CHAIN REACTIONS

THEMATIC BRIEF ICT

BIG DATA – ARTIFICIAL INTELLIGENCE JULY 2020

ABOUT THEMATIC BRIEFS

CHAIN REACTIONS addresses the challenge for industrial regions to increase regional capacity to absorb new knowledge and turn it into competitiveness edge and business value. There is a strong need to help SMEs to overcome capacity shortages for innovation and integration into transnational value chains.

The project aims at empowering regional ecosystems with the knowledge and tools to help businesses overcome those barriers and generate sustained growth through value chain innovation.

During the project lifetime CHAIN REACTIONS will publish about every 6 month five thematic briefs presenting the rationale behind specific innovation deployment within selected business areas.

As demonstrated in the previous briefs, artificial intelligence developed by software development companies has transformed the whole industry worldwide over the last few years and accelerated digital transformation in various domains like manufacturing, health, bioeconomy or even environment management. But now software enterprises are also adopting artificial intelligence in their own software development processes : artificial intelligence was the result of coding, and now software design, coding, testing is becoming more and more the result of artificial intelligence thus coming to a full circle.

Artificial Intelligence in software development

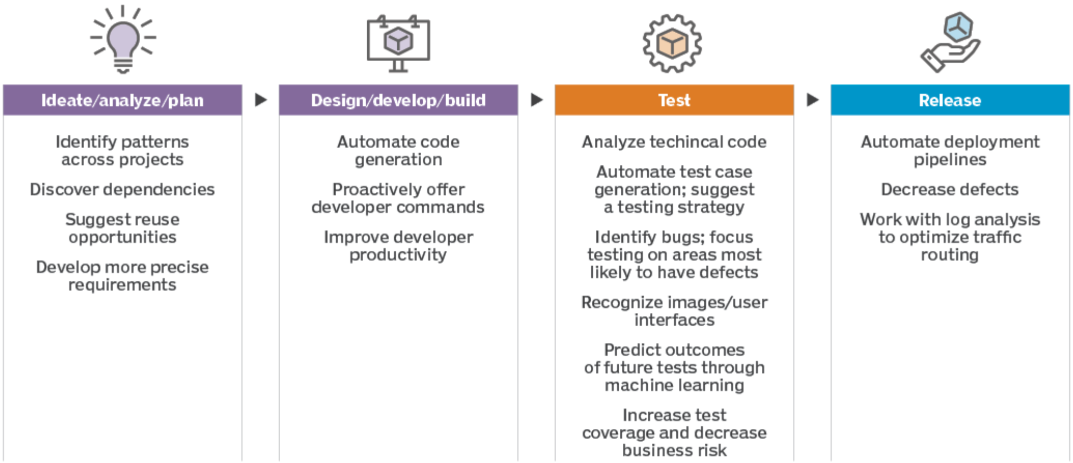
Changing software developement work

AI in Software development is expected to improve development, agile test automation, automation-testing software as well as the way RPA bots functions with the support of the software. Software developers can use AI to write and review code, detect bugs, test software, and even optimize development projects. Not only can companies deploy new software and apps more efficiently, but a whole new generation of developers can learn to code more easily.

According to Deloitte, large and small software vendors have launched dozens of AI-powered software development tools over the last 18 months. [1]. Their analysis also on CB Insights data also find out that startups offering AI-powered software development tools raised US$ 704 million over 12 months ending September 2019.

A 2018 Forrester study found that 37% of companies involved in software development were already using AI-powered coding [2]. Now, with companies such as Tara, DeepCode, Kite, Functionize, and Deep TabNine, as well as many others, providing automated coding services, it's likely that this percentage is higher and growing higher still. This approach is getting more and more popular in software development industry.

What will happen as Artificial Intelligence works its way into your development cycle? How will software development change? How will AI be used to verify ever-enhancing code suites? AI will anyway play a key role in the design, code generation and testing of software. This brief is trying to present the current overall situation and trends.



Ideate/analyze/plan ▶

Identify patterns across projects

Discover dependencies

Suggest reuse opportunities

Develop more precise requirements

Design/develop/build ▶

Automate code generation

Proactively offer developer commands

Improve developer productivity

Test ▶

Analyze techincal code

Automate test case generation; suggest a testing strategy

Identify bugs; focus testing on areas most likely to have defects

Recognize images/user interfaces

Predict outcomes of future tests through machine learning

Increase test coverage and decrease business risk

Release

Automate deployment pipelines

Decrease defects

Work with log analysis to optimize traffic routing

Figure 1 - AI will change Software Development and applications [3]

Requirement Gathering, validation and tracking

Software Requirement Specification (SRS) statements are the formal document through which the customers share their requirements with the development team. SRS is usually written in any of the natural language (NL), convenient to the customer. But, the text written in SRS is observed to be incomplete and ambiguous for the developer in many cases. The requirement gathering, validation and tracking usually requires maximum human intervention and is a major cause of delayed, costly, or failed projects when done poorly.

Therefore, from these incomplete and ambiguous SRS statements and based on AI technologies, several vendors have introduced digital assistants that make an intelligent analysis of the requirements documents, flag ambiguities and inconsistencies, and suggest improvements. For instance, the Infosys Nia solution helps to automate certain processes such as forecasting revenues, forecasting what products need to be built, understanding customer behavior, deeply understanding the content of contracts and legal documents and understanding compliance.

The technique of “Natural Language Processing” (NLP) is there used to make machines understand the user’s requirements in natural language and automatically derive high-level of software models. These tools are trained on widely referenced guidelines for writing high-quality requirements [4] and can detect inaccuracies or other weaknesses—such as incomplete requirements, immeasurable quantification (missing units or tolerances), compound requirements, and escape clauses [5].

Software Design

Planning projects and designing it needs specialized learning and experience to propose a definitive solution. Settling on a correct design for each stage is an error-prone task for designers. Retracts and forward investigating plan forces dynamic changes to the design until the client reaches the desired solution. Automating some complex procedures with AI tools can enable the most capable methods to design the projects.

This approach is illustrated by the AIDA solution: this American website building platform examines various combinations of software design and presents the appropriate customized design as per the client’s needs [6]. Using AIDA, designers can understand the needs as well as the desires of the client and use that knowledge to design the appropriate project according to few parameters. The user is then free to customize the resulting site.

Automatic Code Generation and Analysis

Writing code is still time-consuming and labor-intensive. Experts have therefore approached solutions that writes code before starting development. Intelligence programming assistance can indeed reduce the load and reduce the required keystrokes. Some tools generate a relevance-ranked list of usable code snippets [7]. Other tools work on the same principle as Gmail's Smart Compose, a machine learning–powered feature that suggests words or phrases as a user is composing an email. The vision of a project idea in natural language that a system would understand and convert into executable code is not so far from reality. But this approach of automated code generation is still not good with uncertainties like what target code aims at doing. At the end, collecting these details may take much time like writing code from scratch.

By integrating with AI tools, bugs and error identification is also becoming easier and thus coding is becoming better and enhanced. No need need anymore to struggle with examining the executable files loaded with bugs and errors. Bugs can be found instantly and immediately corrected. Code-review tools already use AI successfully to automatically detect bugs and suggest code changes by understanding the intent of the code and identifying common mistakes and their variants [8]. At Facebook, a bug detection tool predicts defects and suggests remedies that are thus far proving correct 80 percent of the time [9]. The cost of fixing bugs rises considerably further down the software life cycle, as reproducing the defects in a developer's local environment can be complex and business-critical services failure can be costly [10].

Ubisoft, the French video game company, and Mozilla teamed up to develop "Clever-Commit", an AI coding assistant that learns from your code base's bug and regression data to analyze and flag potential new bugs as new code is committed. Ubisoft already uses this tool internally and Mozilla says that it will deploy it to spot bugs in its Firefox code [11]. For Ubisoft, this solution is intended to catch mistakes before developers even commit them in a game's code. Ubisoft claims that this tool can catch six out of 10 bugs accurately and says catching an error before the game hits the shelves can absorb as much as 70 per cent of costs of eliminating bugs [12].

DeepCode is a successful ETH spinoff company from Zürich. The DeepCode solution is a real-time semantic code analysis powered by AI that learns from GitHub project data to give developers AI-powered code review. It covers a broad range of problems, including vulnerabilities such as cross-site scripting and SQL injection, while it also promises to establish the intent behind the code, rather than spotting simple syntax mistakes. The related machine learning (ML) systems are trained using billions of lines of code from public open source projects, which constantly learn and update their knowledge base.

AI in Testing Services

Testing is a vital process that guarantees customer satisfaction within an IT solution. It helps in safeguarding against potential failures. This planed process allows for application assessment and analysis under certain conditions to understand the overall threshold and risks involved in its implementation.

As software testing ensures the quality of the product, this is a crucial phase in software development. Testers need to give feedback and evaluations instantly to the development teams. Releases that happened once a month, now occur on a weekly basis and updates are factored in on almost every alternate day. But testing and repeating tests whenever source code is changed is time-consuming and costly. As the software market demand grows, organizations needed to secure their positions and be ahead of the competitors. To this aim, since the 90‘ automating the testing operations appeared to be a key challenge for software companies.

Here comes Artificial Intelligence to the rescue once again and allow now to test smarter and not harder. Artificial intelligence has the ability to analyze complex data automatically by using smart models and algorithms thus contributing to shorten the software development life cycle and reduce time to market. AI-driven testing will lead the new era of the quality assurance (QA) work in the near future. It will increase the efficiency of the organization to produce more sophisticated software and will create smarter automated testing.

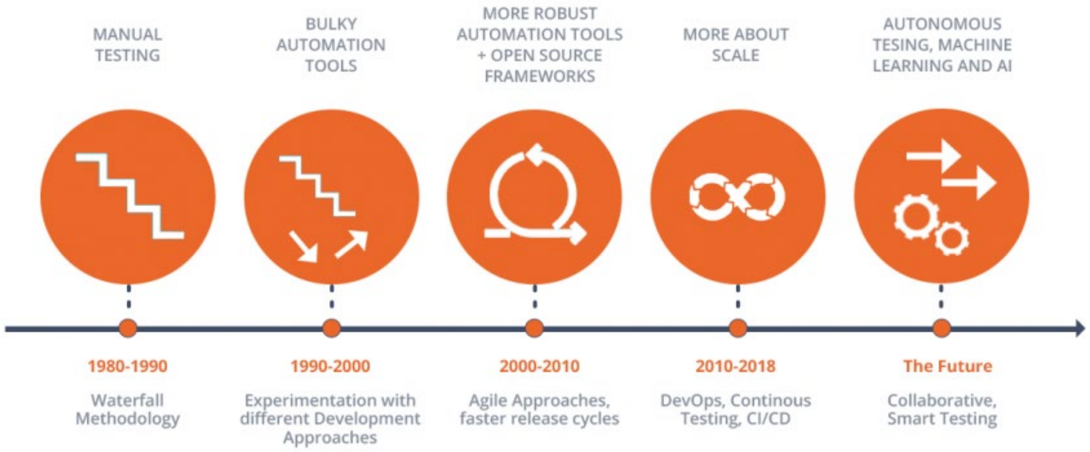

MANUAL TESTING
BULKY AUTOMATION TOOLS
MORE ROBUST AUTOMATION TOOLS + OPEN SOURCE FRAMEWORKS
MORE ABOUT SCALE
AUTONOMOUS TESING, MACHINE LEARNING AND AI
1980-1990
Waterfall Methodology
1990-2000
Experimentation with different Development Approaches
2000-2010
Agile Approaches, faster release cycles
2010-2018
DevOps, Continous Testing, CI/CD
The Future
Collaborative, Smart Testing

Figure 2 - Evolution of testing [13]

Google DeepMind created an AI program that utilizes deep reinforcement learning to play video games by itself, thus, producing quite a lot of test data. In fact, there is already a wide range of tools that employ AI for creating test cases and performs regression testing. These AI tools can automate the testing and further ensure error-free testing (e.g. Appvance, Functionize, and Testim.io).

AI already showed that it can achieve better results in software testing. It will manage and control most of the testing areas and will add great value to the testing outcome and will produce more accurate results in a competitive timeframe. There is no doubt that AI will influence QA and testing industry and will lead this going forward. The smart automation of software testing will improve the quality of the software and will have a major impact on the customers experience through providing a solid defect-free applications and solutions.

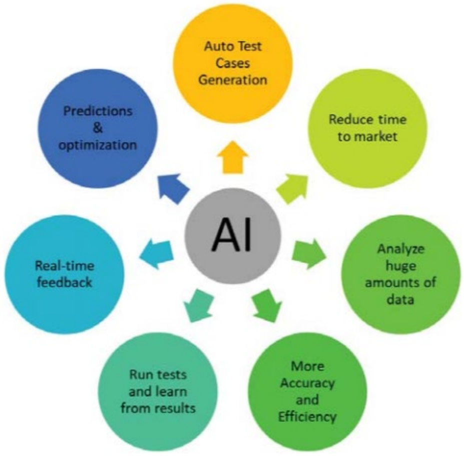

AI
Auto Test Cases Generation
Reduce time to market
Analyze huge amounts of data
More Accuracy and Efficiency
Run tests and learn from results
Real-time feedback
Predictions & optimization

Figure 3 - The AI Software Testing Key Advantages [14]

Deployment Management

Machine learning AI technologies also have some impacts on the efficiency of software deployment. In the software development paradigm, the deployment phase is the stage where developers upgrade the programs or apps to newer versions. In case, the developer fails to execute the process correctly, there will be a high risk in executing the software. A wrong code or deployment failure will not affect the software agency, but a huge loss to the company as well. AI can prevent software developers from such vulnerabilities of upgradation and prevent risks of failed deployment. Developers can examine related to deployment process using Machine Learning algorithms.

Conclusion

Artificial intelligence will in the next future have a significant impact on the design as well as the creation of software. But most software development companies still need to understand the impact of artificial intelligence and the potential benefits it will bring, not only in the way of building software but also in the nature of the software itself.

Beyond requirement analysis, code generation, testing and deployment, AI will offer many other opportunities. Solutions are expected to be developed to estimate software development that involves analyzing historical data from earlier projects of the enterprise to find correlations and statistics. It would there employ predictive analytics as well as business rules to offer accurate estimates of cost, time and effort.

AI will also enable to investigate the data collected from network sensors and software installed on the customer end using machine learning to distinguish irregularities from normal behaviors. Further software development companies adopting AI in their development process can also avoid the delayed warning, false notifications, and alerts.

As a whole, software developers will be able to build better software faster, using AI technologies such as advanced machine learning (ML), deep learning, natural language processing, and business rules. Machine learning solutions are empowered with the ability to learn from past development projects as well as analyzing the performance of existing projects. AI in software development will not only makes development easy but also results in better applications.

Literature

1. AI is helping to make better software, https://www2.deloitte.com/us/en/insights/focus/signals-for-strategists/ai-assisted-software-development.html, January 2020

2. Forrester Research Business Technographics Developer Survey, 2018

3. „How AI will change Software Development and applications“, Forrester Research, 2016

4 Maggie Mae Armstrong, “Engineering breakthrough: IBM introduces Watson AI for RQA,” IBM Internet of Things blog, February 28, 2019

5 IBM, “IBM Engineering Requirements Quality Assistant.” - https://www.ibm.com/products/requirements-quality-assistant - accessed June 20, 2020

6 AIDA solution - https://www.bookmark.com/ai-website-builder

7 Kite, “Kite announces Intelligent Snippets for Python,” September 4, 2019 - https://kite.com/blog/product/announcing-intelligent-snippets-for-python/

8 DeepCode, “Technology” - https://www.deepcode.ai/tech, accessed June 15, 2020

9 Bader et al., “Getafix: How Facebook tools learn to fix bugs automatically.” - https://arxiv.org/pdf/1902.06111.pdf

10 Sanket, “Exponential cost of fixing bugs,” DeepSource, January 29, 2019

11. Frederic Lardinois, “Ubisoft and Mozilla team up to develop Clever-Commit, an AI coding assistant,” TechCrunch, February 12, 2019. https://techcrunch.com/2019/02/12/ubisoft-and-mozilla-team-up-to-develop-clever-commit-an-ai-coding-assistant/

12. BRAD JONES, MARCH 5TH 2018, AI Predicts Coding Mistakes Before Developers Make Them - https://futurism.com/ai-predicts-coding-mistakes-before-developers-make-them

13. QATestLab, „Evolution of testing“ - https://blog.qatestlab.com/2018/11/06/testing-trends-2019, accessed June 15, 2020

14. Hussam Hourani, “The Impact of Artificial Intelligence on Software Testing”, 2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT)