned above in this category, the most common challenges are the need to provide cloud-based tools and services for rapid software prototyping and the need to manage software complexity. large software and data-intensive systems [4].

Finally, regarding service/application lifecycle management challenges, the focus is on the adoption of model-driven development techniques.

Managing development complexity and risk in both the design and runtime phases is considered critical, to increase the quality of the software, to reduce the time. From the challenges mentioned above in this category, the most common challenges are the need to provide cloud-based tools and services for rapid software prototyping and the need to manage software complexity. large software and data-intensive systems [5].

4. Hidden challenges of software engineering

New challenges emerge as technological advancements and new concepts emerge, while existing challenges also involve new turning points and subsequent new research and transformation activities that are needed to master act and solve problems effectively and efficiently. Some of the future challenges of Software Engineering can be mentioned as follows.

Software process is an area of study that has been thoroughly explored, but today there are a number of new advances in technology and practice that cause significant changes in this aspect. Software is measured in terms of usability, reliability, and scalability. New capabilities to easily collect user feedback and monitoring information enable fully informed software development while shorter development cycles require new software production methods really effectively allows for controlled management of such short development cycles. In addition, efficient use of resources and support for architecture-level analysis, optimization of deployment decisions, and automatic orchestration and orchestration of applications/services and application of methods infrastructure as a code approach to eliminate the need to configure and manage infrastructures to be programmable, with a focus on using them from middleware.

Currently, software development products are developed on application models of virtual reality systems and networks of physical devices linked over the internet. The development of services, applications has a lot to do with the ability to adapt to real-time changes, thus allowing for different application contexts. Key challenges presented by IoT-enabled CPS include the development of design models, methods, and tools for IoT/CPS-enabled applications that go beyond studying formal methods for creating abstract concepts and forms to construct and reason about diverse systems of components. Furthermore, the needs brought about by CPS with new approaches to software adaptability, scalability, and maintainability are sometimes not taken into account when designing in large-scale open CPS environments [3].

As suggested above and bearing in mind the different application contexts, further research is also needed in developing design patterns for the systems approach. New patterns at the architecture level describe the obligations/constraints that must be fulfilled by the system on which the software is running [5]. And to validate and standardize them are necessary and methods of applying them to the ever-changing context environment. Therefore, issues such as frameworks of reference, design architecture and interoperability, modeling

languages, tool integration and simulation and analysis, etc. must be dealt with.

The rapid growth over the years of agile delivery methods, as well as the need to reduce software development time, however having to take a research approach can increase system quality, reduce service recovery wait times, and develop agile methodologies for testing quality through the process of software testing [6].

Another core aspect that has a direct impact on any software operation is requirements engineering. New devices, services, and even individuals become part of the software-enabled ecosystem. Flexibility, continuous evolution, and interconnection are at odds with current engineering requirements output, as current approaches sometimes do not take into account additional in-process functionality and request is undefined. A radically different approach is required to capture new behaviors from systems and users. New technologies and trends are shedding light on potential research topics such as multi-channel big data analytics to collect requests from large-scale webs (such as connected smart city infrastructures) combinations of people, machines, and generally system characteristics and behaviors), new methods for interacting with users that are geared towards directly extracting requirements, indirect requirements corresponding to The indirect extraction model exploits the contextual perception of individuals independently of the use of a particular software [7].

Particular attention must be paid to privacy and security in complex distributed systems that in many cases have to handle large volumes of data in a distributed manner [2]. Particular emphasis should be placed on the topic of security and privacy by designing software engineering approaches that contribute to the creation of software products that can operate in a multi-IT infrastructure environment with increased security features [9].

Studying software engineering challenges in this direction includes new tools that use machine learning and data mining techniques to reveal hidden knowledge aspects and extract information from exploiting knowledge that humans cannot dig up, but needs human attention and affection to improve software quality. Research the evolution/decomposition of application frameworks, analyze trends, preferences and user behavior with systems to better understand user needs, tools and methods to identify performance and feature improvement opportunities, identify root causes of errors and system crashes based on log files or rapid updates coming from systems and disparate complex distributed infrastructures, insights collected at runtime about symptoms and context changes that trigger adaptive actions, and perform predictive and description to proactively plan and prepare adaptation actions [4].

In summary, the challenges of Software Engineering include both developing methodology and developing supporting tools to detect and deal with inconsistencies and gaps in requirements specifications, knowledge and skills [1]. Furthermore, software production processes also include organizational challenges that must be met with an interdisciplinary approach to creating and managing communities of code contributors, reviewers, testing, first-level users,... and comprehensive development and communication methods that combine existing tools under a common set of formalized, methodological sets [3].

5. Suggestions

As can be clearly seen, it is very difficult to optimize these properties simultaneously. Attributes may conflict with each other, such as efficiency and ease of use, maintainability. The relationship between improvement cost and performance for each attribute is not linear. Many times, a small improvement in any attribute can be very expensive [9].

Another difficulty of software development is that it is difficult to quantify the properties of the software. We lack software quality metrics and standards. The issue of price must be taken into account when building a software. We will be able to build a software no matter how complicated it is if there are no time and cost constraints. It is important that we build good software at a reasonable price and on a predetermined schedule [3].

We can see that the top difficulty of software development is due to the nature that software is a logical system, not a physical system. It

therefore has characteristics that are significantly different from those of the hardware. Here are the three main factors that create complexity in software development, use, and maintenance.

The first is that the software is not made in the classical sense. Software is also designed and developed like hardware, but it is not predefined. It is only when the development is done that people have a specific product and understand if it works or not. That is, in the intermediate steps, it is very difficult for us to control the quality of the software.

The cost of hardware is mainly driven by the cost of raw materials and is relatively easy to control. Meanwhile, software costs mainly focus on labor costs. The software development process depends on people (knowledge, ability to apply, experience and management) and is developed under diverse and non-diversified (technical, social) environmental conditions. stop changing. Therefore, it is difficult for us to estimate the cost and effectiveness of the software.

Second, software does not degrade but degrades over time. Software doesn't respond to environmental influences that cause hardware to age, but it also degrades over time. In fact, software goes through life needing to be changed (maintained) to meet the ever-changing needs of the organization that uses it. Every time a change is made, there will be a number of new defects that inevitably make the number of potential bugs in the software increase. Gradually, software degraded due to the increasing failure rate to the point of causing unacceptable damage [4].

Software maintenance is much more complex and different in nature from hardware maintenance due to the complexity of the software system and the unavailability of replacement parts for the defective part. We do not replace the defective part with the existing one, but actually create a new module. Therefore, usually only the software manufacturer can maintain (repair) the failure. It is difficult to estimate the cost of software maintenance [7].

Third, most software is built from scratch, rarely assembled from pre-existing components. The software environment (including hardware, background software, people, and organizations) is unpredictable and changes frequently.

These factors lead to high software costs and it is difficult to secure a schedule for software development.

6. Conclusion

In the article, an overview with the current and future challenges of software engineering has been mentioned. Software systems reside in a complex and hyper-connected ecological system. Society is increasingly dependent on software in many areas, such as entertainment, education, politics, industrial and civil infrastructure, economic and business initiatives, as well as work and other individual activities in many other fields. All of these areas have become closely intertwined with software systems and applications. Thus, software development is a field that requires the support, improvement and research of new methods and support tools applied in the software development process, along with the complete development of software development. The knowledge and skills of the human factor will bring expectations to make the software development process easy, ensuring the required criteria for quality and security and usability of the software application.

References

[1] Lim, S. L., Bentley, P. J., Kanakam, N., Ishikawa, F., & Honiden, S. (2014). Investigating country differences in mobile app user behavior and challenges for software engineering. IEEE Transactions on Software Engineering, 41(1), 40-64.

[2] Casale, G., Chesta, C., Deussen, P., Di Nitto, E., Gouvas, P., Koussouris, S., ... & Zhao, Z. (2016). Current and future challenges of software engineering for services and applications. Procedia computer science, 97, 34-42.

[3] Robillard, M., Walker, R., & Zimmermann, T. (2009). Recommendation systems for software engineering. IEEE software, 27(4), 80-86.

[4] Vogel-Heuser, B., Diedrich, C., Fay, A., Jeschke, S., Kowalewski, S., & Wollschlaeger, M. (2014). Challenges for software engineering in automation. Journal of Software Engineering and Applications, 2014.

[5] Crnkovic, I. (2001). Component-based software engineering—new challenges in software development. Software Focus, 2(4), 127-133.
[6] Ghezzi, C., & Mandrioli, D. (2005, May). The challenges of software engineering education. In International Conference on Software Engineering (pp. 115-127). Springer, Berlin, Heidelberg.
[7] Lim, S. L., Bentley, P. J., Kanakam, N., Ishikawa, F., & Honiden, S. (2014). Investigating country differences in mobile app user behavior and challenges for software engineering. IEEE Transactions on Software Engineering, 41(1), 40-64.
[8] Crnković, I. (2003). Component-based software engineering-new challenges in software development. Journal of computing and information technology, 11(3), 151-161.
[9] Zambonelli, F., & Omicini, A. (2004). Challenges and research directions in agent-oriented software engineering. Autonomous agents and multi-agent systems, 9(3), 253-283.