Wilfrid Laurier University - CP317 Software Engineering 2021

Research Paper

THE PAST, PRESENT, AND FUTURE OF SOFTWARE ENGINEERING

Group 15

Ans Riaz [1]

BSc. Computer Science, Wilfrid Laurier University, Waterloo, Ontario, Canada

Tirth Desai [2]

BSc. Computer Science, Wilfrid Laurier University, Waterloo, Ontario, Canada

Hamza Ghaffar [3]

BSc. Computer Science, Wilfrid Laurier University, Waterloo, Ontario, Canada

Methershan Balakrishnar [4]

BSc. Computer Science, Wilfrid Laurier University, Waterloo, Ontario, Canada

ABSTRACT

Software engineering is a relatively new field when it comes to other professions only really gaining traction in the late 1950's. Since then the world of software engineering has gone through major changes and is now today considered one of the most lucrative and fastest growing industries in the world with the average job outlook around 22% in America compared to the national average only being at 7%. With the field of software engineering now becoming a crucial part for the development of the modern era, it is clear that this industry will continue to grow at a rapid rate and will become more and more embedded into our lives, especially with the recent rise of the research and development within the industry.

Keywords: software engineering, industries, research, development, artificial intelligence, blockchain, quantum computing, programming, tools

1. Introduction

The primary goal of this paper is to provide information and research on software engineering and its industry. In particular, we will be targeting three (3) main research questions.

- Research Question 1: What is the history and rise of software engineering?
- Research Question 2: Where is software engineering today?
- Research Question 3: Where do we see software engineering in the future?

As the world slowly began its transition into technology, we quickly entered a so-called “software crisis” where problems stemmed from an inability to apply techniques used to develop small software systems to the development of larger and more complex systems. We had difficulty writing useful and efficient programs in the required time and this was mostly due to the rapid increase in computer power and the complexity of the problems that could be tackled.

This research paper will be organized into sections, with each section representing a phase in software engineering. Section 2 will address the history and rise of software engineering (1950-2010). Section 3 will cover the current and present phase of software engineering, and section 4 will focus on the future and we can expect this growing field to reach.

2. History and Rise of Software Engineering

Software engineering has rapidly developed over the past few decades. In the 1950s computer programs were just instructions to manipulate a physical device. The first programming language that was widely used was Fortran by IBM released in 1957 for mathematical and scientific computing purposes.

Moving on to the 1960s, software engineering was officially accepted as a form of engineering, however, this was a rocky time for the field. The next programming language emerged in 1962 named Cobol. This language was released by the US department of defense and it began to be used in business applications. The term “software engineer” was coined by Margaret Hamilton in 1963/1964 who believed that software developers had earned the title. In the mid 1960s the “software crisis” had started. As software development struggled to keep up with hardware developments. There were many issues that contributed to this struggle, some of which were; software development was running over budget and past deadlines, extensive debugging was required, user needs were often not met or the program was simply not completed, and large amounts of maintenance was required. The “NATO Software Engineering Conferences” were held in 1968 and then in 1969 with the intention of addressing these issues. These conferences made it possible to establish common guidelines and practices for software development resulting in the foundation of software engineering we know today.

The 1970s were a significant period for software engineering as new ideas, languages, and hardware arose. In 1970 the Pascal programming language was introduced by Nicklaus Wirth. Pascal used structured programming and data structuring such as conditional and loop control structures. The C programming language is now one of the most significant programming languages in the world and its development began with a man named Dennis MacAlistair Ritchie in 1972. He also developed the Unix Operating Systems along with Ken Thompson which debuted in the same year. Then, in 1975 the first personal computers were designed for business use. The PC’s introduction to business would revolutionize how business operated. Finally, in 1979 Seattle University began to offer a master’s degree program in computer engineering boosting the appeal of the software engineering field.

The software crisis lasted about two decades before it finally began to wind down in the 1980s. Newer and more advanced programming languages and tools were developed, which supported software engineering to move towards a more “object oriented design” for programming. Two main programming languages were rolled out; first was the Ada programming language designed by Jean Ichbiah in 1980, and the second was the C++ programming language conceived and developed by Bjane Strousop in 1985. C++ was significant proof of progress in the field as this language had more functional, generic, object oriented and procedural features. Computer Aided Software Engineering tools hit the market in 1982.

CASE tools are a set of software applications designed to improve the quality of systems developed while also reducing cost and the development time by automating activities. The Internet is one of the most important and significant tools used around the world and it was born in 1989 and at the time it was mainly only used by the military and scientists.

The 1990s were very important for software engineering as many colossal advancements were made. New and improved programming languages, the internet, and new approaches to software development were introduced. 1990 brought us the WorldWideWeb, the first web browser developed by Tim Berners-Lee. He also created HTTP and HTML which were essential in web development. From 1991 to 1995 we had three more programming languages introduced, the first of which was Python. This language was known for its large standard library and liberal use of white space at the time. The Java language followed in 1995, developed by James Gosling who originally developed it for interactive television, however, the language proved to be far too advanced for such a use. Finally, in 1995 JavaScript was released. Javascript allows for interactive design on web-pages and is used by the majority of websites. Following the example of Seattle University, in 1996 Rochester Institute of Technology introduced the first bachelor's degree program and in 1998 the US Naval Postgraduate School offered the first doctorate program for software engineering. Both of these programs further instilled the appeal of software engineering in the world. A new approach to software development named Extreme Programming was introduced by Kent Black in 1999. This was an agile software development method which allowed quick response to changing requirements from customers. This was done by making changes in small test-driven steps while integrating the program several times a day.

The next decade was less focused on creating new languages or tools to improve what was already existing in the software field. Developers focused on creating or improving new methods of software development which allowed the field to be more responsive to customer needs, be more profitable, and make it easier to create software products. 2001 brought us the Manifesto for Agile Software Development which focused on developing software products through collaboration and multiple teams working together with customers. This also led to Scrum, an agile process of development which used interactive and incremental models. It was introduced by Ken Swhwaber and Mike Beedle and it was in practice since the 1990s, however, it only became widely used in the 2000s. Finally, in 2004 the Software Engineering Body of Knowledge was introduced. This was an international guide for software engineering knowledge meant to bring further standardization to the field. It addressed software issues including but not limited to design, constructions, and maintenance of software products.

3. Software Engineering Today

Software Engineering is much like age with respect to time, that is. At the time of writing this, software engineering is at its most advanced level it has ever been and also the least advanced it will ever be. Software Engineering today is one of the most sought-after careers, with implications in different industries and educational courses being offered at various levels due to the exposure that people have had to technology. Today many companies have an IT department which houses software engineers to minimize human effort, increase efficiency and maximize productivity. Many companies are shifting towards an automated approach which will help reduce costs, limit errors, and increase the quality of goods produced and services provided to its targeted demographic. Automation is possible through software engineering techniques and is considered to be one of the most important uses to come out of the ever-advancing software engineering world.

3.1. Industry outlook and demographics

Currently there are around 45 million software engineers working worldwide in an abundance of industries ranging from top tech companies, pharmaceuticals, gaming, and even small start ups. Globally software engineering is a highly lucrative field to work with job growth and outlooks easily beating the national averages set by countries. The Asia Pacific region shows the strongest growth in the world with countries such as China, India, Pakistan, and South Korea producing a high number of companies that are on route or have already achieved unicorn status. Moving over to Europe, around 5.5 million software engineers are currently employed, with Germany having the highest quantity within its job market with an estimated 901 thousand professionals. Also the split between male and female developers is about 2.5 to 1.

3.2. Emerging tools and technologies

As time passes it seems as if more programming languages and tools are coming. This is because technology is constantly changing and therefore we need more tools that can make software for these technologies. There are a multitude of emerging programming languages and all are released with a specific solution they're supposed to solve. Some examples are functional languages such as Elixir or Elm, procedural languages such as Go, object-oriented languages such as Dart, Pony, or Typescript, and even languages that give you a little bit of everything such as Hack, Kotlin, Julia, OCaml, and many more. Now going back to programming languages that are more commonly used, JavaScript tops the list as the most used language with about 12.4 million developers and steadily rising. Python comes in second with 10 million developers and Java comes in third with 8 million developers.

Each software engineer typically will specialize within fields that they are interested in. With new technologies such as cloud computing. machine learning, and IoT on the rise, programming languages that are most heavily linked to those will also rise in users. Python is currently the most popular when it comes to machine learning, artificial intelligence, and IoT (internet-of-things) applications but it is avoided when it comes to gaming and mobile apps as it lacks the necessary tools. On the other hand both Java and Javascript are heavily used in industries where web, mobile, and cloud computing is done.

3.3. Modern Software Engineering

This is the practice of integrating the most advanced technology and traditional software development. This practice is especially useful for creating prototypes. This is because it is easy to see the new technology that is added as its foundations are laid in the basic development phase making the difference apparent. This helps find problems at an early stage which we have learnt is the most cost efficient stage to find any faults. Another advantage it serves is to help identify which aspects of the developed product should be retained. These continuous advancements also create a disruptive environment businesses can not afford complacency. The disruption is because of the transformation that is required to move towards modern software development. Employees need to stay up to date with these advancements or risk missing out on opportunities and being left behind. In the course we have discussed various methods used to shift towards a new system or software practice.

3.4. Search Engine Optimization

Search Engine Optimization (SEO) improves the traffic towards a website or app from search engines through quality or quantity. To simplify it takes online content of a website or an app, optimizes it such that search engines like google, bing etc. will display it further up in their search results. This helps the website generate more traffic which in turn can lead to better business. $79.27 billion is the estimated spend by agencies and brands in the United States for SEO services in 2020. (Borrell Associates) 2 in 3 smartphone users are likely to purchase from a company that has mobile sites or apps that customize information to their location. (Think with Google) 90% of consumers used the internet to find a local business in the last year. (BrightLocal) These statistics back the heavy spending by companies on SEO. Also making it the hottest trend in Software Engineering. Businesses have also opted to in-house their SEO specialists rather than outsourcing this component as it is the best form of marketing. This is due to the fact that no innovative marketing strategies are required rather a refinement of the current marketing strategies.

4. Software Engineering in the Future

As we enter the next phase of software engineering many new niches and trends are on the rise. Niches such as cloud services, artificial intelligence and machine learning, blockchain technology, cybersecurity, as well as quantum computing.

4.1. Artificial Intelligence

Since the history of computing and software engineering, humans have always been attracted towards the concept of automation, AI, and technology as a whole and what its potential growth is. Mimicking the human way of thinking inside a machine makes up the Artificial Intelligence field, as, generally the main purpose behind Artificial Intelligence and its applications is to create systems that can understand, solve problems and/or tasks intelligently, independently, and effectively.

Google AutoML is a Machine Learning system that is able to create another ML system. Google essentially started with the AutoML vision system which allowed anyone, including non ML developers to upload images to the AutoML vision cloud platforms. The AutoML system will then find patterns in the images uploaded to create custom visioning system models employed for specific business needs. By utilizing the AutoML system, it can potentially help discovering patterns related to translation, speech, video, and even natural language recognition. Furthermore, in hopes of making Artificial Intelligence solutions affordable to anyone, Google announced the TensorFlow AI library, an online open source platform containing free and ready to use open source machine learning code. Google is hoping it allows other developers to use it as a way to accelerate improvements on the domain, which could lead to further developments in AI systems that can be used in the different phases of software development phases.

With companies such as Google cementing their position within the AI world, many other companies are already focusing on incorporating AI into their main business. A very well known current example of this are Tesla and other up and coming electric EV companies. Tesla in specific has been focusing on bringing their revolutionary AI and autopilot technology to the mainstream market by being able to safely take passengers from point A to point B. With AI being incorporated into their vehicles, Tesla has also expanded their focuses into developing the next generation of automation. A general purpose, bi-pedal humanoid robot that is capable of performing unsafe, repetitive, and boring tasks. This is a clear example of where the concept and methodologies of AI and automation can flourish and grow in the future. With the advancements of AI, we can reduce human accidents in the workplace by replacing them with humanoid robots that simply do not make mistakes and if any accident were to occur then minimal human injury would take place.

General or Strong Artificial Intelligence is the concept of when an AI systems' intelligence will reach and possibly exceed that intelligence level of a human. Hundreds of researchers and AI experts were asked when they thought AI would reach the status of strong AI and most agreed that we would not reach that state before the 2040's (Bostrom, 2014). But as AI continues to advance and evolve it is inevitable that the consequence of this may be the possible replacement of software engineers as early as the 2050's. Though this is a possibility, this could lead to positions such as AI software moderators whose role is to help moderate and maintain possible issues that may arise. Also as we transition into a new phase of development and software engineering, especially with AI becoming more common, it is evident that this will require new ways of writing and managing code and can possibly result in new methods within the software engineering world being developed to accommodate upcoming changes and advancements.

As the software engineering field changes and advances, new techniques are expected in the software development process. A possible example includes easy and human-friendly software abstraction layers. This process could potentially aid and make the coding process easier. This outcome was expected as experts have always envisioned AI systems being used for creating codes by exporting or reusing already existing codes based on user requirements (Zohair, 2018).

4.2. Cybersecurity

With Artificial Intelligence and Industrial IoT set to become a $500 billion dollar market by 2025 it is only fair to say that digitally connected ecosystems and industries will become increasingly appealing to cybercriminals and this will require a rise in cybersecurity to maintain and protect these robust complicated systems from potential external threats.

Cybersecurity also links 4.1. Artificial Intelligence and 4.4. Quantum Computing as experts in cybersecurity believe that both will be used to exponentially better cybersecurity.
They also believe we will observe an increase in ransomware threats. Attackers are looking to attempt targeting backup or recovery systems to further prevent possible data recovery. The increase is because ransomware threats directly benefit the attacker financially.

During the pandemic, a lot of systems and data needed to be given remote access causing a spike in another sector of cybersecurity which is secure remote access. Many companies will look to improve their cybersecurity infrastructure especially focusing on secure remote access. A solution for critical tasks is a device that is only capable of performing said task, and has no access to any public network, social media, or email. This physical isolation is one of the best practices of cybersecurity.

All these relate to software engineering as their quality and advancements rely on how much software engineering as a whole develops. The industry needs to produce skilled software engineers for these advancements to take place. It will not be possible to create a single answer to defend against attackers through cybersecurity. But rather multiple procedures, technologies, to minimize the risk of a threat.

4.3. Blockchain Technology

Recently and for some time now we have been hearing a lot about cryptocurrency and how "blockchain" technology is used and how using cryptocurrency could be the future of how we pay for products. A blockchain is essentially a distributed database that is shared among the nodes of a computer network. Blockchains play a crucial role in cryptocurrency systems for maintaining a secure and decentralized record of transactions. When it comes specifically to cryptocurrency, blockchain forms the bedrock for cryptocurrencies like bitcoin. It spreads its operations across a network of computers allowing bitcoin and other cryptocurrencies to operate without the need for a central authority. This not only reduces risks but also eliminates many processing and transaction fees. This relates to the future of software engineering because blockchain is focusing towards decentralization in hopes of there being no one single authority, trustless internist and transparency of transactions as blockchain is being developed to improve the secure and safe way to exchange and intake data. As an example, today we're using passwords and authentication questions to prove our identity online, but blockchain will replace that in the future. Instead of relying on information about you to verify your identity online with questions, digital identification focuses on a unique set of numbers allocated to each user on a blockchain network, which cannot be hacked or modified without access to your private key.

4.4. Quantum Computing

Digital computing has limitations in regards to an important category of calculation called combinatorics, in which the order of data is important to the optimal solution. Complex, iterative calculations can take even the fastest and most advanced computers a long time to process. Due to the potential of quantum computing and quantum mechanics to perform combinatorics and other calculations much faster, many firms around the world are exploring the technology. Possible applications include cybersecurity, bio-engineering, AI, finance, and complex manufacturing. Within the finance industry Goldman Sachs recently announced that quantum technology is approaching mainstream and that they could introduce quantum algorithms to price financial instruments in as soon as five years with Honeywell anticipating that quantum will form a $1 trillion dollar industry in the decades ahead. Now while quantum development is fascinating, the development in the quantum computing world is unlikely to result in many roles for "quantum algorithm developers" in the near term. Though current quantum computers can process algorithms that a traditional computer could not within a human lifetime. Quantum computing is going to open up many opportunities for software engineers and developers while also becoming an effective software tool for development.

5. Conclusion

The world of software engineering is moving fast and is quickly fusing into all kinds of industries. As the field continues to grow, learning to program and improving your skills as a software engineer will be very beneficial and will give high returns in the future. Moreso, being able to identify fast growing industries within software engineering and being able to invest your time into them can also set you up for future success. With the concept of software engineering still being a relatively young field and as it continues to change, we continue to research and find more efficient and applicable ways of implementing technology for the greater good of mankind. There are still many questions to be explored when it comes to new technologies and safety regarding them in some cases. With continuing research and development in the field of technology, we are still to truly see the capabilities of emerging technologies and tools.

6. References

- Robison, K. (2021, March 2). *Here's how quantum computing could transform the future*. Business Insider. Retrieved December 1, 2021, from https://www.businessinsider.com/quantum-computing-investing-computers-enterprise-2021-3#:~:text=Experts%20expect%20quantum%20computing%20to,investing%20heavily%20in%20the%20space.
- Stahl, A. (2021, April 17). *How ai will impact the future of work and life*. Forbes. Retrieved December 1, 2021, from https://www.forbes.com/sites/ashleystahl/2021/03/10/how-ai-will-impact-the-future-of-work-and-life/?sh=ad8166279a30.
- EBR, E. (2021, November 9). *Future of blockchain: How will it revolutionize the world in 2022 & beyond!* The European Business Review. Retrieved December 1, 2021, from https://www.europeanbusinessreview.com/future-of-blockchain-how-will-it-revolutionize-the-world-in-2022-beyond/#:~:text=It's%20been%20predicted%20that%20blockchain,to%20process%20transactions%2024%2F7.
- Stone, M. (2021, November 2). *The future of cybersecurity: What will it look like in 2031?* Security Intelligence. Retrieved December 1, 2021, from https://securityintelligence.com/articles/future-of-cybersecurity-2031/.
- J. Bosch, "Speed, Data, and Ecosystems: The Future of Software Engineering," in IEEE Software, vol. 33, no. 1, pp. 82-88, Jan.-Feb. 2016, doi: 10.1109/MS.2016.14.
- mirchandanimohnish03. (2021, February 2). *The state of software engineering - past, present and future!* GeeksforGeeks. Retrieved December 1, 2021, from https://www.geeksforgeeks.org/the-state-of-software-engineering-past-present-and-future/.
- Richmond, S. (2015, July 8). *Superintelligence: Paths, dangers, strategies. by Nick Bostrom. Oxford University Press, Oxford, 2014, pp. XVI+328. hardcover: $29.95/ £18.99. ISBN: 9780199678112: Philosophy*. Superintelligence: Paths, Dangers, Strategies. Retrieved December 2, 2021, from https://www.cambridge.org/core/journals/philosophy/article/abs/superintelligence-paths-dangers-strategies-by-nick-bostrom-oxford-university-press-oxford-2014-pp-xvi328-hardcover-2995-1899-isbn-9780199678112/FF0ABD55FE60E70D17904E11066C0A4E.
- Zohair, L. M. A. (2018). The Future of Software Engineering by 2050s: Will AI Replace Software Engineers? Retrieved December 3, 2021, from https://journals.sfu.ca/ijitls/index.php/index/index.
- Zvinyatskovsky, S. (2018, August 6). *Transformation to modern software engineering*. Accenture. Retrieved December 4, 2021, from https://www.accenture.com/us-en/blogs/blogs-stas-zvinyatskovsky-modern-software-engineering.
- Honeywell. (n.d.). *The Future of Cybersecurity*. Honeywell. Retrieved December 4, 2021, from https://www.honeywell.com/us/en/news/2020/10/the-future-of-cybersecurity.
- *Artificial Intelligence & Autopilot*. Tesla. (n.d.). Retrieved December 7, 2021, from https://www.tesla.com/AI.

- Hack Reactor. (2019, December 5). *The history of coding and software engineering*. Hack Reactor. Retrieved December 8, 2021, from https://www.hackreactor.com/blog/the-history-of-coding-and-software-engineering.
- *Viking code school*. A Brief History of Software Engineering. (n.d.). Retrieved December 8, 2021, from https://www.vikingcodeschool.com/software-engineering-basics/a-brief-history-of-software-engineering.
- EBR, E. (2021, November 1). *Future of Blockchain: How Will It Revolutionize The World In 2022 & Beyond!* The European Business Review. https://www.europeanbusinessreview.com/future-of-blockchain-how-will-it-revolutionize-the-world-in-2022-beyond/#:~:text=Faster%20Transactions&text=Blockchain%2Dverified%20data%20is%20highly
- Albatross. (n.d.). *History of C++*. cplusplus.com. Retrieved December 8, 2021, from https://www.cplusplus.com/info/history/.
- Harper , D., & Stockman, L. M. (2014). *The history of fortran*. Obliquity. Retrieved December 8, 2021, from https://www.obliquity.com/computer/fortran/history.html.
- Hemmendinger, D. (2008, October 7). *Pascal*. Encyclopædia Britannica. Retrieved December 8, 2021, from https://www.britannica.com/technology/Pascal-computer-language.
- Patel, N. (2021, October 31). *What is seo? (learn how to do it in 5 minutes)*. Neil Patel. Retrieved December 8, 2021, from https://neilpatel.com/what-is-seo/.
- Robison, K. (2021, March 2). *Here's how quantum computing could transform the future*. Business Insider. Retrieved December 8, 2021, from https://www.businessinsider.com/quantum-computing-investing-computers-enterprise-2021-3.
- Scrum.org. (n.d.). *What is Scrum?* Scrum.org. Retrieved December 8, 2021, from https://www.scrum.org/resources/what-is-scrum.
- University of Michigan-Dearborn. (n.d.). *History of FORTRAN Language*. History of Fortran language. Retrieved December 8, 2021, from https://www.livephysics.com/computational-physics/fortran/history-fortran-language/#:~:text=One%20of%20the%20oldest%20programming,of%20math%20formulas%20into%20code.
- *What is Agile Software Development?* Agile Alliance |. (2021, December 2). Retrieved December 8, 2021, from https://www.agilealliance.org/agile101/.