Artificial Intelligence for DevOps and Site Reliability Engineering

Theories, Applications, and Future Directions

Swarup Panda

DeepScience

Artificial Intelligence for DevOps and Site Reliability Engineering: Theories, Applications, and Future Directions

Swarup Panda

SRM Institute of Science and Technology, Kattankulathur, Tamil Nadu, India

DeepScience

Published, marketed, and distributed by:

Deep Science Publishing, 2025
USA | UK | India | Turkey
Reg. No. MH-33-0523625
www.deepscienceresearch.com
editor@deepscienceresearch.com
WhatsApp: +91 7977171947

ISBN: 978-93-7185-970-7

E-ISBN: 978-93-7185-060-5

https://doi.org/10.70593/978-93-7185-060-5

Citation: Panda, S. (2025). *Artificial Intelligence for DevOps and Site Reliability Engineering: Theories, Applications, and Future Directions*. Deep Science Publishing. https://doi.org/10.70593/978-93-7185-060-5

Preface

This book offers an in-depth examination of the transformative impact Artificial Intelligence (AI) and Machine Learning (ML) have on DevOps and Site Reliability Engineering (SRE). It sits at the intersection of the cutting edge in AI and at how actual operations can use smart technology to refine your CI/CD pipeline, tell when incidents are rolling your way, help to automate resolution and improve the eyes on monitoring. Readers will learn complete details on AI-driven observability, finding anomalies, performance tuning, and capacity planning—helping organizations to predict failures, improve up times and accelerate software with a rock rock-solid foundation. With clear and detailed explanations, bolstered by case studies with leaders from the industry, and actionable frameworks to implementation, DevOps engineers, SRE professionals, and IT executives will learn how to effectively operationalize AI within their environments. It also includes critical content on AI ethics, transparency, and governance—a must for today's high-stakes production environments. Readers will walk away fully prepared to use AI to automate the repetitive and time-consuming tasks based on data and to make data-informed decisions that strengthen their infrastructure and deliver operational excellence.

Swarup Panda

Table of Contents

Deep Science Publishing, 2025
https://doi.org/10.70593/978-93-7185-060-5

DeepScience
Open Access Books

Chapter 1: AI-Augmented DevOps: Transforming Software Engineering Through Intelligent Automation and Collaboration

1. Introduction

DevOps ("Development" plus "Operations") is a practice that brings together Development (Dev) and Information Technology Operations (Ops) by promoting better collaboration and communication between the two teams. The objective is to accelerate the system development life cycle and to continue to provide high quality of service in close intersection with business objectives.

These were born at big corporates and are now being democratised and adopted at scale by smaller and midsize businesses. As technology evolves, the next generation of AI capabilities is primed to have a major effect in software development.

2. Background of DevOps

DevOps is a methodology for software development that combines software development and information technology operations to shorten the software development lifecycle and provide continuous delivery with high software quality. It is a key step in the IT evolution that enables companies to respond to the needs of customers and the market rapidly and efficiently. DevOps evolved from the move toward agile engineering. It was created to improve the interaction between the operations team and the development team in order to stabilize and scale software releases while controlling costs and reducing the human element. Instead of development and operations working together only at a specific point in the development lifecycle, they work together throughout the lifecycle,

including design, development, integration, quality assurance, deployment, and product support. In recent years, the integration of machine learning and artificial intelligence with DevOps has led to the emergence of AIOps (Artificial Intelligence for IT Operations).

The AI boost in DevOps capabilities and scalability will open many possibilities in creating applications autonomously by learning from user requirements. The integration of machine learning and deep learning techniques will also help IT operations teams in creating smarter environment monitoring systems and handling IT Operations [1]. Artificial Intelligence with DevOps will reduce the workload, work pressure, and operational errors. An AI-driven process can give organizations the agility to predict challenges early in the cycle, taking proactive steps to remediate those issues. DevOps lifecycle with prediction capabilities helps in faster delivery with improved quality, superior end-user experience, and reduced customer churn.

3. The Role of AI in Software Development

Software development has been a growing field of study over the years and continues to grow. Various areas of software development include programming linguistics, development tools, and software engineering. In recent years, Artificial Intelligence (AI) has been playing a vital role in a man's life. AI refers to systems or machines like human intelligence capability, such as learning, reasoning, and self-correction. Over the years, AI has become one of the most valuable technologies, decreasing human errors while simultaneously assisting with data maintaining, monitoring, and processing. And by reducing errors and automating processes for software development, it will potentially make developers work simpler and handier.

Maintainability is one of the most influential quality features of software implementations in the current era. A thoroughly documented source code along with the usage of meaningful identifiers will definitely boost the stability of software in the long run. Some current testing activities in the software domain, such as test case creation, maintenance of test scripts, test data generation, and testing of new programming languages, are essential but complex and difficult tasks. Artificial Intelligence techniques and methods have been studied for their significant applications in software testing and quality analysis. Natural Language Processing (NLP) is evolving in diverse areas of software engineering,

where most software systems documents are written in free-text in a natural language.

4. AI Technologies Transforming DevOps

Throughout the software development, delivery, and maintenance pipeline, various AI technologies transform the way software is being developed with DevOps. In addition to the pipeline in the defined life cycle, the capabilities introduced by implementing AI Objectives and Key Results (OKRs) address some broader areas as well. Popular AI technologies that are found driving increased adoption of AI in DevOps include Machine Learning (ML), Natural Language Processing (NLP), and Automated Testing.

Machine learning supports DevOps teams by improving efficiency and decision-making, whereby repetitive and time-consuming tasks can be automated, and teams can focus on innovation and high-value activities [1]. The quality of software is improved by identifying vulnerabilities during development and deployment phases. Large volumes of data generated by various tools in the DevOps pipeline can be collated and analysed to predict delays and detect bottlenecks. The computer's ability to learn and perform from patterns in the data enables the prediction process rather than relying on a predetermined algorithm.

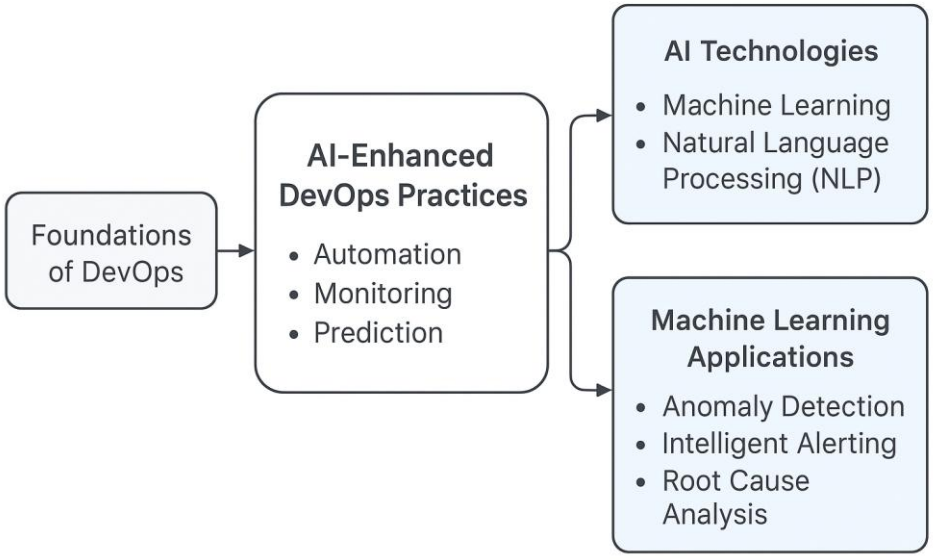

Fig 1. Foundation of DevOps

4.1. Machine Learning Applications

A variety of machine learning applications exist within an AI-empowered DevOps landscapeThe first is automated ticket routing, leveraging their capacity to generalize on labeled data. Freshservice, for instance, has Freddy, an AI - powered agent that assigns incoming support tickets to engineers.. The ability to classify a given dataset with a minimum of human assistance is also helpful in predicting root causes for faults. Platforms such as PagerDuty draw on the patterns discovered in historical operational data to improve the accuracy of their fault predictions. . Finally, ML can also be used to semantically cluster similar data points as in case of anomaly detection and recommendation [1-3]. Jira Software, for instance, analyzes past incidents and advises on the appropriate priority level for a bug.

The inherent flexibility of machine learning gives it the power to support other DevOps functions too. For instance, such models can also help analyze relevant datasets and derive new insights about critical resources and services to better inform decision-making. This is the scientific method that Google Cloud's Operations Suite uses to help your team manage incidents proactively. Similarly, ML-based optimization models can help to maximize streamlining and cost savings. DevOps tool providers like CloudFabrix use these features to optimize infrastructure efficiency..

4.2. Natural Language Processing

NLP is a field of AI that teaches a computer to interact like a human. Speech recognition apps employ NLP to analyze speech patterns with a database, identifying words and phrases. Technologies which already make some use of basic NLP, include Apple's Siri, Amazon's Alexa, Google Assistant and Microsoft's Cortana.

NLP applications to DevOps enhance quality, security, and fault diagnosis of tested software. Developers produce software that finds errors in source code, produces test plans, and identifies code smells. AI can also produce documentation across all software development and testing stages. Organizations Improve construction and organizational processes and give more QA support (They use by NLP). For instance, languages such as Python can build apps combining NLP and AI models, automating tasks like generating user stories for agile processes. The adoption of Alexa and Google Home further demonstrates the efficacy of NLP in automating and expediting development operations.

4.3. Automated Testing Tools

Current development environments, frameworks and programming languages provide automated testing tools designed to reduce the time spent in regression testing during the software development process. They effectively reduce the human effort that otherwise would be spent in developing test cases, resolving bugs and performance testing. However, recently-kilo projects demand a huge amount of testing to be done rapidly in order to save the cost overruns due to late deployment [1-3]. This is achieved by increasing the number of human testers but that also requires more time to be spent on testing processes and higher cost. Intelligent agents have the capability to rapidly and efficiently perform repetitive jobs with good accuracy. Therefore, the testers can be easily replaced by such automated tools for scripted and routine functionalities.

AI-powered testing tools select a subset of test scenarios to perform regression testing. These tools make use of metadata retrieved by traceability from the previous test reports or use a machine-learning engine trained on historic test results to determine the importance of each test case. These test case optimization [81, 70] tools allow only an optimised selection of test cases that are very important and that may be impacted due to new changes in code.

5. Benefits of Integrating AI in DevOps

The incorporation of AI techniques into DevOps has resulted in the emergence of AI-driven DevOps, a practice that employs AI models and tools to support security, business processes, and development. This integration of AI into DevOps not only enhances efficiency and reduces costs but also aligns with the objectives of continuous collaboration and transparency between development and operations teams. The inclusion of AI in these phases further accelerates the software development lifecycle by automating various activities, thereby reducing security risks and enabling timely decision-making. In essence, AI-driven DevOps diminishes human efforts and the probability of errors during development.

Several sectors, such as banking, finance, and the investor community, have already adopted AI-driven DevOps to automate tasks across different phases of the development lifecycle [2,4]. Leveraging the capabilities of AI within a DevOps pipeline enables the acceleration of the deployment process, reduces manual intervention, monitors ongoing deployment phases, and optimizes maintenance activities. During code creation, testing, and integration, AI-driven

DevOps predicts defects and code errors in advance, automates test case generation, facilitates code switching, monitors code plagiarism, and recommends the testing framework most suitable for the particular code under development.

5.1. Enhanced Efficiency

Introducing AI tech into DevOps projects makes the software development process more efficient in a number of ways. This development will enable the AI-based process automation to take away the human slog that engineers are faced with, and increase the speed of their release cycles. Additionally, the utilization of AI enhances the precision in continuous delivery activities, such as infrastructure provisioning, thereby fast tracking the time to deploy new functionality. Finally, even Human-Computer Interaction (HCI) and User Experience (UX) does not escape AI-suppor t: chatbo ts and conversational AIs may recommend to us the web pages to visit and what to do on them without asking, and in general suggest products and services,possible solutions to development-staff, enabling them to use the correct configuration considerably faster.

Furthermore, AI-powered user behaviour tests offer in-depth look at how users actually experience specific features and enhance interface usability. The combination of traditional software telemetry with sensor data has new potential to prevent downtimes of the system, which is certainly a win for both, development and operation peoplepests. Furthermore, these predictions allow the platform to prepare for and prevent potential downtimes. Finally, AI-enhanced code completion enables engineers to finish their tasks more quickly, elevating the productivity of the organization.

5.2. Improved Quality Assurance

An essential part of quality assurance (QA) is testing. QA testing is a key step to verify that software meets business and technical requirements and to minimize defects. Various testing methods exist, such as unit tests, exploratory tests, integration tests, system tests, smoke tests, black-box tests, and white-box tests [5]. Machine learning and artificial intelligence have improved automated testing. Time-consuming and costly testing processes can be carried out quickly without human intervention, thereby reducing the time needed for the overall development cycle.

The AI paradigm is rapidly transforming different stages of development, especially testing. Web application testing, API testing, mobile app testing, and GUI (graphical user interface) testing occupy significant developer time and cost.

Analyzing each web page of an application, setting up test scenarios, and creating test cases are mandatory tasks that consume much developer time and increase project costs. Several open-source tools, such as Appvance, SOASTA, Functionize, ReTest, and Testim, are used to automate the development and management of test cases and profiles with very little development effort or manual intervention.

5.3. Predictive Analytics

Alongside anomaly detection, AI and Machine Learning can be used for predictive analytics that help DevOps teams identify potential risks. For example, services such as Google Cloud's Operations Suite can use anomaly detection algorithms to warn about server downtime. Features like process-mining tools can detect patterns and recommend improvements in deployment, build, and test pipelines.

Several AI features are transforming the software delivery lifecycle. Automated deployment enables continuous or on-demand deployment and rollbacks, preventing resource wastage. Automated code review tools, like DeepCode or LGTM, scan codebases for bugs and vulnerabilities, suggesting corrections. Chatbots used for conversations to execute and monitor tasks. Automated testing creates and runs test cases automatically. Automated surveillance watches application and infrastructure health, noting any anomalies in resource consumption that may affect operation or availability. This can be done in an always-on or on-demand fashion providing visibility at each stage of software pipeline.

Incorporating the above into the practice of the software-testing cycle can contribute to an increase in Test-Driven Development. By crossing BI datasets with test pipelines, devs can predict the business impact of test failures [6-7]. This novel integration additionally results in noise mitigation and the classification of the incidents, significantly improving the overall testing efficiency and the test accuracy.

6. Challenges in AI-Driven DevOps

Despite the huge gain of integrating AI into DevOps, however, there are several non-trivial challenges that have to be looked into. Quality and availability s of data makes it really hard to get to. Poor or insufficient data also may mislead forecasts, and affect the precision and recall. Data Privacy Compliance is

important to ensure that companies are not standing in the way of data protection legislations, thereby complicating the data management even more.

Integrating AI within DevOps isn't something that can be done easily; it requires significant time, investment, and expertise in AI algorithms and DevOps processes. This kind of integration can add time and expense to a project. In the team as well as the AI, a skill shortage impedes the selection of suitable algorithms and the setting of hyperparameters—extending development time and cost. But overseeing the workforce has proven to be a Herculean challenge, which neither AI weaponry nor DevOps can handle on their own, at least with any of the advantages an AI-powered DevOps can offer.

6.1. Data Privacy Concerns

Other than the positive side of AI in DevOps, there are also challenges that need to be resolved, and one of them is data privacy. Over the long term, companies they have to weigh the decision to share data for AI in DevOps against the need to keep data safe via data privacy [7]. Regulations like the EU's General Data Protection Regulation (GDPR) may affect how much AI gets deployed in companies. Not all companies or developers are so forthcoming. Finally, the application of AI introduces questions of ownership and privacy that are themselves growing more complicated.

Adding to this is the skill and knowledge barrier to applying AI. In addition to integrating AI in DevOps, companies should train workers because of a lack of skilled workers. Many of the DevOps-trained experts don't have the skills to include AI in the process. In reality, it is hard to find people with a combination of developer, tester, and data scientist skills. While AI for DevOps holds a lot of potential, organizations should be mindful of its limitations and conce

6.2. Integration Complexity

Also, the use of AI in DevOps is further complicated by the general complexity in the variety of developing organizations. Deployment, testing, and monitoring related activities have many dimensions that need to be maintained over time. In such a rich and diversified infrastructure, using AI is no simple task; and its application requires deep understanding and smooth cooperation among divisions[7-8]. Organizations invest just as much in developing a good skillsets and technical expertise in their employees. Furthermore, information sharing within the organization could be exposed to breaking or leaking, especially monitored by third parties in some of the operations. Furthermore, customers' privacy may be volated if unauthorized parties obtain access to their personal data.

Liabilities such as operational data governance and risk of unknown events is a major deterrent for AI deployment in some companies." Sensitive industries, for example, banking and cybersecurity, does not like that automatic features participate in their internal systems, for a single mistake can lead to devastating results. There are some companies that have extremely strict rules against the use of any automated technologies during process. AI-enabled applications in these fields will need to learn in a supervised manner, where the support automation result instead of the complete automation decision should be the focus, in order to reduce risk in the learning phase.

6.3. Skill Gap in Workforce

The . As a result, the demand for distinct skill sets at all stages of DevOps—from product inception to development, testing, deployment, and release—has shifted, alleviating the pressure on any single stage. Despite this diversification, talent scarcity and the pressing need for rapid product delivery persist. Established organizations usually have the resources to address these challenges, but startups and medium-sized companies might find it difficult to hire a large, skilled workforce quickly.

7. Case Studies of AI in DevOps

The implementation of continuous integration and continuous delivery (CI/CD) enables organizations to respond quickly and easily to potential threats during deployment. One area where artificial intelligence (AI) can fully complement the evolving concept of DevOps is Intelligent Ops, which aims to increase the productivity of operations teams through more intelligent monitoring capabilities. Three aspects of progress exemplify this trend: first, the integration of AI operations (AIOps) across traditional operations disciplines; second, the empowerment of security operations through AI processes, known as SecOps; and third, the combination of customer support functions with AI, referred to as XOps.

Working with a Japanese communications company, IBM implemented an AI-powered DevOps platform that leveraged natural language processing to monitor software deployment and maintenance issues. The platform automated test case generation, prioritized and clustered bugs, predicted delivery timelines, and suggested the required team size for releases. Following an IBM-managed DevOps framework, the company achieved a 30–40% improvement in team efficiency and reduced more than 50% of operations costs. Another case study

involved an AI-enabled test automation strategy that employed image and text recognition to minimize human intervention in defect recognition and bug logging. By analyzing end-user log activities, the organization created an intelligent alert system that expedited issue resolution. This strategy resulted in a 30–40% reduction in overall efforts.

7.1. Company A: AI-Driven Deployment

Large Internet search engines use AIOps to improve the efficiency of the DevOps process. Company A applies artificial intelligence in the deployment process; specifically, it considers deployment-related data in the database to try to predict whether a service will eventually be deployed successfully. Integration of machine learning algorithms throughout the process enables the service management system to predict the probability of a successful deployment, based on the service name, deployment start time, application name, deployment type, and other relevant information. When the deployment failure probability is predicted at 75 or 90%, the DevOps engineer is informed so that appropriate measures can be taken to mitigate failure.

At Company A, the implementation of AIOps is not limited to the prediction of service deployment failure. In the regression test domain, the company uses natural language processing to convert the regression test plan into SQL queries, which greatly speeds up the execution of these SQL queries for test planning. In the area of test case implementation, Company A has integrated a hybrid approach—combining a keyword-driven approach, a state-driven approach, and a data-driven approach—to automate potentially up to 70% of test-case implementation [5-8]. The resulting productivity enhancement has enabled the company to allocate more resources to other, more challenging areas, such as integration testing and intelligent/robustness testing.

7.2. Company B: Automation in Testing

Company B is a multinational technology business operating in many areas besides software development. The company uses AI in the testing of newly developed products. Testing can be very time consuming, but if testing fails it results in further delays and even releases with bugs. Several AI tools automate testing, but the company has developed its own tool. The AI system is fed with thousands of test scenarios that it analyzes for optimal allocation and execution. It predicts which scenarios might fail and when they should be executed. Based on this prediction, the system chooses the most efficient way to execute the tests, mostly the latest ones first, because these have the highest chance to fail. Although the system is not able to create new tests on its own, it contributes significantly to the reduction of manual test efforts in the project.

Company B's second example is about AI in system monitoring. With an increasing number of deployed microservices, system monitoring is becoming a very arduous task. As the current solutions are not able to handle this increased complexity, the company is developing an AI-based monitoring system. It uses sequence-to-sequence models based on Long Short-Term Memory (LSTM) networks to monitor metrics such as CPU utilization or memory consumption of the microservices. This method splits the metrics into a number of segments and predicts those segment values based on the previous ones. By comparing the predicted values to the actual ones, the model detects anomalies. The model requires large amounts of training data to become useful and is still under development.

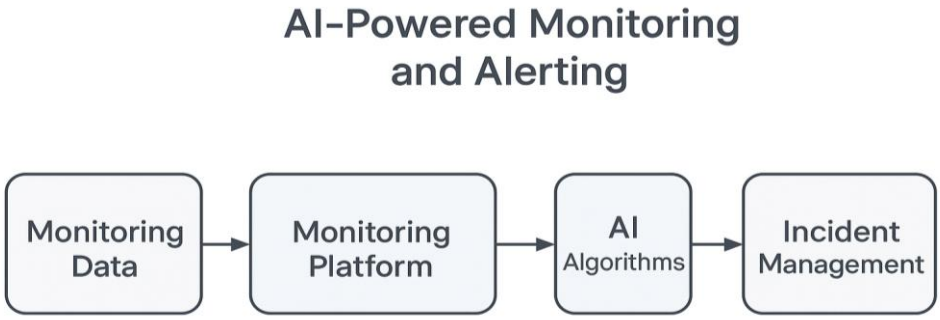

Fig 2. AI in system monitoring

7.3. Company C: Predictive Monitoring

Company C is a large technology corporation that has implemented an integration of AI and DevOps in predictive monitoring. Company C's capabilities enable continuous variation discovery in production systems, to discover new conditions within any type of operational data, such as logs and metrics. The capability also enables forecasting and prediction for any key metrics within the operation. Company C can then detect and create optimization opportunities and provide actionable recommendations. The company notes that users will often have tens of thousands of metrics across the different applications and infrastructure supporting the business, such as database, application, web, network, etc. The site's root-cause service is described as a "directional black box" when the users do not know where to start and they want to determine the sequences of metrics/dimensions that contribute to a symptom. Having tens of thousands of metrics increases the complexity and utility of this task.

Company C states that the implementation of such a system requires high levels of automation and intelligence because manual thresholding would not be

practical given the thousands of metrics across an environment supporting a business [8]. The starting point could be a site level-asset where the perturbation has been identified and then propagate the root cause detection to the underlying infrastructure components of the site. The users then specify the symptoms, the key metrics of the different microservices in each supported area, which can be different from realized changes (variation discovery and forecasting). The AI component should also (semi) continuously compare how these key metrics are behaving. When the users have some insights identified, that can be looped back to the AI Engine to enhance the performance and preventive safety measures of the system.

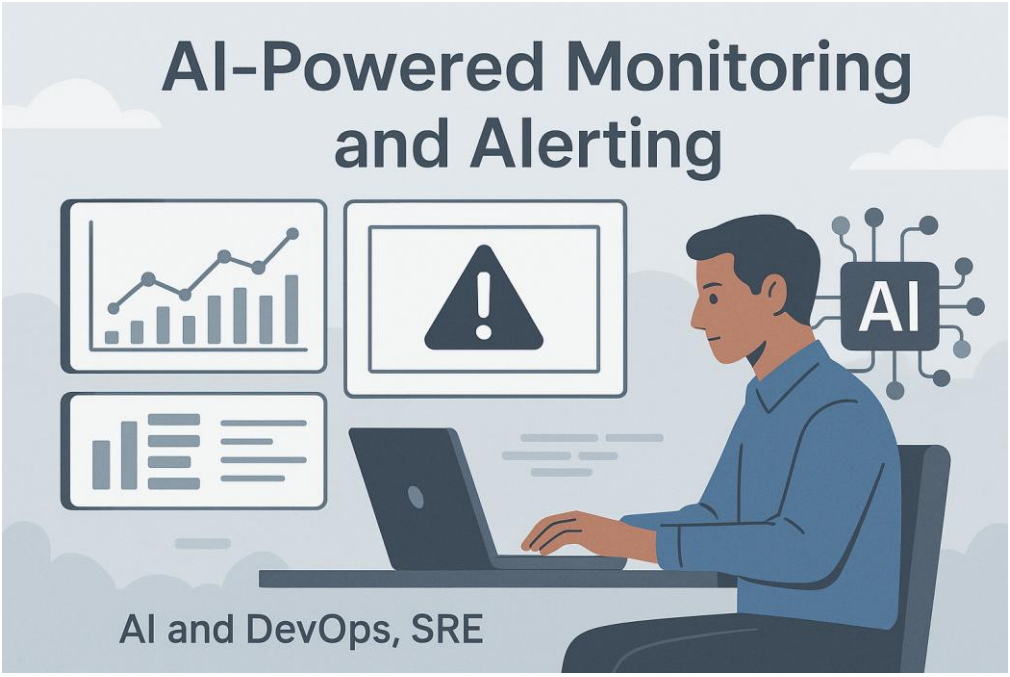

Fig3. AI-Powdered Monitoring and Alerting

8. Future Trends in AI and DevOps

Artificial Intelligence (AI) is exploring part of all business areas, and DevOps is no exception. DevOps can leverage AI as enabling technology with the aim of making the software development lifecycle smarter and more proactive. Combining the cutting-edge technologies of DevOps and AI delivers great potential for automation, timely business agility, and services that can think closely like the human brain. As the principles of DevOps are in continuous

evolution, the integrated use of AI also opens new horizons for trends and benefits. Development and operations teams can integrate AI technology to support their daily work at any level of the software lifecycle. Successful exploitations of AI technology in the software lifecycle can make it smarter by reducing manual work and enabling supporting teams to take appropriate business decision making.

8.1. Increased Automation

The pursuit of continuous delivery, which culminates in automating the entire software implementation process, is the goal of DevOps. Artificial intelligence has already been integrated into every part of the DevOps life cycle, revolutionizing the concept of continuous delivery. In the present day, the Software Development Life Cycle has entered a new phase characterized by AI-assisted control, monitoring, and automatic handling of the deployment pipeline.

Much attention in both research and practice has turned to automating the remaining activities. The capabilities of AI technologies like machine learning and intelligent tools have resulted in notable advancements in deployment, testing, resource pooling, and AI-assisted design in recent times. Companies such as Microsoft, Facebook, and Netflix are increasingly reliant on AI to automate various components of the DevOps life cycle. Consequently, the evolution known as AIOps—DevOps enhanced by AI—emerges as a natural and inevitable development.

8.2. AI-Enhanced Collaboration Tools

A practical illustration of the influence of AI is provided by the integration of OpenAI's Chat GPT. As a conversational AI capable of translating natural-language requests into inputs for other automata, Chat GPT can be linked to a conversational developer assistant. Such an assistant is able to translate natural-language developer requests into a series of actions, invoking then the involved automata. For example, a developer can request, "Please run vulnscan against RAMS4U.com" or "Please deploy backend to live server". The NLU component analyses the requests so that a natural-language-understanding engine understands the utterances and can translate them into a specific planning request, of the form deploy (backend, live_server). The deploy request is passed on to a planning engine that can resolve the request, formulate the appropriate substeps, and assign each substep to the corresponding automaton responsible for executing it [6,9].

The output planning might consider actions such as running the Maven command `mvn clean deploy -Dmaven.test.skip=true -P liveServer`, deploying the backend

to the live server; and performing automatic connectivity and server tests to verify the service's accessibility and functionality. The automation sequencer coordinates these subprocesses until the initial request has been fully carried out. Once the plan for a specific request has been generated, the monitoring dashboard can be used for an overview of all executing, completed, and failed requests.

8.3. Real-Time Analytics

Real-time analytics is integral to the monitoring phase in the DevOps pipeline. It allows businesses to examine operational processes and customer behaviors as they occur. Several groups are investigating continuous analytics for identifying bottlenecks in business operations [10]. For example, path anomaly detection applies analytics to detect fraudulent activity on retailer sites and hospitals.

The practice of machine learning for operational processes and business models improves the quality of services delivered while reducing operational costs. In DevOps, an operation model considers the operational processes performed by resources. Machine learning for these processes aids in pinpointing inefficiencies, scaling of operations, and identifying rules for smooth execution. Business models relate to strategies used by the business with respect to customer behaviors and turnaround time. The usefulness of these models is enhanced by forecasting the behavior of customers, both positive and negative, to ensure SLA adherence.

9. Ethical Considerations in AI Implementation

The essence of DevOps is collaboration. . Its ample integration of the wide range of development, test, deploy and maintain tasks written on its flag, breaks down the barriers between distinct fields in the company that each serves, and it imparts the feeling of one-team-one-vision. An organization is no longer a sum of independent elements, but a single integrated whole where we now have a shared responsibility and shared ownership!

The subtleties of AI are there, at each step, on top of the DevOps lifecycle. In these prescribed positions, AI can assist and augment different roles of development, maintenance and test. There are also many companies that have begun to use AI as part of their DevOps tooling. Organization capability is inevitably elevated when AI helps test complex features and perform computationally intensive testing tasks we cannot always have the luxury of

testing. AI enabled DevOps: The team can concentrate on mission-critical work that is not mechanical and mundane, thus improving morale and motivation.

9.1. Bias in AI Algorithms

The integration of AI unlocks new perspectives for the DevOps enactment, extended the level of automation and optimization provided by the Development and Operations approaches [10-12]. AI-based automation allows companies to allocate their resources to more actionable items by assigning routine work to intelligent machines that are capable of doing it accurately and self-sufficiently. Another benefit is AI's predictive capabilities that can predict future conditions and stop diseases before they materialize.

Machine Learning facilitates the design of Smart Orchestration platforms that assist in deployment, provisioning, and continuous integration and deployment processes. Natural Language Processing enables the creation of Intelligent Chatbots that support the DevOps workforce. Automated software testing is also evolving with AI through smart tools that generate test scripts.

9.2. Transparency and Accountability

Transparency and accountability constitute additional ethical aspects in AI. While AI applications themselves seem to make decisions or predictions autonomously, human involvement is always required to determine the purpose, goal, and setting of their use. This human involvement should be easily discernible, and the decisions and predictions should be explainable. In highly regulated contexts, AI can often be used only if an explanation of those decisions and predictions is possible. Currently, various norm-setting initiatives have defined AI transparency requirements, such as the capability to provide explanations for how the system functions, to explain the processes and methods used for decision-making, to justify choices and decisions by reflecting input data and the decision-making process, or to document and record a system’s decisions during operation.

Another key question concerns who should be accountable for incorrect predictions or decisions made by an AI system. This question is unresolved, but it seems unlikely that AI systems will be held accountable for their decisions in the near future; rather, responsibility for their actions will likely remain with humans. Not only the individuals or organizations that use the AI applications must often comply with accountability requirements; sometimes those who

provide the AI system must also accept responsibility, particularly in relation to the quality of the training data and for the explainability of the system

10. Best Practices for AI Integration in DevOps

It is imperative to exercise care to capitalize on the advantages of AI integration in DevOps while sustaining explicit human monitoring and control of crucial processes and outcomes. Intelligent tools powered by AI can assist in the development of sophisticated The application of AI, ML, NLP in DevOps delivers a number of benefits to the stakeholders involved, such as increased efficiency and cost-effectiveness; improved quality assurance and lower number of defects; automation software testing; better user experience; and advanced prediction and analytics. Being able to address challenges inherent to AI adoption—like the complexity of AI methods themselves, higher levels of automation, integration burdens, lingering data privacy and security issues, as well as regulatory and compliance demands—can drive larger-scale progress across people, processes and technology.

By following industry best practices, organizations can reduce their exposure to the relevant risks and capitalize on the capabilities that AI brings to DevOps. These best practices cover the following aspects—from securing data and process to evaluating formal challenges in deployment. AI/ML should make the consumption and discovery of insights more efficient, not just automate menial tasks. AI/ML has strong capabilities in dealing with large and diverse datasets and therefore, its downstream application across the enterprise life-cycle can be game-changing. Yet the extensive automation provided by AI and other such technologies does not alleviate the need for a continuous vigilance, careful stewardship, and leadership by organizations, business analysts, ethical auditors, and human overseers, in this rapidly changing AI/ML relevant appliance ecosystem.

10.1. Continuous Learning and Improvement

Continuous Learning and Improvement

Struggle for perfection is a never-ending journey. Currently being directed and informed from the past project completions toward that end state of completion and improvement. Continuous learning strategies are, in DevOps processes, used to maintain knowledge and also to integrate these insights into daily work.

Machine learning (ML) approaches are the enabling technology for active analysis in the scope of continual learning scenarios. ML provides a statistical yet repeatable method for uncovering hidden patterns within collected data. Also known as predictive analytics, ML makes predictions about future events as more data becomes available. Including ML as part of DevOps workflows enables early mitigation of anticipated problems [7,13-16]. ML can also help to triage problems during incidents in real time with proposed remediation activities. Within a continuous learning framework, these techniques close the loop back toward the plan phase.

In DevOps jargon, continuous learning is referred to as "Learning Loop". It is one of the essential components a DevOps architecture should address. DevOps, lean production, and agile software development share many principles. Therefore, the PDCA cycle has already been considered foundational for DevOps. Deming's "Shewhart cycle" (named after Walter A. Shewhart, a pioneer in statistical process control) supports the Lean Manufacturing and Continuous Delivery methodologies, among others. Properly accounting for the feedback and learning loops of the Deming cycle is crucial in a modern DevOps architecture.

10.2. Collaboration Between Teams

Collaboration is recognized as a key factor supporting IT\textquotesingle{}s agility and the ability to deliver new capabilities, as highlighted in the Agile Manifesto. In the context of the Agile and DevOps confluence, Alan Shimel pointed out in his 2016 article \\textquotedblfar more important than technology (is) putting in place a culture that ensures constant collaboration between Development, Operations, Business and beyond." However, a survey revealed that less than half of respondents (45\%) completely agree that security teams have good relations with developers, and even fewer (26\%) believe security teams have good relations with IT operations. Gaining the buy-in of the Chief Information Security Officer is also recognized as imperative for DevOps success.

Early implementations of DevSecOps have integrated security into Continuous Integration and Continuous Delivery (CI/CD) pipelines performed in a largely manual manner through human oversight. As CI/CD embraces more automation through machine learning and artificial intelligence, these processes are likely to become more effective, enabling proactive vulnerability detection and prevention.

11. Tools and Frameworks Supporting AI in DevOps

As developers evolve DevOps to New DevOps, they rely on artificial intelligence to help develop software and test it. As these AI technologies evolve, AI will help in many DevOps stages, including coding, building, packaging, testing, deployment, and monitoring. Machine learning models support automating builds, regression testing, performance testing, packaging, and deployment. Natural language processing is extensively employed for requirements gathering, test-case generation, and deployment instructions, while automated testing tools incorporate AI technologies to enhance test-case coverage.

AI increments in New DevOps promote efficient software delivery. Maintainers employ machine learning algorithms to rationalize the number of test cases, propose the sequence for regression testing, and report anomalies in application user interfaces. They forecast the growth of an application to plan infrastructure, which enhances planning and reasoning in several DevOps stages. In feature testing, for instance, AI aids in pinpointing feature faults and recommending suitable testing tools for the software.

11.1. Popular AI Tools

Machine learning is one of the most popular artificial intelligence technologies in software testing. Leveraging input records, data, outcome, and detailed logs, these algorithms optimize test priorities, estimate optimal test cases, and aid error discovery during testing phases in DevOps. These tools also run tests and formulate recommendations, thereby minimizing human effort. Recent proposals include deep learning features that advance test automation and maximize test coverage across different referred aspects, enabling industry teams to choose appropriate data-driven tests for API testing. Natural language processing (NLP) supports the construction of comprehensive test plans and cases by analyzing unstructured data; it forms an industry-wide basis for automating the software testing life cycle. Domain-based approaches utilize NLP to define testing process models, while classification algorithms segment feature information into recommendation clusters, enhancing the exploration of new ideas within DevOps. Automated testing tools play an important role in DevOps by identifying unaddressed test scenarios and generating suitable test cases tailored to specific testing conditions.

Artificial intelligence and machine learning benefit DevOps by accelerating the delivery of quality code and maintaining strong control over integrated

applications. AI-enhanced continuous development and deployment help developers manage several mundane tasks and concentrate on the real aspects of developing new applications. Industry-leading organizations such as Google Azure and AWS utilize AI/ML-driven analytics to analyze production activities, examine root causes, and predict outages. Various tools—Cover Your Code, AutonomIQ, Espresso, Selenium, Appvance, Testim, Apollo, Mabl, etc.—enable the automation of continuous testing through AI, further augmenting the effectiveness of DevOps methods.

11.2. Frameworks for Integration

Artificial Intelligence (AI) can be integrated into DevOps in several different ways, including continuous integration and continuous delivery (CI/CD), automated testing, log analysis and pattern deduction, continuous monitoring and serviceability, anomaly/event prediction, and anomaly detection and mitigation [2,17-19]. The AI-enhanced implementations enable the DevOps team to deliver with increased speed and accuracy. Several traditional DevOps challenges, such as race conditions, build serviceability, release quality and schedule, infrastructure provisioning, risk assessment, and security mitigation, can be resolved using AI.

Machine learning (ML), a subfield of AI, is one of the transformative technologies that are now sculpting many aspects of the DevOps paradigm. The adoption of ML algorithms and models facilitates the use of data generated from various software development activities to reduce human involvement.

12. Impact on Organizational Culture

Integrating Artificial Intelligence into DevOps practices is much more than automating processes; it transforms the positions and capacities of DevOps personnel and generally influences the organizational culture. The major consequence is the evolution of the operational skill set within IT organizations. Changes affect all layers of the culture and lead to the redefinition of responsibilities within DevOps teams. The integration of AI shifts focus from routine tasks to more creative, design-led, and service-focused assignments.

A DevOps environment culture empowered with AI technologies becomes more collaborative and eliminates routine, repetitive actions. AI also influences the communication strategy used in that organization. Large organizations that leverage AI capabilities become hierarchical and tend to empower management with increased decision-making powers—an approach that maximizes the utility

of AI-based speeds of insight. Smaller organizations deliver the responsibility, governance, control, and tactical decision-making powers of AI to individual teams and programmers.

12.1. Shift in Mindset

Technology continues advancing, and with it the art of DevOps as a whole. Artificial Intelligence and Machine Learning, as some of the biggest innovations of the last decades, can have an enormous impact on the way DevOps works and, in some way, can contribute to a certain "second coming" of DevOps. In fact, the technology, the procedures, and the companies involved in DevOps create vast amounts of data that, when combined with Artificial Intelligence techniques, can establish new ways of working that enhance performance and create more comfortable working environments.

Machine Learning is a collection of artificial intelligence techniques that allow machines to learn from historical information. In the context of DevOps, ML algorithms are applying to analyze historical project data. Additionally, natural language processing enhances collaboration by enabling intuitive communication between technical and non-technical team members. These technologies, alongside AI-powered response generation, automate incident handling, thereby boosting efficiency. Automated testing tools streamline quality assurance, reducing manual effort and errors. Collectively, AI integration delivers heightened efficiency, improved quality, deeper customer insights, and predictive capabilities that equip DevOps teams for future challenges.

12.2. Fostering Innovation

Research and development innovation is a key for the success of any organization. Innovation is the key to anything that is fresh and determines the future. As innovation is the main aspect of future organizations, companies should always be ready for research and development innovation. Companies are focusing on automation for their research and development for innovation, but it requires experience in the respective field [3-20]. App development assistants play an important role in mobilizing researchers and developers with different demands. Using artificial intelligence, they provide solutions in different fields for an easy-to-use interface.

The app development assistant concentrates on three major functions: scheduling, estimation, and progress tracking. This provides the development project schedules at different stages of the project, such as initial requirements, analysis, design, and build and test, in a cleanly formatted table. Project estimates for new applications and new features in existing applications are generated

based on app characteristics and area of use. The progress of the development of applications such as completed, in progress, pending, and planned are visually available for each stage of the project.

13. Conclusion

DevOps practices have always been focused on collaboration and integration, focusing on the interaction and close cooperation of the software and infrastructure teams. The application of Data Science methods in the software development and operation areas supporting these teams can be an interesting concept to make work more automated and errors less frequent and risky. For example, Machine Learning classifiers can be used to automatically detect highly risky and error-prone changes that should be scaled up for further verification and testing in the CI/CD pipeline. AI is expected to have a huge impact on the way software is developed and, therefore, these ideas should be introduced to improve DevOps practices. However, the process of obtaining relevant data for AI software is very difficult, and the involvement of many different roles working and cooperating in a DevOps environment is a critical and challenging point. AI should, in fact, be integrated with DevOps in order to obtain AIR (Artificial Intelligence for IT operations), which in turn should enhance the DevOps procedures and traditional tools.

Some of the ideas that would allow AI to be integrated into DevOps for obtaining AIR include automatic change classification, automatic test implementation, and smart forecasting about system security, responsiveness, and stability. These concepts could be implemented to improve the reliability and robustness of the systems developed and managed by DevOps teams. The increasing influence of AI should therefore be supported by these considerations in order to allow AI to become more and more integrated with DevOps.

References

[1] Ugwueze VU, Chukwunweike JN. Continuous integration and deployment strategies for streamlined DevOps in software engineering and application delivery. Int J Comput Appl Technol Res. 2024;14(1):1-24.

[2] Kim G, Humble J, Debois P, Willis J, Forsgren N. The DevOps handbook: How to create world-class agility, reliability, & security in technology organizations. It Revolution; 2021 Nov 30.

[3] Myllynen T, Kamau E, Mustapha SD, Babatunde GO, Collins A. Review of advances in AI-powered monitoring and diagnostics for CI/CD pipelines. International Journal of Multidisciplinary Research and Growth Evaluation. 2024 Jan;5(1):1119-30.
[4] Panda SP. The Evolution and Defense Against Social Engineering and Phishing Attacks. International Journal of Science and Research (IJSR). 2025 Jan 1.
[5] Lo D. Trustworthy and synergistic artificial intelligence for software engineering: Vision and roadmaps. In2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE) 2023 May 14 (pp. 69-85). IEEE.
[6] Shivadekar S, Halem M, Yeah Y, Vibhute S. Edge AI cosmos blockchain distributed network for precise ablh detection. Multimedia tools and applications. 2024 Aug;83(27):69083-109.
[7] De Silva D, Alahakoon D. An artificial intelligence life cycle: From conception to production. Patterns. 2022 Jun 10;3(6).
[8] Battina DS. DevOps, a new approach to cloud development & testing. International Journal of Emerging Technologies and Innovative Research (www. jetir. org), ISSN. 2020 Aug 8:2349-5162.
[9] Maheshwari A. Digital transformation: Building intelligent enterprises. John Wiley & Sons; 2019 Sep 11.
[10] Subramanya R, Sierla S, Vyatkin V. From DevOps to MLOps: Overview and application to electricity market forecasting. Applied Sciences. 2022 Sep 30;12(19):9851.
[11] Dang Y, Lin Q, Huang P. Aiops: real-world challenges and research innovations. In2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion) 2019 May 25 (pp. 4-5). IEEE.
[12] Panda SP. Securing 5G Critical Interfaces: A Zero Trust Approach for Next-Generation Network Resilience. In2025 12th International Conference on Information Technology (ICIT) 2025 May 27 (pp. 141-146). IEEE.
[13] Mohapatra PS. Artificial Intelligence and Machine Learning for Test Engineers: Concepts in Software Quality Assurance. Intelligent Assurance: Artificial Intelligence-Powered Software Testing in the Modern Development Lifecycle. 2025 Jul 27:17.
[14] Erich FM, Amrit C, Daneva M. A qualitative study of DevOps usage in practice. Journal of software: Evolution and Process. 2017 Jun;29(6):e1885.
[15] Qumer Gill A, Loumish A, Riyat I, Han S. DevOps for information management systems. VINE Journal of Information and Knowledge Management Systems. 2018 Feb 12;48(1):122-39.
[16] Panda S. Scalable Artificial Intelligence Systems: Cloud-Native, Edge-AI, MLOps, and Governance for Real-World Deployment. Deep Science Publishing; 2025 Jul 28.
[17] Almeida F, Simões J, Lopes S. Exploring the benefits of combining devops and agile. Future Internet. 2022 Feb 19;14(2):63.
[18] Battina DS. An intelligent devops platform research and design based on machine learning. training. 2019 Mar;6(3).
[19] Battina DS. Ai-augmented automation for devops, a model-based framework for continuous development in cyber-physical systems. International Journal of Creative Research Thoughts (IJCRT), ISSN. 2016 Sep 3:2320-882.
[20] Enemosah A. Enhancing DevOps efficiency through AI-driven predictive models for continuous integration and deployment pipelines. International Journal of Research Publication and Reviews. 2025 Jan;6(1):871-87.

Deep Science Publishing, 2025
https://doi.org/10.70593/978-93-7185-060-5

DeepScience
Open Access Books

Chapter 2: Advancing CI/CD Pipelines with Machine Learning: A Study on Intelligent Automation and Optimization

1. Introduction

The use of CI/CD pipelines is becoming increasingly prevalent in the realm of software development and deployment. CI/CD (Continuous Integration/Continuous Delivery) contributes to the shortening of development cycles and the acceleration of software build, test, and release processes. Extensive prior research has examined the enhancement of CI/CD pipelines via machine learning (ML). The advent of machine learning stems from the development of artificial intelligence (AI)—a discipline dedicated to enabling computers to mimic human cognitive skills. Machine learning, as a subset of AI, empowers systems to learn from data without explicit programmingAt present, the growth in the volume of available data, the progress in algorithms and computing resources has stimulated the proliferation of machine learning which affects many fields of science.

CI/CD is what large companies use to make their software delivery faster and products more competitive. However, running a CI/CD pipeline is a significant overhead so it has the potential to learn what continuous deployment of an application looks like. While CI/CD is mainly considered a purely technical issue, its practices are strongly driven by factors anchored in organization and management, which current research does (not) address sufficiently. As a result, recent studies advocate that using machine learning in software engineering is more than a mere technical analysis of code and pipelines, but an investigation of organizational and management matters.

2. Understanding CI/CD Pipelines

CI/CD pipelines are the bread and butter of the modern DevOps lifecycle. They allow companies to create, test, release, and then monitor software system changes fast. Since software development is performed at an accelerated pace, the business can then meet client demand at increased rates. Thus, the CI/CD pipeline is of paramount importance. The fundamental steps are committed, build, test, deploy, and monitor. Secondly, from the technological point of view, there are a total of eight stages between commit and the last monitoring steps CI/CD pipeline, i.e. a code repository in which we can commit code, a build server, an artifact repository, a container repository, an environment manager, a job scheduler, a monitoring server and a dashboard.

Artificial Intelligence (AI) - or machine learning (ML) - has started to seep into concepts on improving efficiency and quality of modern software development. In terms of the CI/CD pipeline, ML methods can be applied to extract new knowledge from logs, metrics, notifications, and so on [1-2]. This enables the optimization and enhancement of all aspects of the CI/CD pipeline. Many of these can be mapped to ML models analysing the results of a software build helping to optimize the development and release process. In addition, also the implementation of predictive analytics that uses historical datasets as a training set can act as an early warning system and forecast future metrics to even solve a problem before it emerges.

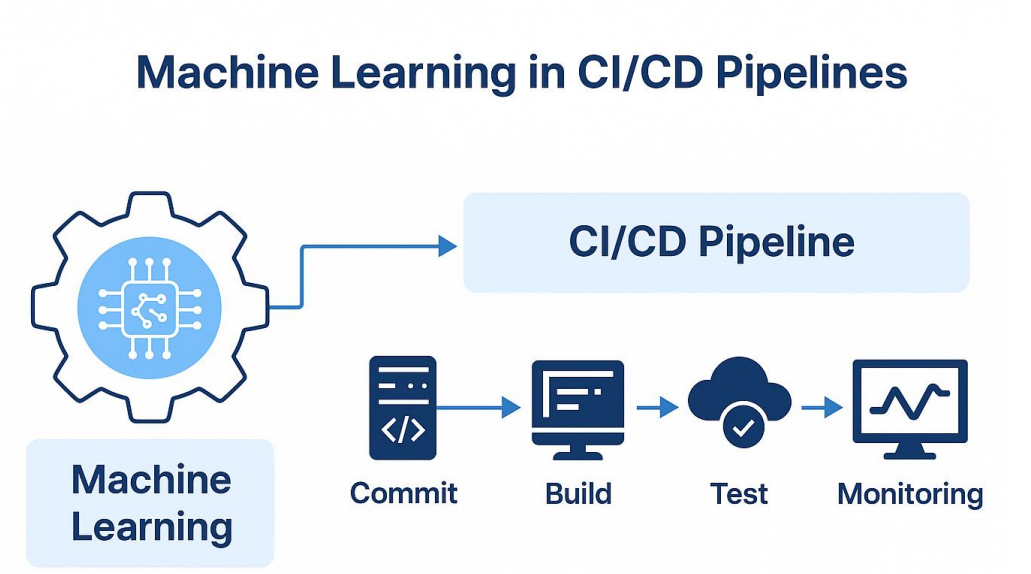

Fig 1. Machine learning (ML) in CI/CD Pipelines

2.1. Definition of CI/CD

Agile software development relies heavily on continuous integration and delivery. The acceleration in the delivery of a product demanded by the market has never halted and has always pushed the software industry to move towards approaches that enable better products and faster delivery. The most recent of these tendencies is continuous integration and delivery (CI/CD), which automates processes and reduces iterative risks. Specifically, continuous integration is a procedure about integrating software every time a change is generated by the development team.

Continuous delivery follows the same pattern, with the difference that once a change is generated, it proceeds with the deployment. Criticism of the inherent brittleness of automated infrastructure has led to an interest in utilizing machine learning techniques to reduce the risks of CI/CD pipelines. Despite the complementary nature of the two areas, the application of ML techniques to support CI/CD pipelines is still under-investigated. These approaches serve as connectors between the different teams involved in the software lifecycle, whether development, operations, infrastructure, or support. The automation of processes can be implemented using specific tools—universal or customized—and a specific pipeline.

2.2. Importance of CI/CD in Software Development

CI/CD pipelines have attracted lots of attention in recent years due to their inherent ability to accelerate deliveries by minimizing the manual effort needed for developing, testing, integrating and deployment of new features. Machine learning is aiming to make software delivery processes even more efficient. Using predictive analytics and pattern recognition, machine learning solution can track potential flaws or incongruities throughout the software development lifecycle - from code repositories, through testing, to production. This automated smarts cut down chances of software getting crashed and improve overall quality. The benefits of using machine learning in CI/CD And the advantages of using machine learning in CI/CD include accelerating delivery velocity, quickening defect detection and prediction, minimizing manual work and improving collaboration at every stage of the pipeline. However, introduction of machine learning to the CI/CD pipeline faces challenges considering the explosion of pipeline data volume and how to select the appropriate machine learning technology for phases.

Further, trial and error approaches in a production environment are expensive and harmful to software delivery quality. The machine learning algorithms can select the proper variants for the application in development, perform automatic continuous testing and monitoring of applying the variants to the app and managing the continuous integration, and deployment of features of the software [1-2]. These algorithms can also efficiently conduct continuous monitoring focused on user safety. By harnessing these capabilities across various phases of the CI/CD pipeline, machine learning actively supports agile teams in delivering faster and higher-quality software.

2.3. Key Components of CI/CD Pipelines

The CI/CD pipeline comprises certain key technical components and automated processes that enable rapid build, test, and deployment of source code. These components address various developer concerns. Source Control provides version control over the source code base and enables developers to work in parallel without code conflicts. Build Automation generates a build for the application from the checked-in code and notifies developers of any build failures. Unit Testing automatically tests the application unit-wise to check for code-level defects.

Artifact Repository acts as a repository manager and stores identified build artifacts that can be deployed on a Kubernetes cluster. Automated Deployment operates under the control of the CI/CD pipeline and deploys the build. Automated Smoke Testing performs smoke testing of the build and informs if the build is ready for further deployment. Automated Performance Testing conducts performance testing and generates a report on the application's performance. Automated Browser Testing executes automated browser tests on the application and generates test reports. Monitoring tracks live application metrics and alerts the concerned teams when any anomaly is detected.

3. Overview of Machine Learning

Machine learning, a key aspect of AI, enables software applications to learn from data and become smarter in order to solve problems and provide services. ML algorithms adjust data models iteratively and line by line at execution time and can be used without additional reprogramming, which makes them a powerful tool for many industries, including recommendation systems, fraud detection, natural language processing, image recognition, facial detection and voice recognition etc. More generally, machine learning can be classified into

supervised, unsupervised, reinforcement, and deep learning. Supervised learning Leveraging labelled past data, learns functions that minimize the generalization error on future observations [2]. Unsupervised learning deals with unlabeled data and tries to find underlying relationships or patterns through techniques such as clustering and dimensionality reduction. Reinforcement learning is a training method where agents interact with their environment to learn objectives by receiving rewards or penalties. DL achieves human brain neuron interconnections in artificial neural networks with strong power for many applications.

Machine Learning models as part of CI/CD pipelines is an interesting concept. When used, ML can provide CI/CD pipelines with valuable input like risk profiles for new features, deployment tactics, anomaly detection, and improved continuous monitoring. This integration, commonly referred to as the CI/CD pipeline of AI based software development is expected to reduce18 human intervention, facilitate, and support all the steps and choices in software development with AI. Nevertheless, the blend of ML into CI/CD introduces significant challenges, demanding meticulous selection of techniques and algorithms to harness the full potential of machine learning enhanced CI/CD pipelines.

3.1. Definition of Machine Learning

Machine learning (ML) is a subset of artificial intelligence and focuses on the development of algorithms and techniques which can enable software applications to learn from and make predictions, classifications, or decisions based on data. Contrary to traditional programming paradigms, where set rules directly specify certain operations, machine-learning models rely on the history of the training data to adjust their internal parameters.

The need for machine learning in system, software or process is a problem with a history data that could be used for learning from as well as a clear goal of reducing manual work or the decision that require human intervention. Machine learning falls into three broad categories: supervised learning, unsupervised learning, and reinforcement learning. These types differ mostly by the degree of guidance with regard to the desired outputs given to the algorithms during training. Supervised learning needs the most supervised (labelled) training data, whereas reinforcement learning doesn't need feedback until decisions or classifications are being made.

3.2. Types of Machine Learning

Machine learning as a part of artificial intelligence, involves learning from data, instead of relying on rule-based programming. It allows software to be more accurate in predicting results without having been explicitly programmed for it. In machine learning, computers can be trained to learn to do something based on a presented sample of data, and then to be able to predict or decide a new previously unseen data.

The field is generally grouped into four catagories: supervised learning, unsupervised learning, semi-supervised learning, and reinforcement learning (which learns from interacting with environments and optimizing actions through trial and error). These techniques are applicable in many domains including predictive analytics, healthcare and software testing and development.

3.3. Applications of Machine Learning in Various Domains

Machine learning is extremely diverse, as it is used in many applications. It can identify objects, recognize spoken words, understand texts, provide recommendations, and make decisions. In the medical context, for example, machine learning systems can assist doctors in diagnosing patients, accelerating manual pre-screening of mammograms, and making complex diagnoses.

Even within software development, the potential use of machine learning models extends beyond CI/CD pipelines. Software estimating models can be trained on historical project data to predict project delivery dates [2-4]. AI-powered help desk chatbots can assist developers with programming-related queries. Smart code analysis tools can suggest code optimization strategies. The primary challenge lies in effectively integrating these models into internal team workflows.

4. Integrating Machine Learning with CI/CD

Computer systems can automate many processes, but there are limits to what automation can achieve; at some stage human intervention is needed. Recent developments in machine learning (ML) have brought additional methods of augmenting CI/CD pipelines. ML is a technique for creating computer programs that improve with experience. Machine learning applications are usually classified as: supervised learning, unsupervised learning, semi-supervised learning, or reinforcement learning. ML is applied in areas ranging from expert systems for medical diagnosis to computer vision.

Development teams are already combining machine-learning tools with a CI/CD pipeline. Having CI/CD pipelines integrated with ML models offers many benefits, including rapid and automatic application development, automatic testing and deployment, and better insight into the application lifecycle and performance. The main challenge of integrating ML with CI/CD pipelines is the need to continuously improve the ML model by rapid identification of the model's failures. Such information enables new ML models to be trained and deployed. Machine-learning techniques can be applied to many stages in a CI/CD pipeline. Maintenance and decision-making capacity can be improved through ML-based predictive analytics. Automated testing and quality assurance can be improved by identifying classes with a high risk of defects and automatically generating test cases for those classes. Some anomaly detection. and root-cause analysis helps in ensuring the stability of the system in the CI (Continuous Integration) process. Multi-faced correlation analysis brings more enlightenment on Cloud-Service Monitoring.

4.1. Benefits of Integration

CI/CD pipelines are the mechanism through which software updates are made to work quickly and with more quality in the continuous integration and continuous deployment (CI/CD) model. And in the same way, machine learning (ML) provides the ability to process information quickly, detect patterns, make intelligent decisions and make predictions. A number of advantages can be drawn from the union of ML together with CI/CD pipelines. Using ML in these pipelines can minimize the error rates, improve the run time durations, save money, and improve overall reliability [5-6]. Predictive models help in predicting build failures; data analysis of previous runs can be used to ascertain that the deployment is ready to deploy. Additionally, scheduling algorithms -- for example, race detection -can benefit from adaptive, ML-based techniques. Anomaly detection methods can use performance metrics to identify abnormal events, and performance metrics, on runtime or stress test, can be embedded into the pipeline for its continuous evaluation.

ML and CI/CD pipelines are particularly useful together for tasks like build automation. When machine learning is applied to problems associated with deployment phases and subsequent rollout, it offers the means to monitor, evaluate, and react to the deployment and release of software updates.

4.2. Challenges in Integration

The fusion of machine learning and continuous integration introduces organizational considerations also. Managers and testers need not be ML experts,

but domain expertise and knowledge about the ML pipeline remain key. Targeting the areas and how ML could help must be done with these stakeholders, be it for application or infrastructure areas. Yet, achieving the necessary level of automation and minimizing human supervision entails significant challenges. Data labeling remains an essential step despite the availability of open-source datasets, dictating that ML techniques cannot be self-sufficient. Application-specific testing requires deep knowledge and substantial infrastructure for execution.

The combined incorporation of Machine Learning (ML) and Continuous Integration/Continuous Delivery (CI/CD) testing presents specific challenges. Firstly, ML gives different results based on training data and may thus predict the same quality metric differently. Second, application testing workbenches themselves must be highly automated and hold domain knowledge. We study in detail predictive analytics, performance analytics and anomaly detection, which enrich both application-targeted aspects (like automated testing), and infrastructure-related aspects (such as failure and bottleneck detection).

5. Machine Learning Techniques for CI/CD Enhancement

Next, machine learning can contribute to improve different phases and parts of CI and CD pipelines. Use predictive analytics to, for example, prioritize tasks and to identify the most likely risks and errors before they happen. Automatic testing would also benefit from deep learning algorithms that can automatically create sophisticated test cases with minimal human effort. Anomaly detection methods find exceptions and anomalies in the software ecosystem and prevent a possible production error [7,8]. Finally, as the pipeline runs, we continuously monitor its performance and use feedback from the model to reconfigure and optimize the pipeline.

Continuous integration and continuous deployment pipelines are integral to the speed of software engineering, and therefore management of them directly impacts the velocity of execution. Managing these resources can be augmented and facilitated using machine learning models for automatic classification, prediction, detection and prioritization of jobs that run on the pipeline. Different stages of CI/CD pipeline—Code, Build, Test, Package, Release, Configure, and Monitor—can benefit from application of ‘different machine football betting

games. Machine learning techniques tailored to enhance their specific functions and mitigate associated risks.

5.1. Predictive Analytics

Predictive analysis uses the historical data for predicting the potential future events. In a CI/CD environment, ML may predict build failures or estimate the testing time. For instance, the chronological test results can be used to predict the failure time a new test while the logs on build engine during compilation can predict build outcome. These predictive capabilities empower engineers to act proactively to prevent issues before they cause operational downtime, and to optimize scheduling.

Test case scheduling is in general a complex problem that is even further complicated when done under time budget constraints, whereML has proved useful due to its ability to predict job runtimes. The deployment EGP is cumbersome as well, therefore it is very valuable to actually end up with the expected result of a new build at this stage of the CI/CD pipeline. Machine learning approaches for automated testing prioritisation to minimize execution time have recently been investigated in recent research. Continuous integration relies on detection and timely notification of build failures, often managed through manual inspection of logs or commands. Automation in failure detection and notification significantly increases productivity, with numerous studies employing ML models to classify failures or predict failure details.

5.2. Automated Testing

Automatic analysis of software artifacts can generate incremental units tests. These tests can be maintained under SCM like manual tests and ran during the build stage. Test oracle design is still a major difficulty. One approach is to use supervised ML algorithms to infer classifiers capable of distinguishing valid and faulty program behaviors. When feature extraction generates training data from existing manual test runs, a variety of classifiers can be generated (decision trees, SVMs, feed-forward nets, etc.). The most effective classifiers can be selected by cross-validation.

Human evaluation experiments are necessary to analyze the effectiveness of the final results. The use of ML helps in the automatic generation of test oracles for continuous testing [9-12]. The generated test oracles are then reused to validate regression faults automatically during the build stage. Automated testing reduces the time during evaluation by eliminating manual testing using smart test oracles generated from software repositories. The key aim of continuous delivery pipelines is to implement code development practices that reduce the time

between feature development and deployment without compromising quality. A key contributor to reduced pipeline lead times is the reduced time for the test phase of the pipeline.

5.3. Anomaly Detection

Anomaly detection in a CI/CD pipeline can be realized using explainable machine learning. The key focus of anomaly detection is to mark a specific class of commit as buggy or non-buggy. Impact estimation of the buggy commit can be performed using ML. Otherwise, an ann is used to predict the probability of failure of the build in the CI/CD pipeline. It is possible to perform the classification task by considering the recent commit history of the project and the historical data of the entire CI/CD pipeline.

Recently, an ML-based approach was introduced by considering historical build data, including the merge-request author, time since the last build, and time of the merge request. The results indicate that monitoring the build time of the pipeline will aid in anomaly detection. Such monitoring techniques can be enhanced using statistical ML. Based on the frequency of anomaly occurrence, further failure analysis is possible [7,3-5]. Yet another approach used the classification of logs considered at each stage of the CI/CD pipeline as a time-series prediction process. This enhanced the failure analysis of a specific stage. Anomaly detection can also be performed in the machine learning infrastructure deployed in a CI/CD pipeline. Monitoring the GPU or network usage of a workload enables the detection of resource utilization anomalies. Upon detection, resource leaking and the inefficient utilization of resources can be avoided by killing specific deadlocks or leaked resources through the identification of such anomalies.

5.4. Performance Monitoring

Once necessary tests are run, proper performance monitoring ensures that new releases do not degrade the performance of the product. Performance monitoring complements performance testing and is performed against the production environment. It involves executing load tests under real world conditions and monitoring the behaviour of the production environment with the particular release deployed. Business KPIs can be defined that indicate whether the release adversely impacts the business functions while system KPIs can include various health indicators of the system when exposed to the increased load during the testing. An ML-driven approach can be introduced where real-world KPIs are monitored and trained under various loads to predict thresholds for the KPIs. The production environment can then be monitored under increased load test conditions after the deployment of the new release. The ML model predicts any

aberrations in the KPIs during the performance testing and raises an alarm if the KPIs go beyond the set thresholds. This makes the performance monitoring completely predictive rather than reactive and helps engineering teams optimize scaling solutions.

6. Case Studies

Constantly evolving markets and consumer expectations require companies to pursue continuous innovation to deliver the features and quality expected by their customers. This translates into requirements to accelerate delivery, continuously monitoring the software for new issues and attacking them as soon as possible. On the other hand, machine learning is a fast-growing field that also has gained significant attention in the last years, being used in several domains and applications. The huge amount of data generated by modern CI/CD pipelines enables the application of machine learning techniques, possibly extracting valuable insight from this information. When applied to CI/CD, machine learning supports the teams by providing prescriptive analytics, helping foresee how to change the current decisions to get a better outcome; predictive analytics, predicting information about changes; or automated decisions, automating activities that require human judgment.

Machine learning application in CI/CD pipelines provides many benefits, such as predictive analytics, anomaly detection, process optimization, automated testing and bug triaging, and performance monitoring and feedback; however, it also entails several challenges, such as identifying relevant and attainable scenarios for its incorporation, selecting appropriate datasets, managing dataset changes over time, defining appropriate metrics of success, and controlling the range and impact of automated decisions [7,13-15]. Several real cases illustrate these benefits and challenges. Predictive maintenance forecasts downtime or time-to-failure, enabling proactive schedules and provisioning. Automated deployment uses data from the project environment to decide when, how, and where to deploy, enhancing operational efficiency. Continuous testing assesses changelogs and historical results to rate the impact of changes and select relevant tests, optimizing testing procedures and resource allocation.

6.1. Case Study 1: Predictive Maintenance

Predictive maintenance refers to the practice of anticipating potential problems before they affect systems or processes, allowing preventive action to be taken. Applied to CI/CD, it involves the use of machine learning to automatically

identify elements of the software workflow that are susceptible to breakdown or other adverse conditions.

In a 2023 study, researchers at the University of Tartu sought to develop a predictive-maintenance model. In their approach, a commit message was taken as being indicative of the probability of a build failure that was dependent on that message. Using classification algorithms, the researchers made the commits themselves the key objects of the prediction, as opposed to files, programmers, or components, which have been the usual focus. Term-frequency measures of words were extracted, with information-retrieval techniques being employed to filter out irrelevant words. Four classifiers were tested: Random Forest, K-Nearest Neighbours, Extremely Randomised Trees, and Logistic Regression.

The approach was assessed using Continuous Integration (CI) build metrics, such as whether the build succeeded or failed, whether test results were included, and whether code coverage metrics had been measured. Data were drawn from 14 open-source projects selected on the basis that they use Travis CI, a popular hosted continuous-integration service, and its corresponding integrated build system, OpenTravis. Analysis covered 763,372 commits for which related builds existed.

6.2. Case Study 2: Automated Deployment

Machine learning algorithms can improve software package deployments by making deployment decisions based on various parameters. Such decision-making supports the continuous deployment workflow very well.

The main goal of continuous deployment is to quickly move software packages to production with minimal manual effort and low risk of failure, while maintaining high quality. The pipeline triggers a release when the pipeline indicates that a piece of software is ready. Releases with a high probability of failure require additional validations such as rollback policies or canary deployments. Machine learning algorithms estimate a risk and possible failure rate, based on a continuous stream of CI/CD pipeline and production data.

6.3. Case Study 3: Continuous Testing

CI/CD pipelines remain at the forefront of discussions on increasing automation in software development and deployment. Machine learning is often applied to enhance current CI/CD pipelines and address new challenges. Testing represents a fundamental component of these pipelines, yet the traditional approach continues to encumber the process with significant manual effort. Scholars have noted that a versatile framework is required to support models capable of adapting to new software versions and learning from associated bugs, all while

preserving model accuracy over time. Continuous machine learning models can classify affected test cases in new commits, thereby enabling a more efficient prioritization of test suites during regression testing.

Another contribution highlights the heavy investment in time and cost that developers devote to testing. Mapping code changes to the affected test targets improves testing efficiency [16]. The experimental support for this approach demonstrates the potential of machine learning to identify the minimal test set required for coverage, thus reducing testing resources. Similar perspectives consider the importance of automated testing within the broader scope of DevOps and continuous delivery, proposing methods to reduce testing efforts.

A drawback of current testing mechanisms is their commitment to verification—the automation of test case execution and reporting—while often excluding validation, or the process of ensuring that test failure results align with business requirements. For instance, although tools like Selenium automate the execution of UI tests and generate reports, the final verdict remains a manual responsibility. All errors exceeding predetermined tolerance thresholds necessitate manual analysis to confirm test success or failure. The substantial manual effort traditionally involved in the validation phase can be mitigated by employing machine learning algorithms to support the final verdict in automated testing tools. Through the analysis of prior testing data, these algorithms can estimate the expected outcome of any test session group.

7. Future Trends in CI/CD and Machine Learning

The greatest potential for future developments lies in the emergent synergy among other technologies such as Artificial Intelligence, Machine Learning, Blockchain and Cloud Computing. Their integration with CI/CD processes is anticipated to shape more enhanced and cost-efficient pipelines. The continuous progress of Cloud solutions is enabling the observation and analysis of the usage of external products across the entire development pipeline. Detailed feedback related to released applications, services and products can be further processed by ML algorithms for predictions related not only to the system's behavior but also stages subsequent to deployment. Incorporating blockchain technologies within CI/CD pipelines allows for management and auditing of software delivery and deployment activities, thereby bringing new dimensions of security to the software development life cycle and positively influencing ESL/BSL policies.

In the context of approaching the 6G era, not only the acceleration of development pipelines is expected but also the provision of improvements during their execution. Applying ML techniques for managing the ci/con aspects of the pipeline can help avoid failures or automatically suggest actions for recovery [9,16-18]. The increasing demand reflected in CI/CD jobs will also result in a similar demand for CI/CD resources and their management, which can be effectively handled through ML techniques. Together, these perspectives provide a glimpse of future trends in the continuous delivery deep learning and machine learning landscape.

7.1. Emerging Technologies

The impact of emerging technologies such as 5G, cloud computing, edge computing, and artificial intelligence on development and delivery processes of products and services is widely recognized. Even companies operating in traditional sectors use these technologies to create innovative products and services. These technologies speed up all business and non-business processes and introduce new concepts like Infrastructure as Code (IaC), Platform as Code (PaasC), Code as Security, and Code as a policy for operations. These ideas help organizations increase the frequency of code deployments to achieve faster delivery of new products and services to market, satisfying customers and attracting new clients.

Infrastructure as Code is based on the idea of migrating infrastructure management into the code. Instead of manually creating virtual machines and configuring them for application deployment and resource management, the infrastructure code allows virtual machines creation and management. Code as Security means defining network and host security rules through code, enabling policymakers to customize security-related rules and describe their regulations about firewalls in a human-readable code format. Smart contracts of the blockchain are also defined by code whose execution is automatically triggered when certain conditions are met. Platform as a Service (PaaS) is one of the service models of cloud computing that automates the provisioning of platforms including hardware and software through code. The idea of Code as a policy for operations comes from Google's Site Reliability Engineering (SRE) concept [2,19-20]. The latter defines a concept based on describing some operational situations and the corresponding actions to take when these conditions are met using code; hence, there is no need to take manual actions whenever these situations occur.

7.2. Predicted Developments

Several recent and emerging technological trends are expected to coalesce, ushering in a new wave of business and engineering process automation. Machine intelligence is one significant driver among these trends. The interest in machine learning has grown rapidly, steadily producing new processes and tools that promise to enhance the entire software delivery pipeline. Machine learning techniques can be applied to the three main stages of a continuous integration/continuous delivery (CI/CD) pipeline—creating a cluster that can deliver a build in a well-controlled environment, building the product, and verifying the build operation and testing the product—thereby mitigating risks and shortening cycle times.

In the cluster-creation stage, the cluster-operator service deploys all services in the cluster and performs static checks on node configurations, network, routing, and so on. During the build stage, the build-operator service consolidates the final build [9,21-23]. In the verification and testing stage, the test-operator service validates the cluster installation, executes a variety of tests (including integration, end-to-end, functional, performance, and scalability testing), and collects test metrics. Machine learning applications enhance cluster management by enabling predictive scaling, node interval recalibration, node-label prediction, and anomaly prediction. Build and test progress benefit from drift inspection, build time and progress prediction, fresh-node allocation, and test time and result prediction.

8. Conclusion

In modern software engineering, the continuous integration and deployment pipeline is pivotal for the delivery of challenging applications. Driving the development of machine learning techniques and their integration into the DevOps pipeline supports software development acceleration. The application of machine learning techniques to auto-regression in the DevOps progress recognition task allows for the detection of vulnerable components in the software project, identifies the risk of merge conflicts, enables timely resolution, and assists in organizing the testing process of the deployed application. Proper analysis of the testing process within the framework of continuous integration and deployment can optimize the automated testing cycle and downtime during application support.

The main machine learning techniques useful for solving CI/CD challenges include predictive analytics, automated testing, anomaly detection, and performance monitoring. Growing concerns about the security of software pipelines have led to explorations of machine learning models for secured CI/CD pipelines, demonstrating superior performance. Real examples of the application of machine learning in continuous integration and deployment—such as predictive maintenance, automated deployment, and continuous testing—prove that these techniques can be used effectively. The summary of machine learning techniques applied to CI/CD can serve as guidelines for future practice. Despite existing achievements in leveraging machine learning for analysis, optimization, monitoring, and security of CI/CD pipelines, new challenges constantly arise that require the development of new ideas and solutions

References

[1] Karamitsos I, Albarhami S, Apostolopoulos C. Applying DevOps practices of continuous automation for machine learning. Information. 2020 Jul 13;11(7):363.

[2] Shivadekar S, Halem M, Yeah Y, Vibhute S. Edge AI cosmos blockchain distributed network for precise ablh detection. Multimedia tools and applications. 2024 Aug;83(27):69083-109.

[3] De Silva D, Alahakoon D. An artificial intelligence life cycle: From conception to production. Patterns. 2022 Jun 10;3(6).

[4] Battina DS. DevOps, a new approach to cloud development & testing. International Journal of Emerging Technologies and Innovative Research (www. jetir. org), ISSN. 2020 Aug 8:2349-5162.

[5] Maheshwari A. Digital transformation: Building intelligent enterprises. John Wiley & Sons; 2019 Sep 11.

[6] Subramanya R, Sierla S, Vyatkin V. From DevOps to MLOps: Overview and application to electricity market forecasting. Applied Sciences. 2022 Sep 30;12(19):9851.

[7] Dang Y, Lin Q, Huang P. Aiops: real-world challenges and research innovations. In2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion) 2019 May 25 (pp. 4-5). IEEE.

[8] Ugwueze VU, Chukwunweike JN. Continuous integration and deployment strategies for streamlined DevOps in software engineering and application delivery. Int J Comput Appl Technol Res. 2024;14(1):1-24.

[9] Kim G, Humble J, Debois P, Willis J, Forsgren N. The DevOps handbook: How to create world-class agility, reliability, & security in technology organizations. It Revolution; 2021 Nov 30.

[10] Myllynen T, Kamau E, Mustapha SD, Babatunde GO, Collins A. Review of advances in AI-powered monitoring and diagnostics for CI/CD pipelines. International Journal of Multidisciplinary Research and Growth Evaluation. 2024 Jan;5(1):1119-30.

[11] Panda SP. The Evolution and Defense Against Social Engineering and Phishing Attacks. International Journal of Science and Research (IJSR). 2025 Jan 1.
[12] Lo D. Trustworthy and synergistic artificial intelligence for software engineering: Vision and roadmaps. In2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE) 2023 May 14 (pp. 69-85). IEEE.
[13] Battina DS. An intelligent devops platform research and design based on machine learning. training. 2019 Mar;6(3)
[14] Battina DS. Ai-augmented automation for devops, a model-based framework for continuous development in cyber-physical systems. International Journal of Creative Research Thoughts (IJCRT), ISSN. 2016 Sep 3:2320-882.
[15] Enemosah A. Enhancing DevOps efficiency through AI-driven predictive models for continuous integration and deployment pipelines. International Journal of Research Publication and Reviews. 2025 Jan;6(1):871-87.
[16] Panda SP. Augmented and Virtual Reality in Intelligent Systems. Available at SSRN. 2021 Apr 16
[17] Dragičević T, Wheeler P, Blaabjerg F. Artificial intelligence aided automated design for reliability of power electronic systems. IEEE Transactions on Power Electronics. 2018 Dec 20;34(8):7161-71.
[18] Fitsilis P, Tsoutsa P, Gerogiannis V. Industry 4.0: Required personnel competences. Industry 4.0. 2018;3(3):130-3.
[19] Panda SP. Securing 5G Critical Interfaces: A Zero Trust Approach for Next-Generation Network Resilience. In2025 12th International Conference on Information Technology (ICIT) 2025 May 27 (pp. 141-146). IEEE.
[20] Erich FM, Amrit C, Daneva M. A qualitative study of DevOps usage in practice. Journal of software: Evolution and Process. 2017 Jun;29(6):e1885.
[21] Qumer Gill A, Loumish A, Riyat I, Han S. DevOps for information management systems. VINE Journal of Information and Knowledge Management Systems. 2018 Feb 12;48(1):122-39.
[22] Panda S. Scalable Artificial Intelligence Systems: Cloud-Native, Edge-AI, MLOps, and Governance for Real-World Deployment. Deep Science Publishing; 2025 Jul 28.
[23] Cheng Z, Jia X, Gao P, Wu S, Wang J. A framework for intelligent reliability centered maintenance analysis. Reliability engineering & system safety. 2008 Jun 1;93(6):806-14.

Deep Science Publishing, 2025
https://doi.org/10.70593/978-93-7185-060-5

DeepScience
Open Access Books

Chapter 3: Exploring the Role of Artificial Intelligence in Enhancing Site Reliability Engineering Practices

1 Introduction

The high and ever-increasing demand for availability and reliability of IT services has led to the creation of a dedicated domain within engineering teams that is solely responsible for the daily operational work. Managing the monitoring portal, reducing false-positive incidents, coordinating the incident management process, implementing efficient workflows for log analysis, creating runbooks for regular tasks, etc. can be tricky, especially when meeting Service Level Agreements (SLAs) and business expectations. The concept of Site Reliability Engineering (SRE) fully encapsulates these tasks and their correct management.

The number of IT companies that have implemented an SRE team is growing exponentially, and with it the volume of operational data that can be collected. That growth enables Artificial Intelligence (AI) tools to make Ichbiah's vision come true, applying AI to make operations more intelligent, efficient, and performant. Today, AI is becoming strongly connected to the concept and practice of SRE, starting with the management of alerts and incidents, then moving forward, slowly, towards the automation of the day-to-day workload. Section 2 describes SRE and its main metrics; section 3 presents an overview of AI and its common uses in IT; subsequently, section 4 shows concrete implementations of AI within SRE. Finally, sections 5, 6, and 7 explore the benefits, challenges, and real world case studies related to the topic, while section 8 discusses future trends.

2. Overview of Site Reliability Engineering

Site Reliability Engineering (SRE) is coming of age, with more companies of all sizes building and maturing their own SRE teams. The practice of applying software engineering principles and practices to infrastructure and operations problems has been proven to make software and systems scalable, reliable, and highly available. As SRE groups form and mature, they strive to improve the maturity of their service and SRE teams by incorporating and applying technologies such as artificial intelligence. Google's SRE team coined the term. The basic idea behind SRE is to treat operations as a software problem. SREs create software and systems that make large-scale platforms more scalable and reliable. Nowadays, an SRE team is commonly viewed as a programmer inclined toward operations [1-2]. It applies the principles of service-level objectives, error budgets, and monitoring. The SRE group aims to bridge the development operations and operations team, ensuring a high-performing, highly efficient team that meets success criteria, which is defined by business goals.

Artificial Intelligence (AI) can help with these criteria. However, most companies in the early days of SRE have a smaller team dealing with multiple operations tasks responsible for the availability of all services. Hence, leveraging AI to increase SRE team power while enhancing the quality and delivery of solutions can help them move from reactive to predictive monitoring. AI-powered platforms can also aggregate and prioritize tasks identified across the organisation.

2.1. History and Evolution

Strategic development and operational processes in information technology have been generating growing interest in recent years. Given that information technology has become almost ubiquitous in everyday life, processes where constant availability is mandatory must be introduced. Traditional forms of IT system management no longer meet the demand for availability at all times. Indeed, the difference between the demand for availability and the ability to maintain it led to the creation of the Site Reliability Engineering (SRE) approach. SRE has become particularly important, and its pioneering ideas are regularly applied within many cooperative projects with companies such as Google.

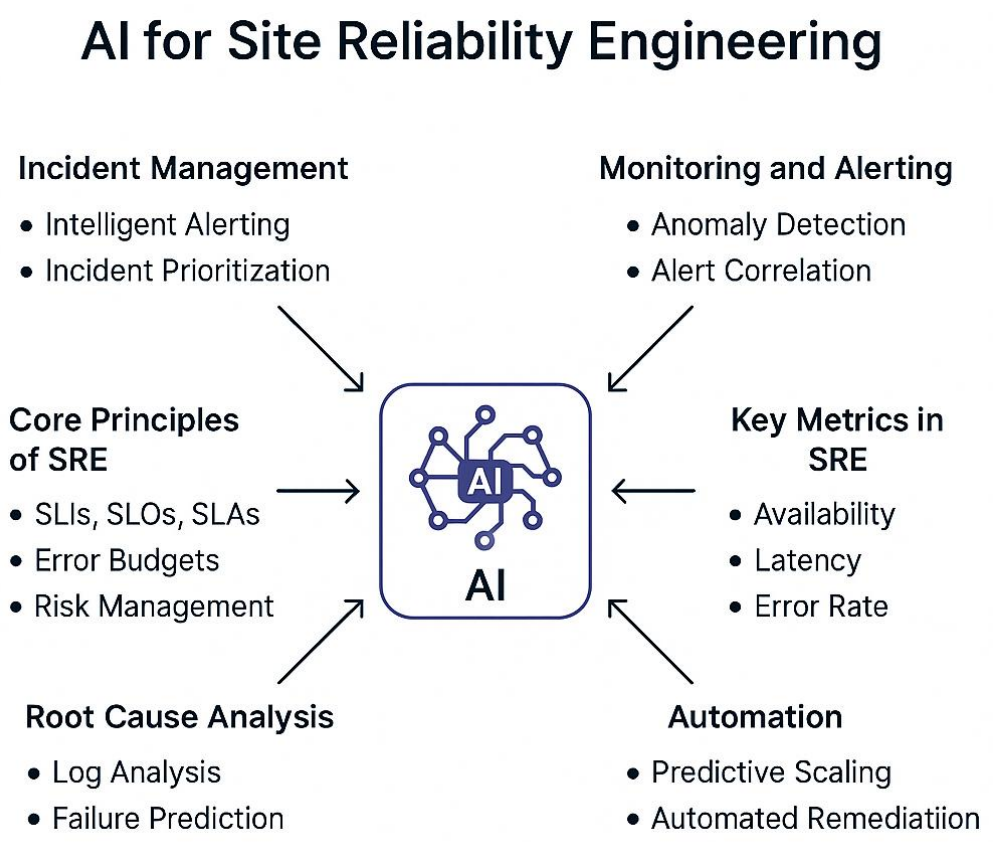

Fig 1. AI for Site Reliability Engineering

Site Reliability Engineering is a holistic approach to system administration [1-2]. It combines automation and system administration with new workloads, concentrating on reliability. Monitoring systems and services thoroughly, alerting teams to service outages, and responding to incidents are essential aspects of SRE. At the same time, SRE performs tasks defined in other approaches. Nonetheless, SRE exploits error budgets, considers performance metrics in terms of availability, and optimises systems for operability, reliability and availability.

2.2. Core Principles of SRE

The core principles and processes of Site Reliability Engineering (SRE) are mostly related to metric-driven automation, monitoring, incident response and postmortem, capacity planning, and management of workloads and tasks. SRE principles emerged as corporations began to implement SRE methodologies. Google and several other enterprises began to describe the culture, operations, and tasks of site reliability engineers.

The industry has determined that the reliability of an IT service or workload should be described by its users, making it a customer-centric matter. Two objectives must be balanced: maximize availability, and maximize the pace of change. Errors are inevitable, and services' reliability should be expressed as a tradeoff between risk and utility. Conceptually, SRE can be defined as the application of a software-engineering mindset to operations tasks to ensure scalable and reliable services. Service Level Indicators (SLIs) are established to quantify specific aspects of SLOs and SLAs. An Error Budget is defined as 100 minus the error-rate budget, serving as a quantified tolerance of remaining errors that may occur within a given period.

SRE principles can be used to ensure the reliability and availability of AI workloads during training and inference, helping to manage error budgets and capacity concurrently with users' requirements.

2.3. Key Metrics in SRE

Some of the key metrics to track when evaluating both stability and reliability in SRE. 3 of these are core metrics in particular. The average time to detection (MTTD) establishes the time period it needs to be discovered if a system has an issue On the other hand, the mean time to resolution (MTTR) is the amount of time spent working on fixing the system after the incident. Lastly, rate of change failure quantifies the percentage of changes or deployments that result in failure in a production environment.

Although the above metrics are indicative of system dependability, reliability, and robustness, error budget is highly associated with reliability and robustness. Error budget is about how much reliability a site should have within a particular time period. SREs have to figure out the lowest level of unavailability or delay that's still sufficient for user satisfaction [3-5]. It is the remaining from 100% and the SLO (service-level objective). The SLO can be formulated using various metrics defined by users, such as availability, latency, Mean Time to Failure (MTTF), Mean Time to Repair (MTTR), or Cycle Time. The error budget is then derived from the SLO and can indicate the allowed downtime due to latency or failure percentages over a specific period.

3. Artificial Intelligence: A Primer

The services we use every day are supported by heterogeneous systems that have grown exponentially in size and complexity. They are used by millions of users and have a high standard for service availability. Companies like Google, Meta,

Amazon, and Microsoft allocate dedicated teams to guarantee the reliability of their systems. These teams are called Site Reliability Engineering (SRE).

Similarly, the mass adoption of the Internet, mobile devices, social networks, democratization of cloud computing, open source, and big data has offered industries and market sectors opportunities never imagined. From the mid-20th century to the present, there has been an accentuated advance in computation systems and, more recently, accompanied by a surge in artificial intelligence (AI). Today, AI is gaining more space in companies and the academic world, becoming a reality integrated with IT.

3.1. Definition and Scope of AI

Artificial intelligence (AI), as defined by artificialintelligence.org, is "the theory and development of computer systems able to perform tasks normally requiring human intelligence." Examining the earliest definitions of AI reveals an emphasis on the human intellect aspect of intelligence, thereby establishing it as intelligence demonstrated by an entity using human reasoning. However, the scope of intelligence has broadened since then, leading to AI technologies that simulate intelligence in any animal or organism and even in human-created machines and programs. Modern implementations of AI, such as artificial neural networks (ANNs), sentiment analysis, and image recognition, illustrate the wide and varied interpretation of AI today.

Buchanan and Smith categorize AI applications in the form of programs into three paradigms that are relevant to AI use cases in information technology (IT) operations. The first paradigm comprises applications that mimic human typing, reasoning, and intelligence [6-8]. The second encompasses the creation of programs with complex architectures that simulate human intelligence on multiple levels. The third involves the construction of systems that determine the adequacy of the information they provide and acquire new knowledge from their environment. These AI paradigms underpin the subsequent applications of AI within the SRE domain.

3.2. Types of AI Technologies

Artificial intelligence encompasses several categories of technology that professional AI groups distinguish. At the strictly technical side are broad sub areas, such as AI following abstract human reasoning, intelligence embedded in biological systems, and perceptual intelligence. Elucidating the contribution of diverse AI technologies within each broad category uncovers their roles in attaining overall objectives—especially when AI is deployed in supporting Site Reliability Engineering (SRE) practices. Such insights emerge by examining the

utilization paradigms of specifically termed AI methods, for example, different kinds of intelligent neural networks or various reinforcement learning strategies.

Another approach starts with the different types of AI-based applications identified by the Information Technology (IT) community, including logical and technical IT functions. These functions—incorporated within the IT community's AI capabilities—are applied to fulfil the previously mentioned objectives. Relating applied AI technologies to SRE practices thus aids the selection of approaches suitable for specific AI application domains. Transformational use of AI within SRE appears in multiple facets, such as integrating AI methods and principles in incident management, as well as applying AI in prediction and forecasting, task Automation and Orchestration, or in business intelligence.

3.3. AI Applications in IT

AI applications in a Site Reliability Engineer's (SRE) work can be grouped into four categories: Automation, Pattern Recognition, Reasoning, and Forecasting. Within each category, several tasks can directly enhance site reliability.

Automation helps the SRE team to handle repetitive tasks, ensuring consistent and timely execution and thereby improving overall system health [9]. For example, automated responses can maintain specified performance levels during incidents. SRE workflows can also incorporate AI-driven automation to enhance procedures and ensure thorough testing of new automation technologies. Pattern Recognition is a set of difficult-to-automate tasks, which aim at gaining insights from heterogeneous data for pro-active issue solving. SRE teams boost the sensitivity of monitoring, correlated incident discovery and roles discovery through pattern recognition

4. Integrating AI into SRE Practices

Artificial intelligence (AI) refers to the ability of machines to perform tasks that would require the application of human intelligence, in particular cognitive functions that are linked to the human mind. Artificial intelligence in information technology is based on artificial neural networks, which exist in both software and hardware. AI systems are composed of machine learning, natural language processing, expert systems as well as speech recognition. There are a number of AI subdomains that are specifically relevant to IT practitioners—cybersecurity AI, DevOps AI, experimentation AI, etc.

For SRE teams looking to increase reliability, minimize toil, and automate as much of the mundane and repetitive work as possible, incorporation of AI in SRE

is inevitable. The plethora of monitoring data in the industry enables incident management and predictive analytics using AI. The automation of common SRE tasks, such as troubleshooting, analyzing logs, scheduling tasks, managing configurations, and documenting also receive the benefits of AI. These KPIs help inform my choice of AI applications. For instance, adherence to Service Level Objectives (SLOs) depends on incident management systems that report the states of issues and analyze the patterns of incidents. AI-based solutions for problem identification, root cause analysis, and remediation help reduce the Mean Time To Resolution (MTTR), while automation of repetitive operational tasks can be an effective way of minimizing toil.

4.1. AI for Incident Management

Challenges with reliability can be disruptive and frustrating for customers, both for B2C apps and internal-facing apps and services within an organization. To minimize negative customer impact and identify reliability problems as quickly as possible, organizations invest heavily in monitoring and alerting systems. These systems notify engineers when potential issues arise and provide a work context for sequential incident response.

The right artificial intelligence can help make incident response more effective by, first, minimizing false-positive discernment via enriched alert grouping based on causal relationships, and then, by summarizing and analyzing the incident context before passing—essentially triaging—the alert to human engineers [7,9-10]. Through this approach, AI can improve SLO performance, increase availability and customer trust, reduce time spent resolving issues, and help ensure customer experience stability.

4.2. Predictive Analytics in SRE

Various commercial vendors offer predictive analytics applications, many of which primarily focus on predictive forecasting and automatic issue detection. For instance, CloudFabrix CognitionHub predicts business and operational risks and the associated root causes; AI ExceptionInsight identifies anomalies and predicts performance issues and their business impact; and Moogsoft AIOps automatically detects incidents and predicts the business impact of IT services. Beyond these, SREs can leverage such tools for risk identification and recommended responses.

Predictive analytics uses machine learning algorithms and models to analyze data patterns and forecast future outcomes. In SRE, it determines the likelihood of a service approaching its SLO violations or identifies anomalous behaviors that might lead to failures. Predictive insights empower SREs to maintain error

budgets, act proactively, and rapidly respond to incidents, thereby supporting reliability targets with greater confidence.

4.3. Automation of Routine Tasks

Automation of routine tasks is a key component of site reliability engineering practice. An implementation of AI technologies in the area automates routine, repetitive work. Tasks suitable for AI automation include root cause analysis of incidents and failures, documentation, escalation of technical issues, hardware failures, and database management tasks.

Root cause analysis identifies the primary cause of an incident or failure. The process involves two distinct phases, detection and identification, and failure analysis. The detection phase determines whether an incident or failure has occurred. Identification and failure analysis determine the root cause. The two areas include anomaly detection using regression, pattern recognition, error detection and classification and information retrieval from knowledge bases. Using machine learning classifications learned from historical incident data, AI technology can automatically perform root cause analysis

5. Benefits of AI in Site Reliability Engineering

Artificial Intelligence (AI) into the Site Reliability Engineer- ing (SRE) domain provides significant enhancement and brings us closer to the vision of a completely self-healing system. AI adoption enhances the reliability of IT systems by automating the detection and handling of service-impacting events. This progression allows SREs to concentrate on the development of end-to-end service reliability, thereby maintaining an acceptable imbalance between new feature development and operational work.

Implementing AI in SRE activities introduces predictive capabilities through observational analysis, enabling the detection of ongoing or imminent problems before a customer impact occurs [1,11-14]. This proactive nature of work enhances the SRE team’s time and effort. AI can also provide automated responses to uncommon and highly sophisticated issues lowering MTTD and MTTR, which will in turn improve both general knowledge and overall workforce readiness. Together, these improvements lead to an overall improvement when handling of operations-related issues within SRE.

5.1. Improved System Reliability

Site Reliability Engineering (SRE) is a discipline that incorporates aspects of software engineering, design, and operations whose goals are to create scalable

and highly reliable software systems. The main goals are to create ultra-scalable and highly reliable software systems. In the beginning, Google's SRE team described their approach to managing and developing services; today, it is viewed as the engagement of operations within application development to produce reliable systems. SRE is not a new discipline, but it can be observed in IT teams within different organizations.

Key system reliability metrics—Service Level Indicator (SLI), Service Level Objective (SLO), and Service Level Agreement (SLA)—are fundamental in the reliability discussion. SREs can improve those metrics by using Artificial Intelligence (AI). Combining AI with SRE can assist in more effectively managing large Cloud and On-Prem services, and operational teams can benefit from AI as a Service (AIaaS) for several operational tasks.

Artificial Intelligence (AI) refers to intelligence demonstrated by machines, in contrast to the natural intelligence displayed by humans and other animals. In computer science, AI research defines "intelligent agents" as any device that perceives its environment and takes actions that maximize its chance of successfully achieving its goals [13,15-17]. Colloquially, the term "artificial intelligence" is often used to describe machines or computers that mimic "cognitive" functions commonly associated with humans, such as "learning" and "problem-solving". Some applications of AI include expert systems, natural language processing, speech recognition, and machine vision.

5.2. Enhanced Operational Efficiency

AI can enhance operational efficiency through the automation of repetitive, mundane, and time-consuming SRE tasks. Such automation helps improve task execution, reduces human errors caused by the unpleasant nature of monotonous work, and contributes to meaningful Time to Live (TTL) for SRE engineers. In a recent survey, 28.57% of respondents affirmed that the adoption of AI has optimized the TTL of SRE engineers within their organizations.

Example SRE tasks that AI can streamline include dashboards and runbooks management, automation for data gathering and display, and root-cause checking. AI assistants may automate these responsibilities or simply serve as support for human engineers. Moreover, knowledge management systems benefit from AI by facilitating information storage, retrieval, and distribution—such as delivering playbooks based on the context of conversations. Training and onboarding also profit from AI integration, as AI-enhanced systems provide introductory knowledge, ensuring that new team members are informed about the company's overarching operational procedures. In the same survey, 37.14% of

SRE engineers concurred that AI implementation has effectively facilitated training and onboarding processes.

5.3. Proactive Issue Resolution

The core goal of SRE is to build better systems with improved reliability and stability, and resolving issues faster is key to achieving this goal. As an important SRE function, proactive issue resolution includes making continual improvements, accelerating incident resolution, and automating recurring tasks.

A proactive SRE system calls for both corrective and preventive action. On the incident management side, the system measures incident and postmortem data with machine-learning support and identifies high-impact services. It enables the team to focus on those areas that cause most outages and hence prevent future incidents.

6. Challenges and Considerations

Several challenges and considerations confront the integration of AI into SRE practices. The main concerns are related with the amount of data used. Data deemed sensitive being moved to centralised data lakes violates legacy security paradigms and raises the level of potential attack. Non–natural language–based AI approaches ranging from the vision-based ones we study here to the oldest forms of AI pose as yet unfamiliar adversarial risks and failure modes to deployed systems. Since datasets used by ML are not the same as those employed by classical rule-based means, misconfiguration and bias present unsuspecting risks to safety, fairness, and privacy. There are also policy implications resulting from misinterpretation and extrapolation of interpretability risks. Bad AI can take up, accidentally, and amplify biases, and damage us through discriminatory or unfair decisions. Injecting AI systems into an SRE work pattern can disrupt established patterns of behaviour and needs careful rollout.

Reaping the rewards of AI by improving the reliabilityAs tempting as it may be to use AI to improve the reliability in SRE, deploying it in SRE also carries risks similar to AI risks elsewhere [18-20]. To properly address these risks, control of AI technology needs to be increased beyond just the standard set of SRE skills, with mitigations baked right into your monitoring and policy enforcement tools. Without these, AI’s intelligence may be countered, or even inverted, by lurking adversarial or interpretability challenges.

6.1. Data Privacy and Security

The discipline of Site Reliability Engineering (SRE) is based on the premise that operational data is crucial during failure handling, root cause analysis, monitoring of availability and latency, etc. Data security and privacy are essential when you use AI in SRE. Laws, practices, algorithms and models in general, and machine learning as subset of AI in particular, can be reluctant not to be inclined to transfer historical data and have a wide range of historical data to train and learn, which might result in revelation of the sensitive infrastructure related information/confidential.

Some profiles would have been inherently secure by virtue of residing inside a secure platform and it is the AI training process that is at risk. “Open-source makes models a little more proven and a very little bit more secure, [since they are] publicly scrutinized. But privacy laws such as the General Data Protection Regulation (GDPR) and new data privacy requirements in the wake of COVID-19 make it clear that private or sensitive data must be treated with respect when being processed by AI. Any AI model for tempering analysis must be deployed and tested within secure environments with appropriate data policies and procedures to prevent data leaks or exposure.

6.2. Bias in AI Algorithms

The increasing adoption of Intelligent Systems, particularly in the mission-critical environments found within Site Reliability Engineering operations, raises important concerns regarding algorithmic bias and its implications for equity, fairness, and non-discrimination. Given the growing reliance of SREs on Artificial Intelligence to proactively address reliability challenges, the data utilized for training diagnostic, remedial, and prediction models must accurately represent the diverse features of the deployment environment, including its many stakeholders.

Symptomatic of the broader population, biased datasets fail to account for the characteristics of subgroups, such as women and minority ethnic populations, and this shortcoming is transmitted through the trained algorithm. For example, a prediction model built primarily from data associated with a single demographic population will fare poorly when applied to other groups, often producing error rates that are several multiples higher. Left undetected, AI systems lacking in equity lead to the misdiagnosis of reliability incidents in particular deployment environments and to inequitable resource allocations for solving their underpinning operational problems.

6.3. Integration Complexity

The transformative effect of AI in SRE is undeniable, yet certain challenges, especially those linked to the privacy of sensitive data, cast a shadow over its widespread adoption. Notwithstanding the advantages in efficiency, effectiveness, and automation, the technology is still susceptible to pitfalls associated with biased and opaque algorithms [19,21-22]. Moreover, the practical implementation of AI may deviate from initial intentions if enterprises attempt to shoehorn AI technologies into existing structures without adequate remodeling and preparation. Commercial tools for alert prioritization, like Moogwares Horizon AI, Moogsoft AIOps, and BigPanda Auto Remediate, heavily depend on automatic integrations with cloud services, monitoring systems, IT service tools, messaging platforms, and collaboration resources. While such integrations facilitate rapid deployment, they might lack explicit connections to core SRE principles, potentially limiting user flexibility and future customization.

In AI adoption, an efficient, effective, and sustainable approach frequently necessitates an incremental process that begins with the decoupling and normalization of data, based on particular business objectives. This phased methodology enables the integration of AI-generated insights, like predictive analytics, into monitored infrastructure or business intelligence to develop real-time AI dashboards that provide intuitive visual explanations as an essential feedback loop. The maturity of each step lays the groundwork for subsequent phases and can be both evaluated and refined over time. However, some AI implementations—particularly those related to root cause analysis and resolution—may require a revisiting of architectural design and long-term business strategy before their promises can be fully realized. By contrast, the incremental system required for alert noise reduction remains valid, regardless of the eventual direction of additional AI integration.

7. Case Studies

Two use case scenarios are presented to illustrate typical benefits of applying AI in SRE. The first example assumes an organization is not Amazon or Google, lacking a large team and extensive operational data; the second involves a company progressing toward a data-driven approach with a task-specific machine learning system.

In the first use case, the organization pursues a gradual integration of existing generative AI products into the SRE workflow. The team describes the incident

with words, and the AI system generates a timeline of event details, the root cause and direct cause, key observation tables, a list of affected customers, and a summary of the actions taken [19,21-22]. Although this process remains manual, the AI-generated outcomes provide a transparent timeline and essential information previously associated with past incidents, enabling the team to create shared documentation and quickly comprehend the progress of the incident. Machine-generated content also scours incident data to pinpoint potential customers affected by an incident and determines the root and direct causes to inform reviews. This solution accelerates incident resolution and mitigates the adverse impact of unknown work shifts.

The second scenario involves a collaborative effort between SRE and data scientist teams to develop a job-monitoring dashboard for a Spark data pipeline. SRE monitors mission-critical batch jobs that depend on the completion of predecessor runs and continuous job execution; failure to complete within an anticipated time frame is considered an incident. Past incidents deviate from a recurring pattern, with the incident timing exhibiting significant variability. Automation is essential. Data scientists design a Transformer-based time series model that, when receiving job metrics as input, estimates the job completion time and predicts the time required for each stage within the job, thereby enabling the estimation of incident timing despite fluctuating execution times. Compared to traditional heuristic models and other state-of-the-art machine-learning models, the transformer model achieves the best inference performance, although high variability during nighttime operations presents ongoing challenges for accurate job time prediction. Subsequent efforts aim to extend the model to cater to diverse job types, supported by the recently introduced data framework.

7.1. Successful AI Implementations in SRE

July 2018: Google has integrated AI-powered chatbots into its internal support organization. The new assistants, named Google Agent Assist, can analyze tickets to determine if there is an existing solution and provide relevant knowledge content to technicians. Those contractors report 10%–20% improvements in resolution times for cases resolved using the assistant. Other Agent Assist implementations include examining reply suggestions made by human agents, as well as engaging with enterprise users of the products in order to troubleshoot their support tickets. Incident Management:

Major incident management at Google is supported by an AI bot named Q. It monitors the creation of major incidents by incident commanders (roles which are usually assigned to senior site reliability engineers). When one is created, the bot reminds the assigned incident commander to introduce themselves to the

channel, reach out to developers and business units affected, and to assign follow-up tasks. The bot also automatically posts a link to the Major Incident Management post-mortem template at the conclusion of the incident, makes recommendations for an incident health score, and tracks release notes relevant to the page.

7.2. Lessons Learned from Failures

Complex, large-scale systems, and the related AI models, are prone to failures. Nevertheless, SRE organizations must do their best to mitigate the negative consequences. With the growing adoption of AI, reliability teams must consider the following concerns, which could lead to failures in the SRE domain:

Data Privacy and Usage. GDPR, Commercial Law, and Intellectual Property describe the rules for data usage, retention, and sharing in organizations. SRE practitioners must work with the DPO, Legal, or other related teams to understand how to use production data in the AI/ML models. For instance, even if the data is anonymized, it could still infer answers to certain questions or be used for unintended purposes. Crucially, data privacy concerns not only relate to Personally Identifiable Information but also to company-specific trade secrets and to personally or commercially sensitive information [23-25]. Therefore, when an SRE organization uses such data either in an AI/ML model or as the raw data, the associated processes should adhere to the organization's data privacy and usage policies.

Bias and Fairness. When training data is not representative of the entire production environment, the related AI/ML model may have biased views. For example, if the training data contains only past year outage information but does not cover the different patterns observed when a rerouting mechanism or a new algorithm is in place, then the model predictions could be biased. By definition, bias also relates to the interpretability of such models. An absence of interpretability can significantly affect the human-centric process of decision-making.

8. Future Trends in AI and SRE

Artificial intelligence promises universal applicability for tackling complex tasks and is set to impact every profession. The SRE sector, inherently difficult due to its relentless operational responsibilities, is no exception. Successful adoption of even foundational AI techniques is capable of redefining the work of SREs in the foreseeable future.

Advanced algorithms, particularly in natural language processing and computer vision, enable AI systems to comprehend and process human language and image data effectively. Such capabilities are valuable for managing support tickets and addressing multimedia-related incidents and problems [26-27]. Emerging brain-inspired or neuromorphic chips perform actions with enhanced decision-making capabilities, rendering their behavior increasingly human-like. Future AI systems, acting as assistant SREs, will likely demonstrate augmented cognitive abilities, improving both efficiency and decision-making quality.

8.1. Emerging AI Technologies

AI steadily advances, spanning myriad domains, including SRE practices. Availability, latency, saturation, and OPEX-related tasks are especially amenable to various AI strategies. Detecting incidents early and accurately through AI techniques like natural language processing and classification increases system reliability. Predicting capacity demands enhances latency and saturation metrics. Automating manual, repetitive processes not only streamlines operations but also reduces risk, benefiting all four metrics.

With the maturation of AI solutions, their adoption becomes straightforward. Enterprises can now leverage state-of-the-art AI during interactions with large, mature vendors that provide AI-supported solutions. The development of customized or open-source AI applications has been simplified by abundant documentation and crowdfunding. Cloud service providers offer highly usable AI services, readily integrated with existing environments. Consequently, a new SRE work style emerges, focusing on AI-based operational assistance.

8.2. The Future of Work in SRE

The future of work coincides with the age of AI. Developments in AI have sparked, in several countries, a discussion about the downsides for society, with warnings of mass unemployment and a mass decline of jobs. While a mass decline of jobs in SRE appears unlikely, the nature of SRE work will change substantially. The reason for that is not that the tasks of an SRE disappear, but that they will be added to by new and more complex tasks.

New jobs will arise that revolve around designing, improving, and supervising AI systems, usually requiring much higher skills and training than the original tasks of a job. SREs will still supervise these new AI systems, but supervision will be more difficult and cognitively demanding, especially in the early stages of the system's deployment[23]. An SRE must understand prejudice and bias in AI systems, be able to assess their quality and reliability, and anticipate

unintended consequences or emergent behaviors. New tasks are usually much more complex and call for higher skills than the original tasks.

9. Conclusion

The role of AI in site reliability engineering remains a fertile topic of research, underpinning efforts that tap these technologies to enhance the reliability of production systems. The impact of AI on SRE emerges through the automation of processes, scalability, and support for decision-making. SRE automatons are monolithic components dedicated to implementing reliability policies, executing tasks such as remediation, lifecycle management, auto scaling, disaster recovery, and cost optimization.

The incorporation of logical AI techniques into IBM's SmartCloud Orchestrator (SCO) framework has enabled the development of an AI capability supporting proactive incident management for SREs. Artificial intelligence automates the detection of anomalies in system performance and identifies recurring problems by correlating diverse data, encompassing logs, network device data, outage reports, social media entries, and weather information. Regression Analysis thereby forecasts threats to site reliability and formulates hypotheses concerning incident root causes.

References

[1] Mohapatra PS. Artificial Intelligence and Machine Learning for Test Engineers: Concepts in Software Quality Assurance. Intelligent Assurance: Artificial Intelligence-Powered Software Testing in the Modern Development Lifecycle. 2025 Jul 27:17.

[2] Erich FM, Amrit C, Daneva M. A qualitative study of DevOps usage in practice. Journal of software: Evolution and Process. 2017 Jun;29(6):e1885.

[3] Qumer Gill A, Loumish A, Riyat I, Han S. DevOps for information management systems. VINE Journal of Information and Knowledge Management Systems. 2018 Feb 12;48(1):122-39.

[4] Ugwueze VU, Chukwunweike JN. Continuous integration and deployment strategies for streamlined DevOps in software engineering and application delivery. Int J Comput Appl Technol Res. 2024;14(1):1-24.

[5] Kim G, Humble J, Debois P, Willis J, Forsgren N. The DevOps handbook: How to create world-class agility, reliability, & security in technology organizations. It Revolution; 2021 Nov 30.

[6] Myllynen T, Kamau E, Mustapha SD, Babatunde GO, Collins A. Review of advances in AI-powered monitoring and diagnostics for CI/CD pipelines. International Journal of Multidisciplinary Research and Growth Evaluation. 2024 Jan;5(1):1119-30.
[7] Panda SP. The Evolution and Defense Against Social Engineering and Phishing Attacks. International Journal of Science and Research (IJSR). 2025 Jan 1.
[8] Lo D. Trustworthy and synergistic artificial intelligence for software engineering: Vision and roadmaps. In2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE) 2023 May 14 (pp. 69-85). IEEE.
[9] Battina DS. An intelligent devops platform research and design based on machine learning. training. 2019 Mar;6(3).
[10] Battina DS. Ai-augmented automation for devops, a model-based framework for continuous development in cyber-physical systems. International Journal of Creative Research Thoughts (IJCRT), ISSN. 2016 Sep 3:2320-882.
[11] Panda SP. Securing 5G Critical Interfaces: A Zero Trust Approach for Next-Generation Network Resilience. In2025 12th International Conference on Information Technology (ICIT) 2025 May 27 (pp. 141-146). IEEE.
[12] Calvo-Rubio LM, Ufarte-Ruiz MJ. Artificial intelligence and journalism: Systematic review of scientific production in Web of Science and Scopus (2008-2019).
[13] Hannousse A, Yahiouche S. Towards benchmark datasets for machine learning based website phishing detection: An experimental study. Engineering Applications of Artificial Intelligence. 2021 Sep 1;104:104347.
[14] Divya S, Indumathi V, Ishwarya S, Priyasankari M, Devi SK. A self-diagnosis medical chatbot using artificial intelligence. Journal of Web Development and Web Designing. 2018 Apr 7;3(1):1-7.
[15] Bonner E, Lege R, Frazier E. Large Language Model-Based Artificial Intelligence in the Language Classroom: Practical Ideas for Teaching. Teaching English with Technology. 2023;23(1):23-41.
[16] Rao AS, Vazquez JA. Identification of COVID-19 can be quicker through artificial intelligence framework using a mobile phone–based survey when cities and towns are under quarantine. Infection Control & Hospital Epidemiology. 2020 Jul;41(7):826-30.
[17] Dragičević T, Wheeler P, Blaabjerg F. Artificial intelligence aided automated design for reliability of power electronic systems. IEEE Transactions on Power Electronics. 2018 Dec 20;34(8):7161-71.
[18] Fitsilis P, Tsoutsa P, Gerogiannis V. Industry 4.0: Required personnel competences. Industry 4.0. 2018;3(3):130-3.
[19] Panda S. Scalable Artificial Intelligence Systems: Cloud-Native, Edge-AI, MLOps, and Governance for Real-World Deployment. Deep Science Publishing; 2025 Jul 28.
[20] Cheng Z, Jia X, Gao P, Wu S, Wang J. A framework for intelligent reliability centered maintenance analysis. Reliability engineering & system safety. 2008 Jun 1;93(6):806-14.
[21] Panda SP. Augmented and Virtual Reality in Intelligent Systems. Available at SSRN. 2021 Apr 16
[22] Moreno-Guerrero AJ, López-Belmonte J, Marín-Marín JA, Soler-Costa R. Scientific development of educational artificial intelligence in Web of Science. Future Internet. 2020 Jul 24;12(8):124.

[23] Chen L, Chen P, Lin Z. Artificial intelligence in education: A review. IEEE access. 2020 Apr 17;8:75264-78.
[24] Devedžić V. Web intelligence and artificial intelligence in education. Journal of Educational Technology & Society. 2004 Oct 1;7(4):29-39.
[25] Alam A, Alam S. Evolution of Artificial Intelligence in Revolutionising Web-Based and Online Intelligent Educational Systems. SGS-Engineering & Sciences. 2021 Sep 15;1(01).
[26] Maassen O, Fritsch S, Palm J, Deffge S, Kunze J, Marx G, Riedel M, Schuppert A, Bickenbach J. Future medical artificial intelligence application requirements and expectations of physicians in German university hospitals: web-based survey. Journal of medical Internet research. 2021 Mar 5;23(3):e26646.
[27] Almeida F, Simões J, Lopes S. Exploring the benefits of combining devops and agile. Future Internet. 2022 Feb 19;14(2):63.

Deep Science Publishing, 2025
https://doi.org/10.70593/978-93-7185-060-5

DeepScience
Open Access Books

Chapter 4: Artificial Intelligence for IT Operations (AIOps) and Observability in Next-Generation Infrastructure

1.Introduction

New technologies are constantly appearing that are trying to automate certain procedures in technology environments; this is known as AIOps. However, before delving into that area, it is important to understand what Observability involves. Observability addresses the problem of understanding how distributed systems behave and whether they experience any failures. It also helps determine if the business is operating well or poorly. Nevertheless, Observability encompasses many aspects and is not just one component.

2. Understanding AIOps

AIOps, short for Artificial Intelligence for IT Operations, is an area within IT operations that focuses on the application of data analysis to improve operations. As IT environments grow more complex, alert management without tools becomes tedious and inefficient. The growth in microservices and cloud native architectures makes it nearly impossible for human operators to monitor and identify the overall health and performance of an entire environment. AIOps employs a combination of big data and machine learning to automate and improve IT operational activities, including monitoring, improving availability, and performing root cause analysis [1-3]. To function effectively, it requires data ingestion and enrichment, including metrics, events, logs, network data, and performance data. The core components of AIOps consist of big data and analytics platforms, which encompass data ingestion, enrichment, correlation, analysis, and visualization, all connected to automation and remediation systems.

The main goal of AIOps is to deliver a better and faster response to anomalies and threats, thereby enhancing operational efficiency, increasing resilience, reducing service downtime, and improving user experience.

Observability is a cornerstone of AIOps. Software and infrastructure teams have long recognized that measuring and monitoring logs, metrics, and traces can provide insights into system health and performance; yet, despite the availability of numerous tools, many operations teams still lack full visibility into their cloud infrastructure. Metrics offer indicators of system health—whether a system is up and running and how it's performing—while logs provide detailed information about specific failure points. Together, they form the basic pillars of Observability. A well-implemented Observability process enables teams to quickly pinpoint root causes in the event of a system outage.

2.1. Definition of AIOps

IT business analysis AIOps (artificial intelligence for IT operations) is the use of big data and machine learning to automate identification and resolution of common information technology (IT) issues. The term, first put forth by Gartner, identifies multibillion-dollar AI-driven opportunities in the growing intersection of IT operations and big data. Also known as Algorithmic IT Operations or Algorithmic Ops, AIOps platforms—such as those by BigPanda and IBM—are growing in popularity due to their ability to ingest, correlate and analyze the ever-increasing volume, variety and velocity of data produced by modern IT infrastructures.

At present, low maturity organizations often rely on manual and repetitive work. Without the right tools for automating operations, handling spam-like alerts or enriching incidents, level 1 analysts can't focus on deeper analysis, resulting in low efficiency and long delays when reacting to changes. On the other hand, AIOps offers a data-driven approach to IT operations enabled by automation, artificial intelligence and machine learning. In all key operational areas—performance monitoring, anomaly detection, root cause analysis, incident clustering, notification routing and incident prediction—AIOps allows for proactivity and faster business actions.

2.2. Key Components of AIOps

A growing number of tools and services are geared towards automating IT operations, several of which are often described using the umbrella term "AIOps." Data ingress and egress are the foundation of any AIOps platform, so integration capabilities with a wide variety of tools (e.g., monitoring, alerting, log aggregation, service desk) are paramount. Integrations feed into an API gateway,

which handles data ingestion into configurable pipelines [2]. These pipelines are the heart of the application, processing incoming data with a variety of algorithms based on source, content, and time. The output of these pipelines can range from supplemental insights to the generation of new or the updating of existing tickets and alerts. More analytics then correlate and categorise incidents to better prioritise what gets sent to operators and engineers. Lastly, automation steps close the loop by addressing known problems in a proactive manner.

AIOps is an alluring proposition because it promises insights enhanced with the power of AI (often coupled with automation). Benefits include: having a common, single pane-of-glass view of diverse systems; finding and fixing incidents before they affect customers; discerning root causes in complex environments to speed mean time to recovery; and addressing the skills gap by improving how operations are performed.

2.3. Benefits of AIOps in IT Operations

There are several benefits of AIOps that can greatly improve IT operations. Being able to get in front of failures or potential failures before they actually occur is one of the biggest benefits. The system can detect vulnerabilities, notifying staff well before damage is done, thanks to the continuous monitoring of IT infrastructure and data analysis for anomalies.

Fast, efficient issue solving is another plus for sure. When something goes wrong, the system alerts the right folks, and details of which components have been impacted, fast. There are even some issues that known issues can be resolved automatically with. Through interconnected workflows with remote teams, the collaboration is improved, and service restoration is expedited. And the more automation included in the process, the more money can be saved by reducing rework and the time it takes to respond to incidents.

3. The Concept of Observability

Observability to IT operations teams is having all the data you need to understand the state of an application, system, or infrastructure, right now and in the past, to maintain resilience and react to changes as they are required. The data is often delivered as dashboards, reports or alerts in a cloud infrastructure view. Observability is all about telemetry: logs, metrics, and traces are the three pillars of the CVAD (Citrix Virtual App and Desktop) environment.

Log records are log statements regarding individual events or operations in application or enough infrastructure, and often are used to debug specific issues occurring in the system [2,4,5]. Metric records contain numerical values dependent on time, and they're generally used to identify trends or correlate events with spikes or drops in activity across an infrastructure stack. The Citrix Consulting team defines trace records as a collection of segments that represent the execution path taken by a request as it propagates through infrastructure services and components. Traces generally enable operators or developers to pinpoint bottlenecks, or other anomalies, that impact the normal operation of an application or service.

3.1. Definition of Observability

Observability is a measure of how well the internal states of the ITS are reflected by their external outputs. Ideally, changes in the internal state of the IT infrastructure (hardware or software) should be reflected by changes in its outputs or in other words, be externally observable. Obtaining data as close to real-time as possible is pertinent to a measure of observability in IT infrastructure.

The output of an ITS can be thought of as the set of investment outputs or key performance indicators (KPI), given the inputs or change requests to the ITS. Little or no change in the output, when change request volume is non-zero, implies an unhealthy system. This remaining information will be called "observability data". In complex systems, these are generally classified into three categories, namely Metrics, Logs, and Traces.

3.2. Importance of Observability in IT Systems

Observability is essential for the smooth delivery of services to customers. It provides visibility into an underlying system's health and allows one to react swiftly in case of any anomaly. From solid monitoring data, proactive alarms can be set to alert teams and help resolve incidents quickly. Logs, metrics, and traces enable the investigation of incidents and identification of root causes, facilitating faster recovery and incident closure. Observability also contributes to continuous service improvement by enabling anticipation and prevention of incidents, thereby improving the overall user experience.

The creation of microservices and the use of Kubernetes for deployment have increased the dependencies in the entire stack, which is spread across multiple geo locations in hybrid cloud environments [6-8]. A failure in any cross-dependent application or infrastructure service can have a cascading impact on the business. As action timescales become progressively shorter, the time taken to build effective monitoring solutions has significantly decreased. For example,

the notification time in SMS/IVR of an escalated call to field engineers is less than 10 minutes to enable rapid issue resolution. Therefore, advanced automation technologies are being deployed in different layers of the Observability platform, often referred to as AIOps.

3.3. Key Metrics for Observability

Observability is a critical part of maintaining the health and uptime of IT infrastructure from the data centre to the public cloud. It is made possible by three types of data — logs, metrics and traces.

Logs provide granular details needed to diagnose and resolve incidents, monitor expected responses of systems and applications, and optimize performance. Natively, cloud providers supply log data for each service via interfaces such as Amazon CloudWatch Logs, Microsoft Azure Monitor, or Google Cloud Operations. In addition, logs can be gathered from operating systems, containers, databases, hypervisors and edge devices. Metrics offer context for associated logs and help identify that an incident has occurred in the first place. Prometheus Metrics offers an open-source source of metrics. Traces reveal the path of requests inside fluid and distributed application programming interfaces. They are made available through the OpenTelemetry framework.

4. The Intersection of AIOps and Observability

The parallel usage of the terms AIOps and Observability in discussions of automation and monitoring within complex IT infrastructure and applications has become apparent. A comparison clarifies their relationship and reveals how they interconnect to optimize modern IT infrastructure. AIOps (Artificial Intelligence for IT Operations) consolidates big data and machine learning functionalities to enhance every aspect of IT operations, from availability and performance to capacity, change control, configuration, vulnerability, remediation, and even security. By collecting large volumes of data from multiple sources, it allows humans to manage infrastructure proactively rather than reactively, identify direct and root causes of issues quickly, predict emerging problems, and automate responses and remediation. Observability helps measure the health of a complex system via data indicators such as logs, metrics, traces, and potentially others [9,10]. The rapidly evolving operations practices of microservices, containerization, and migration toward a multi-cloud infrastructure are driving the need for increased Observability.

Application of AIOps to Observability data can surface meaningful insights about system status and performance that lead to higher resilience and agility."Error budgets" provide a notable example. Establishing an error budget as a performance objective for a service, then testing an Observability system against that objective, allows that system to advance from mere data collection and correlation. To meet the objective, the system must exercise artificial intelligence to identify patterns and causes of anomalous behavior and predictions of future conditions. In addition, it must engage automation in creating a swift, deep link to business continuity whether that is a remediating or restorative action or merely a notification.

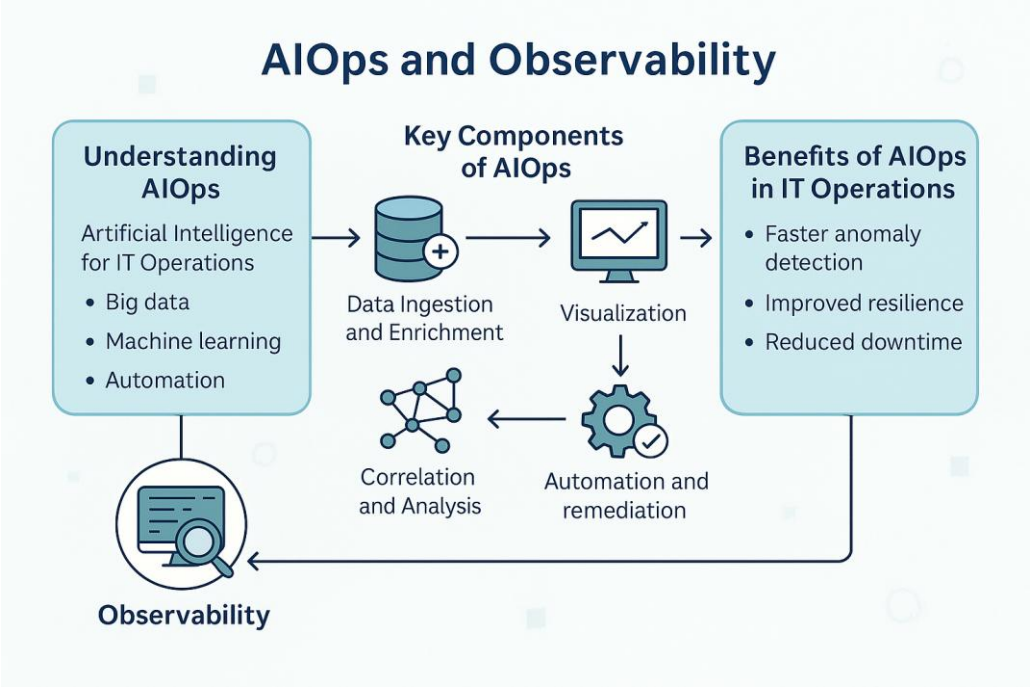

Fig 1. AIOps and observability

4.1. How AIOps Enhances Observability

The Integration of Artificial Intelligence for IT Operations (AIOps) with observability strengthens the depth of IT system monitoring while extending the benefits of dashboarding and alerting into proactive, automatic incident remediation. Observability data, especially logs, metrics, and traces, generate powerful signals that can be funneled into AIOps tools to swiftly pinpoint the root cause of system failures and enable direct business responses. AIOps in turn enhances observability by enriching these metrics and uncovering new data points in the signal-to-noise battle.

Although AIOps and observability are distinct, their paths have intertwined in the shared quest for resilient infrastructure and cloud architecture. Each leverages dashboards, alerts, and sets of labeled metrics to represent a view of system health, albeit on different time frames and with vastly different objectives [11-13]. Cloud infrastructure teams leverage AIOps to eliminate toil and track down toil and alert fatigue—issues that are not tied to a path and some people would rather not follow the AIOps journey. Meanwhile, observability acts as the watchful guardian, looking for anomalies in your infrastructure or cloud architecture. AIOps and observability together form the backbone of a proactive IT operations approach for the modern era.

4.2. Use Cases of AIOps and Observability Together

The manual operation and maintenance become more complex and resource-consuming, and are not only error-prone but also time-consuming. To combat this problem, AIOps looks to apply the insight provided by machine learning and big data analytics to enhance IT operations efficiency and minimize human interference. AIOps analysis processes depend on rich system data from execution and lifecycle management. But the collection of accurate data from such dynamic systems, and its effective utilization, is still a challenge. The concept of observability plays a central role, which means that a system can be understood with the information outside of it, notably in terms of its internals states, from the point of view of outputs it produces. The three primary signals – metrics, logs, and traces – are crucial sources of truth to evaluate and maintain the health of the system, and AIOps is the umbrella term for the techniques use that data to assist developers and operators in their job.

In the complex world of infrastructure, organically, humans can’t ever be expected to notice every tiny change taking place in your system [2,14-17]. Therefore, automation is a game changer at these turning points. There are a handful of real-world examples that show how AIOps and Observability marry together and enable automation where the system becomes more resilient and agile. These cases illuminate the functions of each domain and the strength of their joint use.

5. Challenges in Implementing AIOps and Observability

Yet, despite the rapid adoption of AIOps platforms, organisations are not realising the full value from automation, and the use of AI and ML capabilities. Here are some of the reasons cited by survey respondent for the underuse. The first is that operations teams generally are not skilled in artificial intelligence or machine learning and would need to get education and training, which would inhibit the adoption of the platform. Second, organizational cultural resistance to new automation and AIOps practices is causing organizations to drag their feet when it comes to deploying these tools. Third, the abundance of monitoring tools creates silos of data across the IT operations team, resulting in an incomplete view of the entire IT environment. Lastly, the sheer volume of data generated through monitoring activities overwhelms teams, leaving them unsure of where to direct their monitoring efforts.

AIOps platform adoption leads to the accumulation of vast amounts of data that must be analyzed to be useful. To efficiently monitor IT infrastructure, it is essential to establish sufficient observability into the environment by instrumenting the system under observation to surface its current state and health. Such instrumentation enables real-time status queries and provides a comprehensive view of ongoing operations. Observability deserves special emphasis due to the increasing complexity of infrastructure and software components, particularly with the widespread adoption of public cloud services.

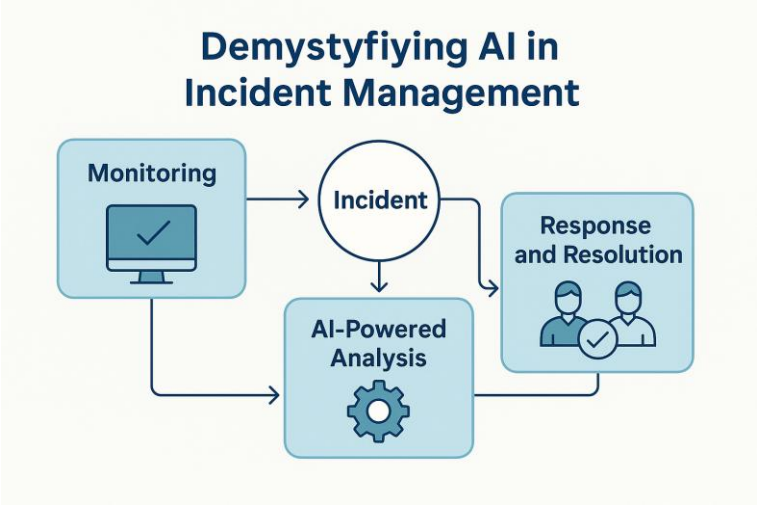

Fig 2. AIOps and Observability

5.1. Data Silos and Integration Issues

Culture is the biggest roadblock for successful implementation of AIOps. Siloed organizational structures, resistance to change, and lack of management buy-in can prevent teams from leveraging AIOps to its full potential. However, technical limitations also influence AIOps adoption. AIOps relies on data ingest—large volumes of historical and real-time data stored in data lakes or data stables, along with metadata such as details about team ownership of the metrics. This data must be accessible for use by AIOps systems. The data quality impacts the insight quality; thus, the data must be accurate and timely. Data facilities must have an API support model. The Parameters to Be Carved Out From Use Case for Selecting Any AIOps Platform Utilities discussion also covered data quality and integration.

There is a shortage of professionals with skill sets encompassing data analytics, application development, and infrastructure support, which is critical for successful AIOps adoption. Implementing best practices can streamline AIOps adoption in organizations. These practices include defining clear objectives, deploying chosen use cases, identifying and mapping skills, piloting the initiative, establishing governance, collaborating with business owners, sharing successes, and employing DevOps Continuous Integration/Continuous Delivery (CI/CD) for continuous improvement. The utilities best practices section elaborates on these aspects.

5.2. Cultural Resistance to Change

Research reveals several challenges encountered during the pursuit of AIOps, corroborated by professional observations. The assimilation of new technologies invariably demands time, requiring superior skills to manage the associated workload. The passage of time alone contributes most to recognition, whereas the search for the right talent brings about the most stress.

Employment Search Score (ESS) analysis indicates that queries on job-hunting portals often pertain to machine learning and AI, followed by subjects like graph theory, NLP, Python, and data engineering. Notwithstanding the influx of AI and ML professionals, a cultural dichotomy between AIOps and IT Operations frequently emerges [9,18-21]. Professionals from traditional IT Operations backgrounds may regard AIOps as an encroachment on their domain; conversely, those from AI/ML realms often voice concerns about the inactivity of IT Operations counterparts. The resolution lies in mutual conviction and upskilling to foster synergy. The lack of such mutual understanding leads to the formulation of discordant inquiries by either faction.

5.3. Skill Gaps in IT Teams

Despite having the right technology, a shortage of highly trained staff can hamper the execution of an AIOps strategy. It can be challenging for organizations to fill these positions, given that only qualified team members can validate the automated alerts and confirm whether the proposed solution of the platform will resolve the issue. Additionally, even if the team has the ability, the time to resolve and automate is often lacking. If team members are already burdened with managing traditional on-premises infrastructure, they may not be able to focus on designing and implementing new AIOps strategies. While building any new process, it is crucial to ensure there are adequate controls and checks in place. Involving the team in the entire integration process is the first step towards managing resource allocation. Providing personnel with ample time to learn new automation tools also helps efficiently combat the skill gap.

6. Best Practices for AIOps Implementation

Implementing AIOps should begin with clearly defined goals based on business requirements. This clarity helps justify investment in AIOps platforms and guides the selection of appropriate tools. AIOps tools should analyse all available telemetry data and integrate with the organization's existing ITSM and DevOps platforms to automate remediation of alerts and root cause analysis.

Recognizing that AIOps is an evolutionary process, organizations need to concentrate on continuous improvement. Deploying an AIOps platform is not an end in itself; rather, it establishes the foundation for an ongoing capability that improves over time. Nevertheless, certain challenges can impede successful AIOps implementation, including legacy and unsupported devices, organisational culture change, resistance to automation, deployment complexity, the absence of dedicated teams, insufficient quantity and quality of data, skill shortages in data science and IT operations, the impact of unaddressed root causes, and difficulties in integrating with existing monitoring systems [22,23].

6.1. Establishing Clear Objectives

Understanding AIOps details the critical but often underestimated process of clearly defining goals and objectives before implementing AIOps, covering aspects such as current state evaluation and success criteria. Cross-referencing with Challenges in Implementing AIOps and Observability further explores the complexities and strategies involved.

The first and most important step in the implementation of AIOps is to clearly define its purpose. Questions to consider include whether system health should be tied to a business metric or whether isolating a bottleneck in a particular stack area is the priority. Are metric, trace, or event analytics being sought? What is the current operational state? How does one want to leverage AIOps? Answering these will shape the final product. Some use cases involve fixing bugs in large enterprise architectures, while others focus on enhancing the observability of cloud or infrastructure. A well-defined goal allows for the allocation of the proper time, resources, and tools necessary to manage IT infrastructure most effectively.

6.2. Choosing the Right Tools

Choosing the right AIOps platform is fundamental. Clearly define goals before searching for solutions; a well-chosen AIOps platform should be so fitting that the decision to adopt it is easily justified [24-26]. Bridging the gap between legacy IT architecture and modern cloud environments requires opening data silos. Collected data must be enriched, correlated, and contextualized, enabling IT operations to perform root-cause analysis and evolve into a proactive support model.

Enriching data, closing the feedback loop, reducing toil, and implementing continuous improvement often necessitate specialized machine learning-based products tailored to specific use cases such as anomaly detection, root-cause analysis, noise reduction, or incident prediction. Merely automated data gathering is insufficient; otherwise, operations teams risk being inundated with events and alerts, leading to alarm fatigue and neglecting critical updates. Moreover, cultural shifts and a skills shortage among operations teams present additional barriers to adopting AIOps technologies in real-world scenarios.

6.3. Continuous Monitoring and Improvement

An AIOps system therefore provides continuous monitoring and improvement by always running manual tasks in the background as well as their automatic counterparts and checking whether the automatic executions performed in the past could have been improved by a manual alternative that happened later. After the initial data collection phase, the quality of automatic executions can be estimated by simulating automatic executions in the time intervals before each manual execution (thus avoiding data leakage) and then estimating decision correctness using one or more accepted metrics.

The ratio of manual to automatic executions is then used as a decision criterion in the following scenarios:

- If the ratio is low, it may no longer be worthwhile to look for automatic alternatives because humans are able to solve the problem quickly and decisively. - If the ratio is intermediate, humans and the automatic system may be trading-off decision correctness for decision speed. As a result, it can be profitable to remove the slow automatic alternatives that have not yet been removed. - If the ratio is high, numerous functionality gaps in automatic executions remain. A possible cause is that implementations of existing or new automatic features have not been comprehensive enough. Therefore, a profitable step is to focus on these implementations.

7. Case Studies

AIOps tools are designed to enhance operational efficiency in complex enterprise environments, where massive volumes of alerts can easily overwhelm teams. Case in point, IBM leveraged its Observability capabilities, integrated into Instana, to build an AIOps tool that aids operations and development teams in pinpointing the root causes of mission-critical problems and resolving issues before they impact business operations. This initiative enables the company to visualize the impact of events such as severe weather or the COVID-19 pandemic on its portfolio, thereby improving the accuracy of future forecasts and ensuring the resilience of its networks.

Another example concerns a major cloud provider that used an AIOps approach to help maintain the Observability of its infrastructure, services, and customers during sustained growth [24-26]. Though its existing pipeline-based Infrastructure as Code model supported a high degree of automation, the increasing complexity of code reviews introduced both inefficient resource use and management overhead. In response, the provider developed an AIOps-driven toolchain that applies established Observability insights to determine where code reviews are truly needed. The resulting reduction in code-review volume has translated into a direct decrease in the time required to deploy change requests.

7.1. Successful AIOps Implementation in a Large Enterprise

Monitoring and maintaining larger systems (Infrastructure Software in the example) is a difficult task. Any unexpected change can result in degraded software-product performance with diminished user experience. Also, the time to detect and remediate the incident using manual analysis is challenging and prone to errors. The AIOps platform is designed to triage infrastructure performance anomalies detected by traditional monitoring tools to reduce the

workload on IT Operations. The platform assists in identifying the probable root cause of the anomaly, reducing the remediation time and effort.

There is an incredible volume of observability data—logs, metrics, traces, events, and other data. With such exponential surge in amount of data, AIOps has become vital for multimodal complex IT Infrastructure. When combined with observability, a powerful AIOps capability allows quick resolution, tremendous automation opportunities, and superior customer experience. The key benefits I want out of AIOps now are around security posture, infrastructure resilience, automation, and cost efficiencies. The AIOps platform can accelerate tasks such as weekly SLO measurement reviews, provisioning cloud resources, dealing with capacity bottlenecks, and inspecting systems for PCI DSS compliance problems as well.

7.2. Observability Enhancements in Cloud Infrastructure

The capacity to oversee ever-more complicated cloud infrastructure is a critical aspect of digital transformation, helping to make the build of reliable and resilient systems. The ability to see into cloud infrastructure, particularly when coupled with automation tools like AIOps, allows organizations to quickly detect services issues [20-22].

AIOps, or algorithmic IT operations, incorporates advanced analytics and machine-learning technologies to identify and address complex IT infrastructure problems. The continuous evolution and growing complexity of enterprise IT infrastructure introduce operational challenges that can hinder digital transformation initiatives by reducing processing power, storage capacity, network performance, or becoming a single point of failure. The resulting issues—such as end-user impact, violated service-level agreements, non-compliance with monitoring requirements, security breaches, and loss of revenue—can lead to business outages.

8. Future Trends in AIOps and Observability

Artificial intelligence and machine learning are causal factors of the broader adoption of artificial intelligence for IT operations (AIOps), although the principal drivers are the increase in sheer data volumes, and the ever-growing complexity of IT infrastructures. Today, numerous roles within an organisation combine human and machine intelligence to proactively address operational

issues and other business needs. Three major themes reflect emerging trends in these technologies for the near future.

First, there is growing awareness of observability and its critical role in AIOps. Without sufficient observability into all components of the infrastructure, indeed the application itself, data science teams cannot build effective machine learning solutions for predictive analytics [27,28]. There is no single location for having logs, metrics and traces available at a high rate and low latency with query options for application/data scientists to explore insights. Today there is no vendor that provides a full suite of open source telemetry data platform components with alerts and insights engine to support these growing requirements.

8.1. Emerging Technologies and Innovations

Emerging technologies for AIOps are beginning to address the difficulties that have made early attempts at large-scale automation and AI difficult for operators to embrace. One such technology is observability, which involves making a system more monitorable, and generating logs, metrics, and traces to understand the system's status or diagnose problems. AIOps uses observability to automate routing, status display, and triage, improving problem management. Increasing the observability of cloud infrastructure helps operators understand new architectures and operating models, as well as improving management, governance, and compliance.

Effectively implementing AIOps requires a comprehensive monitoring strategy based in observability. Teams should understand that full observability is hard and expensive to achieve, and focus their approach on the areas that provide the most value to their business. Advancements in AI and machine learning are paving the way for new-generation AIOps toolsets, enabling operators to fully leverage automation capabilities.

8.2. The Role of Machine Learning and AI

Artificial intelligence (AI) and machine learning are driving substantial changes across various industries, ranging from manufacturing and healthcare to research and development. These technologies enable significant automation, leading to improved efficiency and productivity. Information technology operations (ITOps) are expected to undergo a similar transformation, with automation capable of scaling the achievements of existing professionals. The primary challenge lies in developing the software to realize this vision. The ever-increasing pace and complexity of modern business have led to digital transformation and the adoption of cloud-native infrastructure and applications

capable of responding rapidly to the evolving demands of customers and employees. While these applications and infrastructure are touted for their elasticity and adaptability, many enterprises have not yet leveraged them to their full potential. Moreover, few organizations are equipped to monitor and manage this complex environment whether on-premises data centres or cloud providers [19,29]. AIOps presents an opportunity to reimagine IT operations, enabling infrastructures and applications to detect and resolve problems without human involvement through the use of big data analytics and machine learning (ML). Besides monitoring applications and infrastructure, logging, metrics, and tracing now provide similar insights into processes, workflows, and business data, facilitating the identification of bottlenecks and areas for improvement.

AIOps is berthed on the foundational need for Observability. AIOps are the means of improving the Observability of an IT system or infrastructure by leveraging the large volume of Observability data captured via metric, logs and tracing from internal processes. These capabilities constitute the livelihood of an enterprise's IT chain: from the seamless usage of services and applications to the continuous monitoring of supply chains. Implementing SRE capabilities in these processes is thereby important in detecting, diagnosing, and remediating errors before they affect business continuity. Owing to the data-intensive nature of these services and the various moving parts involved, observability and AIOps play an increasingly pivotal role in fulfilling these business goals. This section explores the robotic-automation facet of AIOps, while the Telemetry facet is discussed upon next.

9. Conclusion

Modern IT infrastructure improvements are difficult without an understanding of how the application is performing, the user experience, resource availability, performance, and the latest security threats. It is a lost opportunity to not have 100% visibility, as AIOps and Observability can remove complexity, centralize data, improve time to root cause, reduce toil, automate, improve collaboration, and traffic shift to optimize experience.

AIOps leverages machine learning and analytics capabilities on the massive and diverse data produced by IT infrastructure and applications. Observability provides useful alerts, informs better decision making, and helps to automate and orchestrate such actions so that organizations can proactively identify future real and potential business-impacting issues and mitigate the risks much earlier.

References

[1] Lo D. Trustworthy and synergistic artificial intelligence for software engineering: Vision and roadmaps. In2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE) 2023 May 14 (pp. 69-85). IEEE.

[2] Battina DS. An intelligent devops platform research and design based on machine learning. training. 2019 Mar;6(3).

[3] Hannousse A, Yahiouche S. Towards benchmark datasets for machine learning based website phishing detection: An experimental study. Engineering Applications of Artificial Intelligence. 2021 Sep 1;104:104347.

[4] Divya S, Indumathi V, Ishwarya S, Priyasankari M, Devi SK. A self-diagnosis medical chatbot using artificial intelligence. Journal of Web Development and Web Designing. 2018 Apr 7;3(1):1-7.

[5] Bonner E, Lege R, Frazier E. Large Language Model-Based Artificial Intelligence in the Language Classroom: Practical Ideas for Teaching. Teaching English with Technology. 2023;23(1):23-41.

[6] Rao AS, Vazquez JA. Identification of COVID-19 can be quicker through artificial intelligence framework using a mobile phone–based survey when cities and towns are under quarantine. Infection Control & Hospital Epidemiology. 2020 Jul;41(7):826-30.

[7] Dragičević T, Wheeler P, Blaabjerg F. Artificial intelligence aided automated design for reliability of power electronic systems. IEEE Transactions on Power Electronics. 2018 Dec 20;34(8):7161-71.

[8] Fitsilis P, Tsoutsa P, Gerogiannis V. Industry 4.0: Required personnel competences. Industry 4.0. 2018;3(3):130-3.

[9] Panda S. Scalable Artificial Intelligence Systems: Cloud-Native, Edge-AI, MLOps, and Governance for Real-World Deployment. Deep Science Publishing; 2025 Jul 28. Mohapatra PS. Artificial Intelligence and Machine Learning for Test Engineers: Concepts in Software Quality Assurance. Intelligent Assurance: Artificial Intelligence-Powered Software Testing in the Modern Development Lifecycle. 2025 Jul 27:17.

[10] Erich FM, Amrit C, Daneva M. A qualitative study of DevOps usage in practice. Journal of software: Evolution and Process. 2017 Jun;29(6):e1885.

[11] Qumer Gill A, Loumish A, Riyat I, Han S. DevOps for information management systems. VINE Journal of Information and Knowledge Management Systems. 2018 Feb 12;48(1):122-39.

[12] Ugwueze VU, Chukwunweike JN. Continuous integration and deployment strategies for streamlined DevOps in software engineering and application delivery. Int J Comput Appl Technol Res. 2024;14(1):1-24.

[13] Brusilovsky P, Peylo C. Adaptive and intelligent web-based educational systems. International journal of artificial intelligence in education. 2003 May;13(2-4):159-72.

[14] Panda SP. Augmented and Virtual Reality in Intelligent Systems. Available at SSRN. 2021 Apr 16

[15] Moreno-Guerrero AJ, López-Belmonte J, Marín-Marín JA, Soler-Costa R. Scientific development of educational artificial intelligence in Web of Science. Future Internet. 2020 Jul 24;12(8):124.

[16] Chen L, Chen P, Lin Z. Artificial intelligence in education: A review. IEEE access. 2020 Apr 17;8:75264-78.
[17] Devedžić V. Web intelligence and artificial intelligence in education. Journal of Educational Technology & Society. 2004 Oct 1;7(4):29-39.
[18] Alam A, Alam S. Evolution of Artificial Intelligence in Revolutionising Web-Based and Online Intelligent Educational Systems. SGS-Engineering & Sciences. 2021 Sep 15;1(01).
[19] Maassen O, Fritsch S, Palm J, Deffge S, Kunze J, Marx G, Riedel M, Schuppert A, Bickenbach J. Future medical artificial intelligence application requirements and expectations of physicians in German university hospitals: web-based survey. Journal of medical Internet research. 2021 Mar 5;23(3):e26646.
[20] Panda SP. The Evolution and Defense Against Social Engineering and Phishing Attacks. International Journal of Science and Research (IJSR). 2025 Jan 1.
[21] Almeida F, Simões J, Lopes S. Exploring the benefits of combining devops and agile. Future Internet. 2022 Feb 19;14(2):63.
[22] Moayedi H, Mosallanezhad M, Rashid AS, Jusoh WA, Muazu MA. A systematic review and meta-analysis of artificial neural network application in geotechnical engineering: theory and applications. Neural Computing and Applications. 2020 Jan;32(2):495-518.
[23] Bello O, Holzmann J, Yaqoob T, Teodoriu C. Application of artificial intelligence methods in drilling system design and operations: a review of the state of the art. Journal of Artificial Intelligence and Soft Computing Research. 2015;5(2):121-39.
[24] Panda SP. Securing 5G Critical Interfaces: A Zero Trust Approach for Next-Generation Network Resilience. In2025 12th International Conference on Information Technology (ICIT) 2025 May 27 (pp. 141-146). IEEE.
[25] Kietzmann J, Paschen J, Treen E. Artificial intelligence in advertising: How marketers can leverage artificial intelligence along the consumer journey. Journal of Advertising Research. 2018 Sep 1;58(3):263-7.
[26] Panda SP. Augmented and Virtual Reality in Intelligent Systems. Available at SSRN. 2021 Apr 16
[27] Karamitsos I, Albarhami S, Apostolopoulos C. Applying DevOps practices of continuous automation for machine learning. Information. 2020 Jul 13;11(7):363.
[28] Swain P. The Artificial Intelligence and Machine Learning Blueprint: Foundations, Frameworks, and Real-World Applications. Deep Science Publishing; 2025 Aug 6.
[29] Brusilovsky P, Peylo C. Adaptive and intelligent web-based educational systems. International journal of artificial intelligence in education. 2003 May;13(2-4):159-72.

Deep Science Publishing, 2025
https://doi.org/10.70593/978-93-7185-060-5

DeepScience
Open Access Books

Chapter 5: Exploring the Role of AI in Enhancing Infrastructure as Code Practices and Optimization Techniques

1. Introduction

Infrastructure as Code (IaC) allows for defining and managing infrastructure configurations in a high-level declarative language. When used together with continuous integration and continuous delivery (CICD) pipelines, it can automate repetitive infrastructure management tasks, thereby advancing the implementation of the DevOps practice. The rapid adoption of IaC has also led to numerous research investigations aimed at optimizing the coding and execution of infrastructure code, including static analysis, automated testing, resource allocation, and integration within CICD pipelines.

Recent advances in artificial intelligence (AI) underscore its potential to automatically undertake activities that normally require human intelligence. While many AI applications target the automation of high-level tasks, there is substantial scope for exploiting AI at the foundational levels of infrastructure development. Thus, the integration of AI within the IaC lifecycle could enable engineers to leverage AI capabilities to create improved infrastructure. Moreover, the ability to associate AI intelligence with the infrastructure promises further innovations.

2. Overview of Infrastructure as Code

Infrastructure as Code (IaC) significantly contributes to the management of modern infrastructures. Advances in AI and machine learning are leading the way towards a new generation of AIOps toolsets that will provide operators with

much-needed automation. Key IaC concepts—introductory Automation, Configuration Management, and Orchestration—serve as the cornerstone for today's modern infrastructure.

The term Infrastructure mean writing script code that describes how to provision, configure and manage servers and computing infrastructure. Various technologies and cloud tools for managing infrastructures in the cloud have emerged supporting IaC [1]. This approach represents a paradigm shift in infrastructure management, incorporating ideas introduced by the DevOps movement, such as continuous integration, continuous delivery, and shifting left. Applying these DevOps principles to infrastructure and environment management helps organizations implement more reliable testing, accelerate the continuous integration and continuous delivery of applications, and, ultimately, deliver higher-quality software with greater frequency.

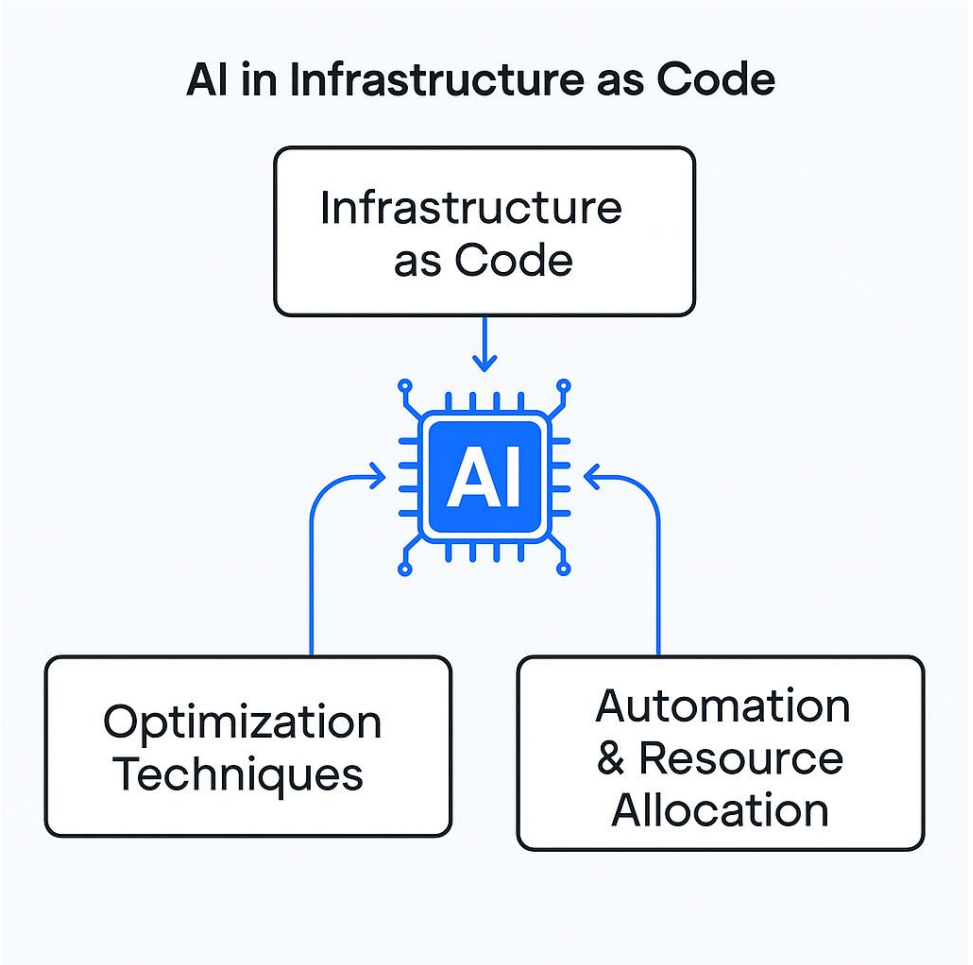

Fig 1. AI in Infrastructure as Code

3. The Evolution of Infrastructure Management

The demands placed on infrastructure management continue to increase. Many industries now rely on digital platforms for operations, customer engagements, and business performance. As demands grow, manual management of system and network resources becomes insufficient. Infrastructure management must adapt to support rapid growth and scaling.

Infrastructure as Code (IaC) has emerged as a solution, providing scripting capabilities and automation for infrastructure management. IaC offerings encompass companies such as GitHub, Atlassian, Puppet, Terraform, Ansible, Oracle, and Microsoft. These platforms enable users to define and manage infrastructure through code, facilitating automated deployment and configuration.

4. Artificial Intelligence: A Primer

Interest in artificial intelligence stems from its ability to mimic human cognitive functions. It provides machines with the capability of learning without explicit programming. It is an interdisciplinary science with multiple approaches, but advances in machine learning and deep learning are creating a paradigm shift in virtually every sector of the tech industry.

Among business applications, artificial intelligence is used for diagnosis, prediction, and classification. Particularly, operations management, the department involved in managing purchases, product creation, quality control, and distribution, can benefit significantly from artificial intelligence integration. Some argue that data scientists are the new consultants, utilizing data operations to streamline manual processes [1,2].

Numerous techniques such as genetic algorithms, Bayesian and Markov networks, decision trees, support vector machines, ensemble learning, swarm intelligence, and artificial immune systems demonstrate promising results. TensorFlow, developed by the Google Brain Team, has emerged as the default platform for artificial intelligence, machine learning, and deep learning applications.

5. Integrating AI with Infrastructure as Code

The growing importance of Infrastructure as Code (IaC), a practice that codifies infrastructure management into code, has led to an interest in enhancing its capabilities with Artificial Intelligence (AI) techniques. AI is beginning to permeate the IaC landscape, addressing the reliance on formalized instructions and potentially optimizing processes through learning from observed information. AI can transform IaC by introducing smarter code quality testing and enabling context-driven resource allocation.

The integration of AI in IaC promises to leverage the vast amount of data generated around IaC resources, code, and artifacts. Automating repetitive tasks and eliminating human error remain key enablers of IaC, and further assistance at the planning, designing, or development stage would enhance its effectiveness. Encouraging cybernetic collaboration, rather than dependency, between AI and humans at these stages could facilitate meaningful and effective use of AI. Such synergy paves the way toward the next generation of infrastructure as code, enabling more efficient and adaptive infrastructure management.

5.1. Benefits of AI Integration

The complexity inherent in distributed infrastructure operations has been exacerbated by escalating business demands. AI will allow processes to be stream lined through automation beginning at the core of infrastructure. When Machine Learning-driven models represent infrastructure state and Reinforcement Learning creates closed control loops to do (invoke) what no human hand should do and implement AI algorithms, methods will re-define the approach to managing infrastructure.

The combination of Infrastructure as Code practices with Artificial Intelligence can automate testing and validation, automatically produce dynamic resource allocation that best serves the operation, design continuous integration and deployment pipelines, supercharge continuous compliance and more.

5.2. Challenges in Integration

Although the foregoing integration with AI in Infrastructure as Code has many advantages, several issues need to be solved for successful development. One of the main challenges AI might contribute to infrastructure is the “added complexity it brings to managing infrastructure”. AI-Predicted Automated Changes that were not appropriately annotated with the true state of the infrastructure ($f\not\in$), resulting in a potential for misinterpretation. Transparency in such systems is thus essential.

The groundwork of AI technologies and tools, is also another relevant struggle. Understanding what AI can and cannot do is a precondition for getting the most out of it [3-5]. Additionally, this field is always expanding and thus current concepts may not be sufficient, which makes it hard to apply AI in existing workflows. Developers face a more demanding integration process, navigating around strong limitations of AI methodology and utilizing emergent tools and frameworks. Despite numerous efforts in research and engineering to overcome these limitations, they present significant obstacles to general acceptance.

6. Optimization Techniques in Infrastructure as Code

Validating Infrastructure as Code (IaC) scripts via static analysis is important to produce good quality IaC code. The auto-matization of IaC test and validation brings up the demand for AI to discover the defects. Dynamic resource allocation is another significant application of AI to IaC infrastructure management [6,7]. The core of the concept of dynamic resource allocation weak dot is calculating the values of variables according to certain states of the system. The main objective is to assign the suitable resources to the different projects at the appropriate time and to do so such that the costs are minimum and the services are maximum. Dynamic Resource Allocation in Infrastructure as Code is detailed via: (i) Allocation of Resource Demands; (ii) Optimum Resource Allocation

Infrastructure as Code optimization is an important technique used to reduce manual efforts in infrastructure management. Infrastructure as Code contains optimization techniques that reduce the cost and provide better utilization of resources. Infrastructure as Code optimization can be broadly classified into: (1)Static Analysis Optimization, and (2) Dynamic Resource Allocation.

6.1. Static Analysis and Code Quality

Static analysis tools can drastically improve the quality of Infrastructure as Code (IaC).Tools that evaluate syntax correctness, closely resembling lint checkers, have been developed for certain IaC frameworks, such as TFLint for Terraform and cfn_nag for CloudFormation. Quality of code goes above and beyond just getting syntax to work, but for IaC developers, there is a standard and a convention which exists around structures and configuration management which is required for stable and reliable deployment. Some recent studies have proposed static analysis tools that help find anti-patterns in Terraform IaC as a way to pinpoint places for improvement.

Developing a full compiler for Infrastructure as Code is quite a difficult challenge endeavor due to the interplay of various external entities—such as interfacing platforms, cloud services, and execution layers—as well as the side effects embedded within the scripts, including variable declarations and references. Nevertheless, continuing research efforts are apparently working towards a unified compiler grammar for Terraform IaC.

6.2. Dynamic Resource Allocation

Dynamic Resource Allocation As another example of optimization with Infrastructure as Code, dynamic resource allocation can be done. When an application in the cloud goes up, the infrastructure is spun up together. It's not an easy thing to figure out how much machines, how much memory or storage do you need — or, even worse, overprovisioning for the worst case scenario just to not fail due lack of resources [2,8-10]. With Infrastructure as Code, overprovisioning resources can be solved as resources can be allocated to the services once they are needed.

Like static analysis techniques, ascroar relies on machine learning models to determine the amount of resources that must be dynamically allocated for a deployed service. ascroar ensures business is running by auto balancing when an application needs a service with its resources availability.

7. AI-Driven Automation in Infrastructure Management

One of the places in which this partnership between AI and IaC is evident is infrastructure as code testing and validation. Infrastructure-as-code definitions allow for reliable and repeatable execution and deployment of services. The testing and validation of the code help ensure that the definitions produce the expected results and have fewer errors and vulnerabilities before being deployed. Infrastructure-as-code testing improves the stability and reliability of infrastructure configurations, while infrastructure-as-code validation guarantees the format, syntax, and semantics for the code are correct [1,11-12].

AI support for automated testing and validation of infrastructure-as-code definitions improves the reliability and stability of deployments. Incorporating AI in testing and validation enables quickly reminding of login credentials or configuration options, recognizing the mistakes and vulnerabilities found in the code, or worst-case suggesting the commands to create an infrastructure as desired. Another popular automation use case is using AI support directly in CICD (continuous integration and continuous delivery) pipelines. AI support in

CICD pipelines is responsible for recommending validation commands, providing vulnerability mitigation guidelines, implementing mitigations, and other aspects. The AI agent grounds on the state of the IAC definition repo — its current condition, recent changes, and previously identified problems and generated mitigations.

7.1. Automated Testing and Validation

The increased adoption of Infrastructure as Code (IaC) has made managing infrastructure easier and more efficient. However, challenges still exist that must be addressed to fully realize the potential benefits of IaC. One common issue is the inability to test the IaC code, which leaves the code unverified, unvalidated, and potentially having unknown quality. Also, inclusion of errors and the lack of accident, fault, and intrusion testing result in an error-prone environment. Automated testing and validation of IaC code help to avoid these issues and contribute to an assured IaC workflow [13-15].

AI Techniques such as natural language processing, computer vision, and even cross-functional neural networks can be incorporated to test, verify, and validate the code to a far greater extent than manual validation. The involvement of machine-learning techniques also allows the IaC code to be proactively checked against intrusive and malicious tests. The dynamic and predictive capabilities of AI techniques empower early detection of bugs and vulnerabilities in the IaC code. Many leading organizations are already incorporating AI techniques into their code-testing process, as it helps attain greater stage-level approval of the IaC code in their pipelines.

7.2. Continuous Integration and Deployment

The combined IAAS-CICD approach supports the software development life cycle by bridging the gap between the application and infrastructure. It helps elastically provision the infrastructure needed to support the application during critical phases, like system testing or onboarding a new product. This approach can also be automated, as a pipeline stage that is executed post-merge to continue the process toward production.

AI helps BotKube automate common infrastructure and pipeline issues that arise during a PR. The infrastructure required to support system testing, UAT, and production can be bought, monitored, and released through an auto pipeline using BotKube. It can check for configurational mistakes that would otherwise result in degraded application or infrastructure performance [16]. BotKube also allows integration with different AIOps systems to provide infrastructure sensing and

prediction to guide the IAAS-CICD approach toward proactive infrastructure provisioning.

8. Case Studies of AI Implementation

The advent of Infrastructure as Code (IaC) practices has irrevocably altered the way IT infrastructure is deployed and managed. The planet no longer relies on human operators manually installing and configuring servers anymore, but using code to declaratively, immutably, and idempotently perform these tasks. However, present practices still have their shortcomings. Static analysis of IaC can detect defects and security smells, automated testing and validation can make code deployments less error-prone, better resource allocation can reduce waste and inefficiency, and automation via Continuous Integration and Continuous Delivery (CICD) can improve deployment speed and reliability [16,17]. Such shortcomings can be addressed through the adoption of Artificial Intelligence (AI) into IaC.

The integration of AI within IaC practices promises a new age of automation-enabled efficiency and optimization. Nevertheless, complexity and integration challenges abound. Throughout history, the automation of infrastructure management tasks has continuously evolved and improved infrastructure technologies. A few examples where AI techniques and technologies have been utilized to ameliorate existing issues within IaC are presented, including the survey of AI application areas in Infrastructure Management.

8.1. Success Stories

Infrastructure as Code (IaC) has become indispensable for infrastructure management, and Artificial Intelligence (AI) fittings have started to appear within IaC. Prior sections examined the integration of AI with IaC and the optimization techniques enabled by it [12,18-20]. The discussion now turns to case studies that highlight the successes and lessons learned.

Automation can handle configuration burdens, ranging from coding speed to testing and validation. One significant advantage of automation is the ability to evaluate code before real-life application. Although numerous studies exist on the automation of IaC, resource optimization still relies heavily on manual input. Applying AI techniques to resource allocation can improve the efficiency of the Infrastructure layer. For example, AI and Machine Learning algorithms—and corresponding tools like Senlin, which focus specifically on clustering of resources—can be deployed within the operational environment to optimize and

readjust resource distribution. In this model, the AI-oriented tool functions as an agent of the Openstack platform [21-23]. Furthermore, small ad hoc open-source tools that leverage AI methods tend to remain experimental and rarely integrate with established Openstack components.

8.2. Lessons Learned

Domains seeking to leverage Artificial Intelligence enjoy a variety of suitable tools. At the other end of the scale, other domains recognize that advances in machine learning can be used to address some of their own long-standing problems. Infrastructure as Code for automated infrastructure management is one such domain gathering an increasing body of lessons learned from experiments with the integration of AI.

Infrastructure as Code practitioners have generally identified two main benefits from the introduction of AI into the infrastructure automation lifecycle. First, the automation of mundane or routine tasks. Second, better use of the information available within Infrastructure as Code. More recently the industrial control and optimization functions of AI have been used to assist with testing, validation and CICD automation. Other lessons learned include a marked increase in system complexity, a corresponding risk of losing the dynamism of Infrastructure as Code, difficulties in integrating AI into legacy systems, problems caused by bias in AI algorithms and data, and various security risks.

9. Future Trends in AI and Infrastructure as Code

Advancements in artificial intelligence technology and research have expanded its applications beyond traditional domains like computer vision and natural language processing. Today, emerging AI methods are influencing many other fields. Infrastructure as code (IaC), a DevOps practice that scripts infrastructure specifications, is among these fields. Generally, IaC scripts—across provisioning, configuration, and deployment domains—manage infrastructure infrastructure. Optimizing IaC can reduce business expenses and improve reliability, quality, and security. The large scale of infrastructure introduces challenges in cost, automation, and fabric quality.

Future trends in artificial intelligence methods applied to IaC therefore focus on addressing these optimization challenges. Static analysis techniques manage costs via cost model-based testing, dynamic resource allocation reduces costs by calculating the minimum resources needed for load handling, and process automation in infrastructure concentrates on automating testing and validation.

Such automation enhances the quality of continuous testing, continuous integration, and continuous deployment (CICD) pipelines, easing the testing of IaC scripts. Integration of smart technologies into IaC can further automate infrastructure control, monitoring, and maintenance, mitigating the constraints of manual operations and facilitating efficient, automatic management of real-time infrastructure service quality.

9.1. Predictive Analytics

Applying services in IT infrastructure is a process often associated with complex operations due to geographic locations, network distances, and other factors. It is a worldwide problem, especially for resource-constrained environments such as those presented at the border of a network where it is difficult to ensure sufficient resources to properly execute specific operations.

In IoT (Internet of Things) environments, the optimization of resource allocation is a recurring problem because, in some situations, conditions cause a constant change of devices, internal or external networks, and devices that support communication for IoT devices. For example: when the gateway of a network is changed, the new gateway can be totally different to the previous one, which can cause changes in the data-processing patterns.

9.2. Self-Healing Infrastructure

Self-healing infrastructure represents an important step in addressing the insatiable demand for cost and time savings while guaranteeing reliability. An AI-enabled self-healing infrastructure can reduce downtime, speed up issue identification, and take precise actions to fix the issues at hand. With systems growing larger and more complex, troubleshooting is becoming inefficient and slow; furthermore, missed or erroneous configurations enhance the exposure to attacks and malicious disruptions Self-healing mechanisms may prevent mis-configurations and improve resilience in proactive and reactive fashion [24,25].

In the field of IaC, attempts to create self-healing systems for misconfigurations and configuration errors have started. For example, we have already seen works that deals with automatic remediation of wrongly configured Ansible playbooks thanks to static analysis and model-checking tool driven testing. AI agents, working as optimization strategies, can be assimilated with any IaC pipeline. These AI agents may be used to scan the IaC files proactively to find misconfigurations and develop anomaly detection models for determining anomalous cloud resource configurations, and thus may further secure the CICD pipeline. It is also possible to extend such a superior pipeline to include models

for automatic testing and anomaly detection in the resulting application deployments, and make it part of a broader self-healing infrastructure ecosystem.

10. Ethical Considerations in AI Deployment

Artificial Intelligence: A game changer for infra at scale. But, AI has its own vulnerabilities and constraints that limit the efficacy of applying AI to IaC in a reliable and secure fashion. Model bias and unfair decision are key concerns, especially when decisions regarding resource allocation are at stake. For instance, an Agent may assign a lower amount of resources to a region to another according to an inadequate input data which does not consider the susceptible and exposed population in the entire regions. Moreover, next-generation Multi-Modal LLMs, e.g., ChatGPT, have also been demonstrated to be vulnerable to prompt injections, where an adversary is able to exploit prompt to control the model output [26-28].

As more critical applications are deploying AI, attention from regulators is also increasing. The best answer to both traceability and audit is really baked in from the get-go by following IaC practices. Roboticists should not turn a blind eye to ethical concerns and risks in the pursuit of powerful AI technology.

10.1. Bias and Fairness

The cutting-edge in Infrastructure as Code (IaC) is in embedding AI at the crossroad of software engineering and infrastructure operations. The synergy with AI provides a promising journey to address the current state of the art limitations of IaC and achieve its dream of self-managed intelligent infrastructure and thus become a natural ally to enhance IaC. Advances in AI have induced a paradigm shift that can revolutionize many disciplines, including software development and infrastructure management. The history of managing infrastructure is one of: “Gee, we really should automate that!” Infrastructure as Code extends these earlier efforts, which have taken form as Infrastructure as Code. Management of infrastructure directly faces issues as a result of the blow away-system-architecture diversification, for which recently available AI methods are a useful partner as well, both in the sense of boosting automation and overcoming IaC confines.

AI has demonstrated remarkable performance in automating tasks with minimal human intervention. There has been significant interest in applying AI techniques

to overcome IaC limitations and to bring automation to its next level. Current IaC state-of-the-art explores the intersection with AI, particularly where software development meets infrastructure management. These disciplines can mutually benefit from each other. For instance, in software development, AI techniques have been used to automate the generation of code changes related to fixes, enhancements, refactors; propose code reviews and learn from code reviews; fulfill user intent and simulate user behavior; do unit and integration testing; and improve performance. Infrastructure management can also benefit from AI-driven automation. However, inherent trends associated with AI systems, such as bias and fairness, need explicit attention when integrating AI with infrastructure as code.

Machine learning–based techniques are gaining interest in many disciplines, not without risks. Machine-learning (ML) techniques learn from input data to perform predictions. When ML is combined with IaC, the predictions will impact the code construction or the underlying infrastructure. Ultimately, if the data feeding the ML is skewed, such bias will be reflected in the predictions of the ML, and the bias will be propagated into the generated IaC. Early attempts at bias detection in IaC have highlighted the need for specific solutions because biased IaC can adversarially affect the good functioning of the systems it provisions. Consequently, ML in IaC needs to come with checks and controls that actively detect these potential areas of bias early enough to ensure fairness and justice [29-31].

10.2. Security Implications

Complex security concerns must be examined when automating software development methodology with an AI-focused orientation. Potential AI bias generates ethical considerations around algorithmic fairness and security implications created by adversarial machine learning.

The IAAC AI-based contemporary approach must consider sub-problems regarding bias. The first sub-problem examines bias within the IAAC PBR Model caused by biased AI in automated infrastructure code design, testing, and development. This aspect must ensure that AI-generated code is designed and implemented without having built-in ideological or semantic bias. The second sub-problem investigates the potential impacts on the PBR Model when external biased AI is used as supporting tooling during the various system stages. Adversarial machine learning within the IAAC PBR Model represents methods where harmful or malicious exploits or attacks can occur by way of performing IAAC PBR Model automated processes with known attacks, such as AI injection, poisoning, extraction, fraud, privacy, and evasion. A dedicated security

viewpoint model addresses these bias and adversarial machine learning-related security considerations.

11. Tools and Frameworks for AI in Infrastructure as Code

Artificial intelligence encompasses not only complex neural nets but also a vast array of ML, NLP, and CV algorithms. Huge proportions of the data generated each day are neither video nor audio but textual. Given the related field of natural language understanding, considerable developments in AI have been made around textual data. Components such as text classifiers, sentiment analyzers, and one-shot and zero-shot text generators offer tremendous potential to speed up and automate the work of infrastructure management. An example is Gradini, which leverages such components to simplify and improve infrastructure management, focusing on automation and optimization methods for Infrastructure as Code. Closely aligned with these automation and optimization goals, Gradini validates Infrastructure as Code templates using a natural language description of the desired state, integrating this semantic validation as a stage of the Continuous Integration/Continuous Delivery pipeline. A thorough exploration of how infrastructure management has evolved, the limits of current approaches, and how AI technologies can help addresses these challenges can be found in "Integrating AI with Infrastructure as Code." Powerful and linguistically similar services from large providers such as Amazon Comprehend and Google Natural Language are also available.

Additional works have demonstrated the usefulness of AI in automating the testing of Infrastructure as Code templates, particularly those expressed in the Kubernetes Deployment format. A different approach to optimization focuses on leveraging AI to optimize resource allocation in Kubernetes clusters. Although each of these works addresses different facets of Infrastructure as Code management, all share a common goal: the application of AI techniques and tools to Infrastructure as Code so as to make infrastructure management simpler, reduce mistakes, and curtail one of the major drains on the time and attention of development teams.

11.1. Popular AI Tools

Infrastructure as Code (IaC) describes the management of infrastructure through configuration files—blueprints defining infrastructure and serving as executable code—that require high quality, resembling application code. Artificial

Intelligence (AI) promises to help improve IaC-related processes by injecting smart decision-making capabilities, automating mundane tasks in testing and validation, enabling predictive analysis in Continuous Integration, Continuous Delivery and Deployment (CICD) pipelines, and enhancing resource utilization. Such improvements have the potential to contribute to cost savings, error reduction, and faster application releases.

Static analysis tools for infrastructure as code can analyze configuration files and automatically identify quality issues, potentially preventing erroneous code from being merged and deployed in production. Although the IaC community has produced several valuable open-source static code analyzers, these tools are primarily rule-based and alignment with static analysis tools for other programming languages is limited. Deeper analyses could be conducted to check the overall quality of IaC source code by combining NLP, code smell detection and ranking approaches based on historical data or with news stories (e.g., natural disasters) [3,32,3]. Automated scripts can monitor cloud platform prices and usage statistics and be triggered when the script detects that the infrastructure is over- or under-provisioned.

11.2. Emerging Technologies

Emerging technology focuses on section of AI presents a wide array of technologies and new methodological ideas that can be applied to increase the efficiency and security of programs written in isolated Turing-complete programming languages. The cross-pollination of ideas across different communities has substantially improved static detection mechanisms for bugs, bracket mismatch errors, statically-detected improper memory accesses, and common CVEs. Emerging technology also includes new approaches in infrastructure optimization, such as predictive analytics for forecasting demand, automated remediation by anticipating potential failures, and multi-cloud management for enhanced resilience.

Despite these advances, many integration and adoption challenges remain. For example, static detection tools typically provide only bug reports, whereas a substantial portion of emerging infrastructure focuses on mitigation in addition to detection. Recent exploratory efforts toward mitigation in other communities—such as automated fixing in the Java and JavaScript communities—offer valuable guidance. Within infrastructure-as-code security, emerging technology includes work on vulnerability repriming, recommendation engines, and the development of a unified testing framework. The breadth and diversity of the underlying solutions and problem contexts complicate the acquisition of hands-on experience for practitioners in this niche [16,17].

Furthermore, infrastructure optimization—including dynamic resource allocation—stands to benefit from numerous AI and reinforcement-learning techniques: game-theoretic modeling for resource allocation, reinforcement-learning-driven elasticity control of software-defined clouds, and reinforcement-learning-based dynamic resource scheduling. Although the body of literature remains narrow, recent pioneering studies applying reinforcement learning to performance-aware resource allocation and provision for virtualized environments indicate fertile ground for deeper exploration in dynamic resource allocation.

12. Best Practices for Implementing AI in Infrastructure as Code

Even though the technology behind Infrastructure as Code has visibly matured in recent years, important challenges must be addressed. The growing complexity of infrastructure templates, the high number of components to be integrated, and the availability of tools that assist in integrating AI concepts with Infrastructure as Code turn every deployment into a very complex task. An organization must consider all of these aspects when incorporating AI concepts into Infrastructure as Code. If there is no specialist engineer in AI integration, the advantages previously discussed can turn into a disadvantage because of setbacks in implementation and productivity. Therefore, following the recommendations presented ensures that the integration achieves the desired outcome.

First, specialized engineers must validate the resources that will be defined within the templates. Considering the availability of the cloud provider and minimizing the possible risks that some resource could have within it is essential. Not all cloud resources, even when deployed with sophisticated technology, present the same stability. Enriching deployment templates with information focused on business aspects reduces the number of risks. Second, when creating the templates, a topology must be defined to optimize the use of resources. Smarter definition of a resource topology automatically directly impacts the organization’s end savings. Finally, it is recommended to integrate the deployment templates with the CICD pipeline so they go through automation methodologies before being executed on the cloud infrastructure. A pipeline removes hundred percent of the infrastructure code from the hands of a developer so you can deploy more efficiently and more reliably.

13. Measuring the Impact of AI on Infrastructure Practices

The utilization of AI technology in infrastructure-as-code may enhance multiple dimensions in infrastructure provisioning and management such as security, reliability, resource supply and automation. However, realizing such improvements is far from straightforward, challenges arising include the use of state-of-the-art modelling methods and applying AI within already-existing technologies [11-13]. The relevance of these issues may be gauged in terms of well-defined key performance indicators (KPIs) such as those suggested in terms of reduction and automation and measuring the achievement of objectives laid down in the earlier discussions. Assessing several KPIs, this study covers and completes a 360-degree view on the added value intelligence brings to Infrastructure as Code. AI-based automation adds value by accelerating testing activities and reducing the probability of misconfigurations reaching production, as previously noted. Here, the right KPIs allow the impact of automation to be measured.

13.1. Key Performance Indicators

Tracking the development and implementation of Artificial Intelligence (AI) in Infrastructure as Code (IaC) workflows against stated objectives relies on the definition and monitoring of key performance indicators (KPIs). Two domains merit particular attention: automation and optimization. Metrics drawn from the literature include:

Automation of error prone analysis and consumption control for IaC workflows minimizes the threat of failure of weak IaC workflows and services. It also enables fast and accurate testing and validation of IaC and pipelines in planning and production environments with IaC code and pipeline predictors, allows for quick debugging issues in CICD pipelines, and helps us diagnose exactly where IaC code failed in the production workflow.

Automation of error-prone analysis and consumption control for IaC workflows mitigates failure risk for at-risk workflows and services [2,8,11]. It also provides quick, accurate test and validation of IaC code and pipeline in planning/production as well as takes part in fast finding of issues in CICD pipeline and accurate mapping of failures of IaC code in the production workflow.

13.2. ROI Analysis

Enabling AI capabilities in Infrastructure as Code (IaC) brings specific value propositions but requires significant financial effort. A robust business case is, therefore, critical to make the point for resource investment in AI. Defined KPIs help measure the impact of AI and justify the business case for investing, understanding what is critical about AI and its relevance for KPIs.

There are two types of KPIs to steer the evaluation. Product quality, customer satisfaction, time to market and staff utilization are all strategic KPIs. Operational KPIs Operational KPIs cover tests automation, nonproductive downtime ratio, and headcount drop estimation. In the end, the metrics we chose needed to serve as tangible proof points showing AI would improve infrastructure engineering.

14. Conclusion

Infrastructure as a Code (IaaC) is often seen as a fine grain pattern for setting up a infrastructure framework. It successfully distributes the resources to different endpoints, schedules the operation orders, and uses code for storing related information. The goal of IaC is to make common services (e.g., resource provisioning and scheduling) straightforward to implement. As infrastructure system management grows in complexity and with businesses in need of scale and the speed to market applications, the waste and shop floor byproducts of manual infrastructure management has been exposed. Therefore, infrastructure management intelligent automation hasbeen an irresistible trend.

The introduction of AI in IaC provides benefits such as more sophisticated translation and interpretation of business intent in natural language, smarter IaC code spawning, enhanced infrastructure resource optimization, and end-to-end effective IaC code testing and validation. These use cases can drive higher quality IaC code, the level of automation possible, resource efficiency, and time to market for releasing products. While this number serves to associate AI with a variety of threats, they do speak to the need for IAAS management to grow more automated. But there are also some caveats to bringing AI into IaC. AI models are intricate and have black-box like behaviour, which makes it challenging to handle outputs, and AI models are computationally intensive that might increase running costs. Moreover, the wide range of AI models represents diverse choices of AI models to integrate with IaC optimization, and bias in AI models is inevitable, which may lead to fairness issues. However, successful use of AI

applied to IaC has the opportunity to improve infrastructure management as we know it, and become a leader within the IAAS space.

References

[1] Divya S, Indumathi V, Ishwarya S, Priyasankari M, Devi SK. A self-diagnosis medical chatbot using artificial intelligence. Journal of Web Development and Web Designing. 2018 Apr 7;3(1):1-7.

[2] Bonner E, Lege R, Frazier E. Large Language Model-Based Artificial Intelligence in the Language Classroom: Practical Ideas for Teaching. Teaching English with Technology. 2023;23(1):23-41.

[3] Rao AS, Vazquez JA. Identification of COVID-19 can be quicker through artificial intelligence framework using a mobile phone–based survey when cities and towns are under quarantine. Infection Control & Hospital Epidemiology. 2020 Jul;41(7):826-30.

[4] Dragičević T, Wheeler P, Blaabjerg F. Artificial intelligence aided automated design for reliability of power electronic systems. IEEE Transactions on Power Electronics. 2018 Dec 20;34(8):7161-71.

[5] Fitsilis P, Tsoutsa P, Gerogiannis V. Industry 4.0: Required personnel competences. Industry 4.0. 2018;3(3):130-3.

[6] Panda S. Scalable Artificial Intelligence Systems: Cloud-Native, Edge-AI, MLOps, and Governance for Real-World Deployment. Deep Science Publishing; 2025 Jul 28. Mohapatra PS. Artificial Intelligence and Machine Learning for Test Engineers: Concepts in Software Quality Assurance. Intelligent Assurance: Artificial Intelligence-Powered Software Testing in the Modern Development Lifecycle. 2025 Jul 27:17.

[7] Erich FM, Amrit C, Daneva M. A qualitative study of DevOps usage in practice. Journal of software: Evolution and Process. 2017 Jun;29(6):e1885.

[8] Qumer Gill A, Loumish A, Riyat I, Han S. DevOps for information management systems. VINE Journal of Information and Knowledge Management Systems. 2018 Feb 12;48(1):122-39.

[9] Ugwueze VU, Chukwunweike JN. Continuous integration and deployment strategies for streamlined DevOps in software engineering and application delivery. Int J Comput Appl Technol Res. 2024;14(1):1-24.

[10] Panda SP. Securing 5G Critical Interfaces: A Zero Trust Approach for Next-Generation Network Resilience. In2025 12th International Conference on Information Technology (ICIT) 2025 May 27 (pp. 141-146). IEEE.

[11] Myllynen T, Kamau E, Mustapha SD, Babatunde GO, Collins A. Review of advances in AI-powered monitoring and diagnostics for CI/CD pipelines. International Journal of Multidisciplinary Research and Growth Evaluation. 2024 Jan;5(1):1119-30.

[12] Cheng Z, Jia X, Gao P, Wu S, Wang J. A framework for intelligent reliability centered maintenance analysis. Reliability engineering & system safety. 2008 Jun 1;93(6):806-14.

[13] Shivadekar S, Halem M, Yeah Y, Vibhute S. Edge AI cosmos blockchain distributed network for precise ablh detection. Multimedia tools and applications. 2024 Aug;83(27):69083-109.
[14] De Silva D, Alahakoon D. An artificial intelligence life cycle: From conception to production. Patterns. 2022 Jun 10;3(6).
[15] Battina DS. DevOps, a new approach to cloud development & testing. International Journal of Emerging Technologies and Innovative Research (www. jetir. org), ISSN. 2020 Aug 8:2349-5162.
[16] Maheshwari A. Digital transformation: Building intelligent enterprises. John Wiley & Sons; 2019 Sep 11.
[17] Devedžić V. Web intelligence and artificial intelligence in education. Journal of Educational Technology & Society. 2004 Oct 1;7(4):29-39.
[18] Alam A, Alam S. Evolution of Artificial Intelligence in Revolutionising Web-Based and Online Intelligent Educational Systems. SGS-Engineering & Sciences. 2021 Sep 15;1(01).
[19] Maassen O, Fritsch S, Palm J, Deffge S, Kunze J, Marx G, Riedel M, Schuppert A, Bickenbach J. Future medical artificial intelligence application requirements and expectations of physicians in German university hospitals: web-based survey. Journal of medical Internet research. 2021 Mar 5;23(3):e26646.
[20] Almeida F, Simões J, Lopes S. Exploring the benefits of combining devops and agile. Future Internet. 2022 Feb 19;14(2):63.
[21] Moayedi H, Mosallanezhad M, Rashid AS, Jusoh WA, Muazu MA. A systematic review and meta-analysis of artificial neural network application in geotechnical engineering: theory and applications. Neural Computing and Applications. 2020 Jan;32(2):495-518.
[22] Bello O, Holzmann J, Yaqoob T, Teodoriu C. Application of artificial intelligence methods in drilling system design and operations: a review of the state of the art. Journal of Artificial Intelligence and Soft Computing Research. 2015;5(2):121-39.
[23] Subramanya R, Sierla S, Vyatkin V. From DevOps to MLOps: Overview and application to electricity market forecasting. Applied Sciences. 2022 Sep 30;12(19):9851.
[24] Dang Y, Lin Q, Huang P. Aiops: real-world challenges and research innovations. In2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion) 2019 May 25 (pp. 4-5). IEEE.
[25] Liu J. Web Intelligence (WI): What makes wisdom web?. InIJCAI 2003 Aug 9 (Vol. 3, pp. 1596-1601).
[26] Maddox TM, Rumsfeld JS, Payne PR. Questions for artificial intelligence in health care. Jama. 2019 Jan 1;321(1):31-2.
[27] Battina DS. An intelligent devops platform research and design based on machine learning. training. 2019 Mar;6(3).
[28] Hannousse A, Yahiouche S. Towards benchmark datasets for machine learning based website phishing detection: An experimental study. Engineering Applications of Artificial Intelligence. 2021 Sep 1;104:104347.
[29] Verghese A, Shah NH, Harrington RA. What this computer needs is a physician: humanism and artificial intelligence. Jama. 2018 Jan 2;319(1):19-20.
[30] Rahwan I, Zablith F, Reed C. Laying the foundations for a world wide argument web. Artificial intelligence. 2007 Jul 1;171(10-15):897-921.

[31] Fensel D, Bussler C. Semantic web enabled web services. InAdvances in artificial intelligence: 25th Annual German Conference on AI 2002 Sep 4 (Vol. 2479, p. 316).

[32] Long E, Lin H, Liu Z, Wu X, Wang L, Jiang J, An Y, Lin Z, Li X, Chen J, Li J. An artificial intelligence platform for the multihospital collaborative management of congenital cataracts. Nature biomedical engineering. 2017 Jan 30;1(2):0024.

Deep Science Publishing, 2025
https://doi.org/10.70593/978-93-7185-060-5

DeepScience
Open Access Books

Chapter 6: Exploring ChatOps Integration with Autonomous Response Systems in AI-driven Incident Management

1 Introduction

The overarching aim of incident management is a rapid and reliable restoration of operational systems. Technical procedures and organizational processes—including the involvement of service providers and users—must therefore be both prepared and carried out effectively in the event of an incident. However, incident management tasks such as addressing Simple User Incidents and Service Requests frequently involve straightforward, repeating activities. In these cases, automatically resolving the incident and informing the customer independently offers significant advantages over manual team work. The introduction of Artificial Intelligence (AI) and Natural Language Processing (NLP) has gained increasing attention in operations for the fully automated resolution of simple user incidents and service requests.

One examined approach focuses on the interaction between the requestor in an AI-assisted incident management scenario and only partially automated service provider responders who decide whether or not to accept the system's AI-Suggestions. ChatOps, understood as a collaboration model that connects people, tools, process, and automation into a transparent workflow, is a reasonable instrument for this interaction. It enables centralized communication and operational convergence, supporting the transparent assignment, execution, and monitoring of actions and work items in chatbot-based autonomous response scenarios through corresponding message-routing.

2. Background of Incident Management

Strategic alert management plays a crucial role in service management. It enables a swift and effective response to alerts and tickets generated by an organization's monitoring system. With responsible staff on standby 24/7, interruptions can be mitigated promptly. The evolution of technologies supporting alert handling in recent years has introduced ChatOps, a conversational model that integrates chat with automation tools, thus enhancing operational communication and support. The growing adoption of automated response systems has also improved follow-up monitoring for issues [1,2]. Building upon these concepts, the integration of ChatOps and autonomous technologies offers real-time reaction capabilities and comprehensive communication with relevant users, thereby advancing the management of affairs. ChatOps and autonomous response systems have been implemented with a focus on artificial intelligence. Indeed, the development of an AI-enhanced incident execution system within ChatOps can empower the service desk to respond more rapidly.

2.1. Historical Overview

The term incident management is often associated with the current state of a collaborative response to an unplanned event that has caused a service disruption or a reduction in the quality of a service. However, the modern understanding of incident management has evolved along with the industry that it supports. In a business context, incidents and their related procedures are tightly connected with the IT infrastructure whose operation is essential to the organization. As a consequence, the growth of the IT infrastructure directly affects the incident procedures.

While previous iterations of Incident Management incorporated ticketing procedures, they often lacked true automation for incident verification and resolution. Moreover, there was no standardized procedure for the detection and of communication (such as sending real-time alerts over Slack) to the relevant responders when an incident occurred. Currently, with the advancement of AI and robotic process automation, it has become possible to develop systems capable of acting autonomously based on a set of pre-defined playbooks (or runbooks); such systems are known as Autonomous Response Systems. Thanks to a closed-loop design, these systems are capable of continuously assessing the overall impact of a situation by receiving feedback at each step of the incident response process.

2.2. Current Trends

The capability of response systems to assist users in handling incidents has become a central issue for many organisations. Investigations in recent years have focused on artificial intelligence-based systems capable of suggesting, through the analysis of historical data, the next most appropriate action for addressing an incident [1,2]. Addressing this challenge involves defining methodologies that enable an integrated response system to receive the next recommended response and forward it to the organisation's communication channel, thus facilitating its execution.

ChatOps represents a relatively new framework that combines collaboration with practical hands-on work. Beyond promoting conversations, it enables the execution of tasks without the need to leave the chat environment. Important classes of ChatOps bots act as assistants by automating specific processes and eliminating common obstacles and errors. Some bots offer a way to regulate discussions, ensuring topics remain within noise boundaries; others provide particular utilities such as accessing relevant data; and some execute tasks that would otherwise require the chat participants to leave the environment. The role of conversation bots is steadily evolving towards becoming an autonomous response system.

3. Understanding ChatOps

ChatOps is the use of chat tools to facilitate teamwork and communications between humans and virtual assistants. It helps users identify problems, access relevant diagnostic data, and implement effective responses. ChatOps integrates short-term contextual information such as support tickets and discussion-channel conversations with longer-term information in knowledge bases and runbooks, providing a shared platform for analysis and comment without requiring participants to possess technical expertise [2].

Some of the benefits of ChatOps include Minimizing context switches while keeping chat participants aware of the actions being taken towards resolution. Leveraging automation capabilities of virtual assistants to provide data and to perform routine actions, in order to free-up human telephony-channel support resources. Building a shared repository of information and communication for audit, accessibility and reusability purposes.

3.1. Definition and Principles

Incident management provides a comprehensive set of procedures and requirements that organizations must follow to address the consequences of an adverse event in their systems. The administrators in charge of such situations analyse the information provided and execute all necessary manual actions to restore the service [3-5]. The continually growing adoption of ChatOps has introduced collaboration and communication to the incident handling process, enabling response and restoration without leaving the conversation. Autonomous response systems are technologies that are capable of handling incidents and events on behalf of the service administrators. The integrated use of the two with ChatOps, enabled by ChatBots, is a productive way of blending communication, collaboration, automation, reaction, and restoration in the pursuit of incident management which requires minimal or no operator involvement.

Incident tickets are difficult to manage and track, so explicit instructions are required regarding what the problem is, what is being done, and the next steps. Why is this bad: No information leaves providers to investigate and analyze the current status and state of the ticket, this lengthens resolution time. ChatOps can be a successful collaboration and automation tool when working an incident ticket. In ChatOps, the ticketing tool is connected to chatbots and system bots. These interactive chatbots can to analyze natural language queries or command style ticket and then collect data from the relevant system and displays them in the ticket window. In addition to the frequently asked questions, the chatbots can be configured for self-service events of an organization. Along with self-service commands and automated queries, the chatbots also send notifications or updates to the users via chat platforms about ticket status, thereby automating communication and improving the user experience.

3.2. Benefits of ChatOps

The benefits of ChatOps during the incident response process are apparent. emphasizes that the ChatOps approach allows organizations to reduce the cost and time duration of incident response by enabling collaboration between team members via a shared communication channel that can be simultaneously used for command execution and automation. Furthermore, common information such as product documentation or root cause analysis, historians, and so forth can be directly added to the communication channel for quick access. Yoshida reinforces that ChatOps enables organizations to perform tasks faster by automating repetitive tasks and managing operations in a centralized environment in which human operators and chatbots collaborate. Moreover, collaboration and communication through a chat channel help create an inclusive

environment in which everyone's expertise can be readily shared. Kwiatkowski confirms that the automation of routine and repetitive tasks is one of the driving forces for embedding the ChatOps approach in most incident management workplaces. Automated actions are less error-prone and are able to execute commands with fewer delays, both of which decrease reaction time under stressful conditions. The visibility and shared communication channel created by ChatOps keep all of the teams informed [2,6]. Sentell et al stress that ChatOps is "an approach to running operations and support that emphasizes collaboration and communication." The execution of operations commands through a chat-based channel enhances transparency, accountability, and traceability through chat logs and message histories. Jankowski prioritizes the benefits of ChatOps as "collaboration, speed, traceability and sharing." ChatOps fosters communication and cooperative problem solving in an easy-to-use manner, while concurrently providing rapid response capabilities by allowing participants to act in a common workplace. Command execution through a chat platform enables traceability and auditability of performed actions because the chat archive maintains a record of the request that initiated a given action, the responses produced, and any associated status updates. ChatOps also makes it easier for technicians to share their knowledge and experiences by openly incorporating comments and informational links into the communication channel.

4. Autonomous Response Systems

Although automation can deliver operational tasks, it frequently requires a human as a decision-making or validation component in the incident-management process. An automated, semi-automated, and human-centric incident-management system can support incident identification, the gathering of incident evidence, and in some cases, provide potential resolutions. However, these actions require human validation and execution. An autonomous response system can enable fast mitigation of incidents; in some cases, humans may choose to override the actions of an autonomous system to prevent a major outage or a security breach.

Autonomous response systems are an emerging category of automation that harness fault prevention measures, intelligent observations, and logic-based remediation—all in real-time [7-9]. They translate remediation plans into executable playbook scripts, expediting actions that lessen recovery time, cost, and impact. Moreover, response systems can behave autonomously and apply different technologies, e.g., machine learning, natural language processing,

artificial *intelligence, Robotic Process Automation (RPA), Computerised Maintenance Management Systems (CMMS), Digital Twins, ChatOps to cover weaknesses. Their primary function within an incident system is to perform key operational tasks automatically, in order to mitigate system faults or security breaches—thereby significantly reducing the response time for incident resolution.

4.1. Overview of Autonomous Systems

Incident management comprises identification, analysis and correction to avoid the same happening again. Lastest advances in generative AI enable the creation of smarter systems and tool to assist the incident management operations. One such advancement is the combination of ChatOps with an autonomous response system - enabling instantaneous, adaptive actions to cyber incidents.

Resolution of real-life incidents requires effective communication and cooperation across the incident lifecycle. ChatOps philosophy encourages the teams to work together and help the customer to know and show the priority of your request. Auto response is a system that auto-magically watches and takes action against events. The proposal of ChatOps by auto response is about how to expand the assistance proactively, by using ChatOps as a way to interact in real time and taking decisions in real time.

4.2. Key Technologies

Operational management concepts are frequently borrowed to benefit the development of intelligent autonomous systems, such as Capitalizing on the existing knowledge of project management and job scheduling for building intelligent autonomous project management tools. Classification algorithms play a key role in autonomous systems, providing support for knowledge reasoning and decision-making capabilities. In Reinforcement Learning (RL) methods the system maximizes a reward for itself by taking decisions that the IL agent affects the environment, which has recently been used in autonomous systems. In particular, Deep Reinforcement Learning (DRL)—the integration of RL and DNN—has been achieving remarkable success in different areas such as robotics, video games, and self-driving cars.

An autonomous system for IT service desks can rely on Logical Reasoning. For instance, anySupport determines the most suitable department for solving service desk requests by classifying customers' messages using text classifiers and applying logic rules. Moreover, techniques from Virtual Assistants can help in the development of autonomous systems to support the communication with users. PersonaChat aims to provide a virtual chatting partner interacting in natural

language. The conversation of a Virtual Assistant tends to be more oriented to casual dialogue instead of goal-oriented dialogues. Some techniques may also enhance the user experience of the chatbot. The system developed in presents the responses to users as cards, providing multiple suggestions for follow-up questions, and allowing users to give positive and negative feedback to improve the system.

5. Integration of ChatOps and Autonomous Systems

In an increasingly interconnected world, the demand for efficient and rapid responses to service incidents is growing. To meet this demand, autonomous response systems (ARS) capable of making independent decisions are becoming a reality [10]. These systems perform preventive or mitigating actions against incidents in a closed-loop manner by leveraging methods such as artificial intelligence or rule engines. Network management applications such as alarm correlation, fault detection and diagnosis, and traffic engineering help ARS achieve their objectives. Leveraging the benefits of ARS requires rapid deployment, and their integration with existing chat-based collaboration systems is a logical choice.

ChatOps uses chatbots and chat-based collaboration systems to achieve various tasks, thus enhancing communication and automation. In ChatOps, a chatbot observes communication and can take further actions independently to achieve shared goals. The combination of ChatOps and ARS presents several challenges that must be addressed to achieve a successful integration. A practical integration framework is described, supported by three recent real-world use cases, one of which is an enhancement of an already deployed chatbot with autonomous workflows for AI-based incident management. A case study illustrates an implementation that integrates three different Autonomous Response Generators to provide recommendations to an Incident Manager. Additionally, a performance feature that monitors the acceptance and rejection rate of recommendations is described, along with a feature that analyzes suggested actions to a recommendation to enable content-based feedback.

5.1. Framework for Integration

As organizations become more dependent on technology and automation, the number of system failures often increases and becomes harder to handle manually. To ensure reliable and stable services, incidents must be addressed

rapidly and effectively, making automated incident handling systems (AI Ops) essential. These systems automatically detect and resolve incidents in highly monitored environments, such as large-scale cloud environments that generate a massive number of alerts [10,11]. Autonomous response systems perform automatic responses, like auto-scaling, and utilize responses proposed by domain experts, such as executing commands mapped to detected symptoms. Teams manage and monitor production services using collaboration tools like Slack and Microsoft Teams, which connect remote workers regardless of their physical locations. By integrating ChatOps with an autonomous response system, real-time responses with a better user experience are achievable.

Today, company services often rely on IT infrastructure services within Cloud Providers or Web Service Providers. If any of these underlying services become unstable or unavailable, it can cause significant trouble to the company. Incident Management services, whether in-house or outsourced, are increasingly vital to reduce service downtime and ensure reliability in the digital economy. ChatOps is a method of interaction that involves conversations and commands inside a group chat. Commands issued from the chat interface serve to both interact with data and devices and open a window to continue reviewing operations within the chat. Its main benefit lies in allowing developers and operators to perform tasks without leaving their conversation window. Regardless of being a developer, security analyst, operator, or help-desk agent, working within a ChatOps setup means making work social, collaborative, transparent, and monitorable.

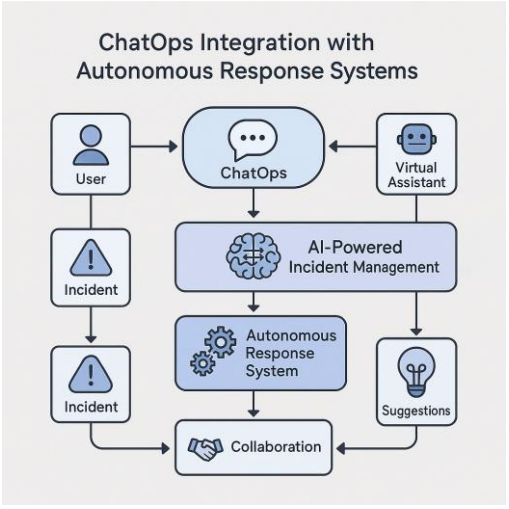

Fig 1. integrating ChatOps with an autonomous response system

5.2. Challenges and Solutions

Without careful planning and preparation, integrating ChatOps with an autonomous system could become a major pitfall. A number of potential challenges are identified here, together with solutions that have proved effective in practice.

The integration challenge. For human-machine interactions to operate at full efficiency, the ChatOps platform must be integrated with the autonomous systems that perform incident responses and remediation actions. When keeping users informed in real-time about their requests, the ChatOps platform should be linked to the notification mechanisms that the different autonomous systems use to inform the incident management team about actions being executed and their status—i.e., ongoing, succeeded, failed. These integration dependencies mean that the design principles identified above are not protected from the integration risk. When integrating ChatOps with an established ecosystem of autonomous systems, possible limitations in those autonomous systems might prevent full support of the APS design principles, especially the principle of a single human-machine communication channel for situational awareness during any incident. The principle of using ChatOps to establish and enforce sequence tasks with a pre-flight approval mechanism might also be affected by the capabilities that the underlying autonomous systems make available.

6. Case Studies

The integration of ChatOps with autonomous systems in incident management can be illustrated by the real-world example of IBM's Smarter Cities initiative. The autonomous system Valael processes complex urban data streams in real time and autonomously responds to incidents—such as cleaning up a methane leak—by orchestrating stakeholders, ranging from citizens to government agencies and private organizations [12-14]. However, there is a disconnect in the way communication between stakeholders is handled. Currently, Valael launches a Storm Web Chat—a channel with a chat history—and sends the participants a link that directs them to this channel. However, users do not desire to shift their conversation for every incident into a different, technical chatroom; many simply want to use their default communication channels, such as SMS or WhatsApp.

In this concrete scenario, the integration of autonomous response systems with ChatOps becomes evident. Valael would not only initiate real-time incident responses but also manage and enhance the communication process, reaching

users on their preferred platforms and offering easy access to all relevant information, data, and tools required for incident handling. In this sense, Valael would be the engine of Smarter City's Incident Management, while ChatOps would form its gearbox and steering wheel—enabling it to communicate more naturally, effectively, and efficiently with the people involved.

6.1. Successful Implementations

Several organizations have successfully merged ChatOps with autonomous response systems for AI-driven incident management. A crucial factor for success is designing an information architecture capable of integrating and processing information from multiple source tools. When ChatOps is combined with autonomous response systems, users benefit from a unified access point that supports efficient information gathering and smart, automatic response capabilities.

The Cortex ecosystem exemplifies successful integration, automating the runbook engine with a set of Cortex Responders—scripts that block file hashes, restrict IP addresses, and conduct other remediation actions with predefined access privileges. Connectivity to Slack is also established. During an emergency, the responder engine executes smart actions that block an attacker's commands in real time. Prompt escalation of informative chat messages enables the security team to collaborate on the incident and initiate further responses.

6.2. Lessons Learned

Two examples with different levels of maturity provide a glimpse into the lessons learned [12-14]. A monitoring system for highly available services is implemented with autonomous responses. Incident-related data are transferred to an incident-management platform and automatically processed. Upon incident escalation, a communication channel is opened and filled with incident-related information, e.g., a summary, links to dashboards and data. However, the channel is not used for the communication itself since it is not integrated in the users' processes. Users instead continue to use direct communication channels (e.g., phone, or a tool designed specifically for chat) because that was the practice before implementation. Creating the communication channel automatically does not provide any value to the process.

The same functionalities are implemented in a highly available file-based storage. The system is mature, can communicate effectively and maintains itself when failing. The time-to-repair metric is the most important one and is reduced with an autonomous response significantly—not only for severity 1, but also for incidents with lower severity (the system is more stable, fails more rarely, but

also recovers more quickly). The system uses a communication channel as a single source of truth for the response. The complete communication sent to the channel is preserved as documentation and all information necessary for the operational staff is directly accessible for the incident owner in the channel—the owner does not have to seek information in other sources.

Based on these two examples, the following lessons can be formulated:

– Focus on the incident-response communication when designing an autonomous response. Creating a communication channel automatically provides added value only when the channel becomes an integral part of the user processes.

– Aim to make incident-related communication the single source of truth for the response.

7. Impact on Incident Management

Integrating ChatOps with autonomous response systems within AI-driven incident management offers a blend of operational efficiency and enhanced user experience. ChatOps provides the infrastructure for coordinating incident management processes and for sharing information across cross-functional teams [3,15-17]. These advantages have practically supported the development of self-reliant response mechanisms in the form of policies that are enforced on-the-fly, connecting actions to detections.

A pragmatic backbone for integration of that approach covers challenges of automation needs, effective communication, support for tool integration and relief of team members' cognitive load. Stories from the frontlines allow for successful case studies and the extraction of key lessons. These increased benefits in terms of operational efficiency and user experience support the justifications

for integrating autonomous response techniques into ChatOps.

7.1. Efficiency Improvements

ChatOps coupled with autonomous response mechanisms results in substantial increases in efficiency for operators, as it capitalizes upon the automation technology. By consolidating tools and invoking all operations from a single, ChatOps interface, there is no longer any need for handoff between products and tools and IT Ops can be very efficient and relatively error free. In addition, ChatOps, supports broad automation and orchestration, and allows creators to create and connect offers, automate routine workflow concerns, and reduce

governance and control overhead. Operators get the newest automation items, add their own products to the platform, and can also easily find internal and external automation capabilities.

For users, the incorporation of ChatOps generates significant efficiency gains. Operators are able to document all standard investigations and resolutions in the platform, enabling users to address incidents independently. Moreover, the platform extends its functionalities beyond incident management, delivering assistance and elevation in all operational situations. Users appreciate the increased convenience afforded by performing operations conveniently on a daily basis, coupled with the assurance of continuous availability around the clock and the year.

7.2. User Experience Enhancements

Autonomous response systems have the potential to provide enormous benefits to incident management [18-20]. Several analyses have identified the impact that AI can have in these kinds of activities. One relevant top-level goal is the improvement of "efficiency and cost savings." On the other hand, ChatOps provides an easy way for the users to perform certain automated actions just by sending a message through a regular collaboration platform or chat application. In addition, ChatOps implementations frequently make use of capabilities exposed through an API, which, in the context of incident management, allows the incorporation of additional tools and technologies in the different phases of the incident lifecycle.

The integration of ChatOps and autonomous response systems allows combining the advantages of both approaches. From the operational perspective, the team members directly face the actions executed by the autonomous system and, at the same time, are able to execute those actions themselves when they prefer to do so. Furthermore, the automatic sending of information to the teams also leads to major improvements in terms of.

8. Future Directions

The future development of incident handling will be furthermore embedded in the system-wide application of artificial intelligence (AI). Consequently, a larger part of incident management will be handled autonomously—i.e., performed without human interaction. Autonomous systems will not only accelerate response but also enable 24/7 operations and a continuous reduction of the human workload. Nonetheless, completely autonomous intervention poses a high risk,

as incorrect responses may cause further damage. Hence, investigations dealing with a combined approach of ChatOps and automated incident handling are emerging [21-23]. Operations that cannot be handled autonomously can be handed over to the responsible teams using a dedicated collaboration tool, thereby minimizing response time and maintaining full control. Manual operations that cannot be accelerated or eliminated by the AI-driven incident handling system are well covered by ChatOps. Potential future work can derive a concept of how an autonomous response system can be integrated into a ChatOps environment.

Practical applications of future research in this field enable a combined approach that uses an AI-driven response system in close cooperation with ChatOps. By connecting ChatOps with an autonomous system, users take advantage of both strategies. The response system evaluates incoming alerts and executes defined operations automatically, whereas operations that cannot be handled independently are handed over to the responsible teams via ChatOps. Different challenges have to be overcome before such an implementation becomes possible, such as the technical integration of the AI-driven system and making the capabilities of the autonomous response system transparent to ChatOps users. Beyond practical concerns, several ethical considerations have to be addressed, particularly relating to data-privacy and accountability.

8.1. Emerging Trends

With a full range of incident management activities built upon autonomous response systems, ChatOps provides a communication environment that elevates all aspects of incident management toward artificial intelligence. Cricket, for example, is designed to process many IIoT-generated alerts with extremely low latency in the Incident Triage phase and then communicate with the resolution teams regarding suggested solutions and next steps.

ChatOps can be combined with Cricket to serve as an integrated supporting ecosystem behind the incident management communication. Business and real-time user experience are just two of many important considerations at every stage of incident management. The basic idea is to leverage ChatOps to enable a more connected experience at every stage of incident management, on every platform, and for every team. Organizing that effort within an autonomous response framework ensures that these user experience considerations are based on real-time signals.

8.2. Potential Research Areas

Emerging research on AI-driven incident management highlights the potential of integrating ChatOps with autonomous response systems. Autonomous response systems employ artificial intelligence, machine learning, and automation technologies to detect, assess, and respond to incidents without human intervention. This integration leverages the benefits of ChatOps, which facilitates collaboration, communication, and automation, to address the operational, technical, and user-related challenges of autonomous response systems [9,24,25]. A practical system engineering framework guides the execution of implementation projects.

Several perspectives suggest that combining ChatOps and autonomous response systems enhances the effectiveness of incident management. Case study research shares practical knowledge by examining the application of autonomous response functions and ChatOps interaction in an organizational context. Complementary action design research extends the domain–method matrix by anchoring the strategic alignment of the chatbot function to a real R&D challenge. Finally, exploratory interviews investigate the potential use of ChatOps to meet the needs of users and stakeholders affected by autonomous response functions, thereby contributing to a positive user experience and minimizing inconvenience.

9. Best Practices for Implementation

Implementation begins with a carefully prepared plan. Thorough documentation of the incidents or disaster scenarios that the system aims to handle is essential for clear requirements definition. Developing a prototype validates the design, and establishing a complete test environment supports comprehensive evaluation of performance and reliability, ensuring quality before deployment.

Deployment is a three-step sequence. Promotion of the incident management system lays the groundwork, followed by gradual rollout—starting with receptive users—to demonstrate benefits and build support. The final stage is full deployment to all target users. Post-deployment monitoring evaluates the system's quality and impact. Regular reviews—at least monthly during the initial period and quarterly thereafter—assess the outcomes against the original incident and disaster plans, enabling continuous improvement.

9.1. Planning and Strategy

ChatOps has become essential in AI-driven Incident Management. Planned integration of ChatOps collaboration tools with an Autonomous Response System will greatly affect operational efficiency. An Autonomous Response System initiates corrective actions automatically, and ChatOps enhances the dialogue between the communications tool and the Autonomous Response System. Solid planning and strategy can enable a successful ChatOps-Autonomous Response System integration.

Conceptual exploration sets out practices that a methods group may consider. Mapping the dialogue, sources, and recipients of data can indicate cross-domain, cross-team, and cross-border contacts. Control mechanisms ensure that only permitted data proceed to decision-making components. The process can distinguish between data sources of different security statuses and direct operations to keep confidential data protected [26-28]. Tracking the dialogue and maintaining an audit trail under ethical regulation protect user privacy and enable accountability.

9.2. Monitoring and Evaluation

Public services and essential infrastructure are often automated and rely heavily on cyber components. The continuous functioning of these systems is ensured by autonomous incident management systems. Artificial intelligence algorithms detect incidents quickly, select suitable countermeasures, and begin their automatic execution. Monitoring AI-driven incident handling decreases the operator's trust, improves the system by providing feedback, and helps resolve incidents more effectively. ChatOps provides a convenient way to communicate with operations tools and may also provide a simple interface for monitoring other processes.

A quality feedback loop helps autonomous response systems make fewer mistakes, avoid repeated failures, and increase trust. Monitoring may be executed by humans, artificial intelligence, or both. Humans may verify confirmed incidents and provide additional explanations to affected users or other interested parties. Artificial intelligence components automatically provide verification feedback by learning from previous incidents through historical data, user interactions, or both. As incident handling is often fully automated, taking immediate changes or other responses when the policies are violated (for example, when failing to fix an incident within a stipulated time). The AI or the human operators are informed of the detected nonfulfillment, and the required action is taken.

10. Ethical Considerations

From an ethics standpoint, integrating automatic responders into ChatOps raises a number of issues. Now as the conversation between passengers and drivers as well as decisions for how and when service operates are automated, questions of responsibility come to the fore. In case there are wrong behaviours and conflicts that are not well defined, we should also decide who ultimately is responsible for them: the human programmer (who may reason over them), the organization, the system, etc. Furthermore, the sensitive data that is dealt with when handling incidents is important that processes are put in place to prevent it becoming something it shouldn't be, or else used as something it isn't intended for [6,29-31].

Responsible Innovation and Ethics by Design can provide guiding principles and tools for tackling these ethical issues in autonomous response systems. In fact, as more autonomy is added, there is a need for people and organizations to have transparent interfaces that they can audit to verify system behaviours and ensure they are aligned with users' interests. While the development of AI-based incident management industry will only be further developed, but the ethical considerations should also be the essential part of its implementation and social adoption.

10.1. Data Privacy Issues

The application of ChatOps and AI-run response systems in incident management seems promising, yet also contains serious ethical implications. Primary among these are risks to data privacy, security and confidentiality. Your focus should be on making sure that while you collect, store and distribute useful data in support of the incident management process (For example, recognizing hacking attempts) people must have a way to say "Nope, this kind of data is not cool". This could be, for instance, personal information or information that reveals credentials to the outside world. Designing a system that takes these factors into account contributes not only to legal compliance internationally but also to the overall security of the organization implementing the approach [32,33].

An autonomous response system might introduce an untraceable factor in the incident management flow. Such systems may independently execute various response actions and transactions without human intervention. This adds to the incident-management toolset, and often expedites the work process getting things

done more accurately and efficiently but also raises the prospect that errors or even catastrophic decisions can happen without anyone noticing or without an audit trail. In extreme interpretations, these concerns might lead to condemning anything posed as a "response action." Clearly, response actions must be subject to proper planning, monitoring, and evaluation, determined by the organizational context, human oversight requirements, and the risks involved.

10.2. Accountability in Autonomous Systems

As autonomous response systems gain the ability to make decisions without direct user intervention, maintaining a sense of accountability can become challenging. In particular, organizations must be able to hold themselves responsible for negative consequences that might someday result from the operation of such systems. Clear documentation regarding feature design and interaction with other system components can help elucidate which stakeholder groups share responsibility for particular actions taken by an incident-management service.

Given that a chat-based experience nevertheless requires a computer agent to perform interventions on behalf of human users, service operations should ideally also support aspects of human directing, collaborative domain-expertise, full-decision authority and automation oversight. The ability of a ChatOps system to allow a user to "speak directly" to a service thereby enables incident management that is not only highly efficient in operational terms but also aligned with human requirements for responsibility, collaboration and control.

11. Technical Challenges

The previous section outlines a practical framework for integrating ChatOps with autonomous response systems to bolster the incident-handling cycle of AI-driven managed services. Although the framework is effective, integration presents several challenges, not least of which concerns technical execution. Other challenges relate to ethics, training, and user experience.

Technical considerations also include the ability to adapt the framework to various AI-driven managed services. The feasibility, stability, and flexibility of the ChatOps integration approach can be enhanced by addressing such issues.

11.1. Integration Difficulties

Integrating ChatOps with Autonomous Response Systems in AI-driven Incident Management presents a number of practical challenges. Although ChatOps is

naturally supportive of AI-assisted and autonomous incident management systems in terms of automation and communication, the use of ChatOps with autonomous response systems in such complex scenarios remains limited. This section elaborates on these integration difficulties and proposes appropriate solutions.

Artificial Intelligence (AI) techniques are often employed in incident management systems to identify and classify incidents, while automation techniques are used for their ensuing resolution. In current practice, classification and response suggestions are generated within the incident management system itself and then presented to the user through different communication means. Applying ChatOps to such incidents can help decrease the time taken to notify the correct users and obtain their actions. However, such integration entails considerable effort because the ChatOps system must be designed and implemented alongside the incident management system to ensure seamless integration. Fully autonomous response is feasible if the incident management system can automatically recommend and execute remedial actions, though such systems often face scalability limitations in development and deployment.

11.2. Scalability Concerns

The integration of ChatOps with autonomous systems in incident management enables analysts to rapidly unlock assets, remediate environments, update documentation, and collaborate throughout an incident's response—all in a single window and all within the context of the ongoing dialogue. This reduces latency, alleviates operational burdens, and greatly improves the operational and user experience. However, the move towards real-time and responsive incident handling presents scalability concerns.

Managed environments have rapidly expanded in size and diversity, seeing growth not only in digital footprint but also in business and market footprint. Available actions for analysis and mitigation have grown accordingly. Yet, the ability of halls of analysts working around the clock has not [34-36]. Given these realities, growth in the managed environment has outpaced response-proprietor availability. The latency introduced by analysis and execution now dictates the pace of remediation in reactive defensive strategies.

12. User Training and Adoption

Integrating autonomous response systems into a ChatOps environment can enhance the incident management process in many ways. It enables a part of the

response to be fully automated, removes the “black box” feeling around these decisions by providing them in a transparent way in the active communication channel, tracks the reasoning behind response decisions, creates a central place to assign and maintain follow-up tasks and canned incident response actions, and automates status communications to stakeholders directly in the communication channel. However, to fully reap the benefits, users need to be trained and encouraged to use ChatOps instead of legacy communication tools.

The transition can be challenging due to the prevalent use of communication applications in mission-critical situations such as incident management and response. Changing these tools is difficult given the reliance on instant communication, the limited amount of text input that users are willing to provide when entering or searching for incident information, and the need for fast reminder information when replying to teen-driving text messages. Tools that are familiar to end-users but can also receive input from a bot when changes happen in the incident and response lifecycle provide a hybrid approach that blends the strengths of legacy tools with the benefits of dedicated technical incident management tools.

12.1. Training Programs

Training supported the transition to ChatOps, but with special attention to transform the culture as well, because of the impact of the collaboration initiatives. This process has been rewarding and helped popularize the benefits to other teams and to the entire organization. It is worth emphasizing that it was possible to attract other teams somewhat automatically, offering a pleasant experience to users, with automations that reduce the time for them to perform their usual activities.

Four modules are defined, with the details of the content developed for the stage of ChatOps integration with autonomous response systems in AI-driven incident management. The first two modules focus on the ChatBot user experience, whereas the last two address support for operators and managers.

12.2. Cultural Change Management

Cultural transformation within an organisation cannot be subsumed under a mere training program but must be pursued as an evolutionary process. According to Lewin's phases of change, it should be gradual and uncoercive. Whenever a change is proposed, three psychological phases can be distinguished in those affected: unfreezing, change, and freezing phases. In the first phase, the reasons and necessity for the intended change are elucidated and the employees’ minds prepared for it. During the change phase, the new behaviours are learned,

practised, and anchored in the organisational routine. In the last phase, the new routine is secured and stabilised by the establishment of new convictions and attitudes.

13. Tools and Technologies

ChatOps has emerged as an approach to incident management that enables providers to take advantage of automation without sacrificing real-time collaboration and communication. The rise of autonomous response controllers—which employ recommendation engines or closed-loop automation—is changing the landscape of incident response and inevitable integration with ChatOps. A practical framework is presented for integrating ChatOps with autonomous response systems, along with an analysis of the challenges and best practices for real-world implementation.

Incident management is a growing challenge as infrastructure grows in scale and complexity. Artificial intelligence for IT operations (AIOps) systems have begun providing incident management assistance, but operators often have to turn to external tools, such as issue tracking software, to close the loop. ChatOps introduces a family of patterns that connect people, tools, and processes into a collaboration suite, enabling a "human-in-the-loop" approach, so that humans can manage work from a conversational interface. The integration of ChatOps with autonomous response systems—tools capable of real-time impact on infrastructure—aims to create a "human-on-the-loop" mode, offering a collaborative framework for human oversight in real-time closed-loop control of systems and infrastructure.

13.1. Software Solutions

Software solutions for the integration of ChatOps and Autonomous Response Systems encompass a broad spectrum of applications within the operational-mainitoring, DevOps, and collaboration-management categories, fulfilling the recommended functional capabilities [6,9]. Key performance indicators emphasize innovative incident-management features and sophisticated automation that alleviate the burden of pragmatic analysis and manual configuration tasks. Aesthetic objectives consider intuitive user interfaces and sleek design, which enhance the overall experience for both backend administrators and end customers–employees and users of the IT service. Evidently, automation flourishes most in the milieu of communication; consequently, software destined for integration with Ansible, the most popular

operation-automation tool, expands the ChatOps horizon far beyond the confines of a Linux command shell.

ChatOps builds upon the foundational capabilities of chat clients, chat rooms, chatbots, and IRC bots, introducing new software categories and classifications. Multi-platform chatbot engines and frameworks facilitate the development of conversational agents, while chatbots deployed on web services provide support for teams and individual users. Collaboration and project-management platforms not only empower users and teams to optimize their work but also serve as the operational and internal communications center for the entire organization. Command bots and operational bots control the execution of operational commands and infrastructure management, enabling streamlined and secure interactions. Furthermore, chatbots dedicated to monitoring and alerting silently collect metrics and logs, continuously scanning for anomalies, outstanding incidents, problems, and changes to systems and services. Finally, chat-specific Marketplace Applications integrate with third-party services or deliver novel services within the collaboration platform, including autonomous-response systems that actively participate in incident management processes.

13.2. Collaboration Platforms

ChatOps capitalises on the collaboration tools framework widely adopted within agile organisations, a set of modern computer-mediated communication platforms—primarily Slack, Microsoft Teams, and Discord—that facilitate instant and synchronous interaction across a distributed user base. The trend to adopt such tools creates an opportunity for increasingly efficient incident management. ChatOps commands can be shared within channels, exchange rooms, or chat groups, and form an essential element in the creation and operation of autonomous response systems. It is also important to consider how other types of incident, support, or service workflows can be initiated, progressed, and concluded.

Microsoft Teams is a communication platform service developed as part of the Microsoft 365 family of products. Teams is a proprietary software offering that combines workplace chat, video meetings, file storage, and application integration. Designed for business communication, it supports chat, videoconferencing, file storage, including collaboration on files, and workflow automation with Microsoft Power Automate. Slack is a business communication platform offering many IRC-style features, including persistent chat rooms organised by topic, private groups, and direct messaging. It also integrates with many third-party services and supports community-built integration through a well-documented application programming interface. Discord is a VoIP and

instant messaging platform, specialised in video gaming communities but with growing uptake in business settings [36]. It allows users communication by voice calls, video calls, text messaging, media, and files in private chat channels or as part of communities called "servers". Users can streamline communications within Slack, Discord, or Microsoft Teams by integrating it with other systems, services, applications, and tools.

14. Performance Metrics

Integrating ChatOps solutions with autonomous response systems can measurably improve the experience of incident management in AI-driven environments. Such improvements are therefore worthy of measuring, and various Key Performance Indicators (KPIs) can be chosen for the purpose. These KPIs can be applied to both planned experiments and to an ongoing monitoring program, with targets and threshold values determined as appropriate in each case.

Research suggests that the KPIs corresponding to a high level of Tooling and Automation represent a comprehensive set. Prolonged breaches of the empirical Golden Signals of Latency, Traffic, Errors and Saturation are commonplace. These could therefore be augmented with a suite of KPIs related to the quality of the user experience during the full sequence from detection through diagnosis to remediation. For example, recent research has demonstrated the measurements of Average Speed to Respond, Failure Rate, Average Time to Remediate, Efficiency and Quality at appropriate points along the detection–reaction–resolution pipeline.

14.1. Key Performance Indicators

Linking KPI and overall strategy is a general best practice. Resource-usability and customer-satisfaction KPIs respond to these questions: How many resources were created? How usable are the created response resources? How _usable_ means the end users' satisfaction with the response resources. Time-saving KPIs respond to this question: How much time was saved by the system? Resource-usability and time-saving KPIs are helpful for operations teams; customer-satisfaction KPIs are helpful for project leads. Autonomous response is commonly developed with multiple functions, such as self-remedy, repair recommendation, incident update and collaboration, and chatbot. From the customer's perspective, the self-remedy function can promptly handle simple incidents by therefore improving operational efficiency; the repair

recommendation function makes solving simple incidents easier for operation staff; the repair-recommendation-and-customer-update functionality provides instant incident status updates to customers; the collaboration and chatbot functions help operation staff coordinate with other involved departments regarding any operational-band incidents.

Any time-saving ability of various autonomous-response functions should be evaluated using KPIs. Different structures of these KPIs are suggested, according to the different functions. The time-saving KPI for the self-remedy function is defined as the ratio of repaired incidents to all processed incidents. The time-saving KPI of the repair recommendation function is defined as the ratio of replied incidents to all processed incidents. The time-saving KPI for the incident-update function is defined as the ratio of incident-response channels created to all processed incidents. The time-saving KPI for collaboration functionality is defined as the ratio of tickets created to handle any operation-band activities to the total number of tickets created. The satisfaction level of relevant staff is adopted as the customer-satisfaction KPI. The customer-satisfaction KPI for repair recommendation function is defined as the satisfaction level of operation staff regarding the recommended repair content; that for the incident-update function is defined as the satisfaction level of the customers who responded to preincident or postincident updates; that for the collaboration function is defined as the satisfaction level of the personnel who used the function [12-16]. A suggested implementation of the collaboration function and related time-saving and customer-satisfaction KPIs is presented in Key Challenges.

14.2. Measuring Success

Once the objectives for integrating ChatOps and autonomous response systems have been clearly defined, the project can be measured for success. For example, if the aim is to reduce resolution time, are incidents being resolved more quickly? If the goal is to boost automation, are more areas enjoying seamless reduction and resolution? List the goals, define how success can be measured for each of them, and then complete the assessment.

Establishing concrete measures of success is essential. The following steps can help focus on the practical benefits of integration: clearly specify the project intentions, identify canaries within the flock that highlight success, and highlight bigger-picture benefit areas. Answer key questions such as: What are the primary objectives? What can be actually measured? What are the known gotchas or dependencies? When is adjustment required or project stopped? And what are the signals that tell you whether the project is moving the organization in the right direction?

15. Conclusion

Takeaways on integrating ChatOps and autonomous response mechanisms It is evident that combining ChatOps with autonomous response to augment AI driven incident response is an unbeatable combination for the sake of operational effectiveness and collaboration. It's nearly impossible to overstate the value of a rapid, effective response to an incident, when coupled with the scalability and risk management considerations and fine-tuning applied there. These benefits stem from the combination of human guidance and prompt automated response, while privacy and liability are addressed.

To fully benefit from these advantages, practitioners must carefully consider resource allocation, monitoring, training, and performance assessment. This kind of careful preparation will go a long way toward confronting challenges as the more widespread dissemination of AI continues to transform incident-handling practices. Deeper insight into these underlying issues can drive more widespread adoption and allow for the transfer to other related fields.

References

[1] Karamitsos I, Albarhami S, Apostolopoulos C. Applying DevOps practices of continuous automation for machine learning. Information. 2020 Jul 13;11(7):363.

[2] Dragičević T, Wheeler P, Blaabjerg F. Artificial intelligence aided automated design for reliability of power electronic systems. IEEE Transactions on Power Electronics. 2018 Dec 20;34(8):7161-71.

[3] Moayedi H, Mosallanezhad M, Rashid AS, Jusoh WA, Muazu MA. A systematic review and meta-analysis of artificial neural network application in geotechnical engineering: theory and applications. Neural Computing and Applications. 2020 Jan;32(2):495-518.

[4] Panda SP. Securing 5G Critical Interfaces: A Zero Trust Approach for Next-Generation Network Resilience. In2025 12th International Conference on Information Technology (ICIT) 2025 May 27 (pp. 141-146). IEEE.

[5] Cheng Z, Jia X, Gao P, Wu S, Wang J. A framework for intelligent reliability centered maintenance analysis. Reliability Engineering & System Safety. 2008 Jun 1;93(6):806-14.

[6] Moreno-Guerrero AJ, López-Belmonte J, Marín-Marín JA, Soler-Costa R. Scientific development of educational artificial intelligence in Web of Science. Future Internet. 2020 Jul 24;12(8):124.

[7] Chen L, Chen P, Lin Z. Artificial intelligence in education: A review. IEEE Access. 2020 Apr 17;8:75264-78.

[8] Devedžić V. Web intelligence and artificial intelligence in education. Journal of Educational Technology & Society. 2004 Oct 1;7(4):29-39.

[9] Alam A, Alam S. Evolution of Artificial Intelligence in Revolutionising Web-Based and Online Intelligent Educational Systems. SGS-Engineering & Sciences. 2021 Sep 15;1(01).
[10] Maassen O, Fritsch S, Palm J, Deffge S, Kunze J, Marx G, Riedel M, Schuppert A, Bickenbach J. Future medical artificial intelligence application requirements and expectations of physicians in German university hospitals: web-based survey. Journal of Medical Internet Research. 2021 Mar 5;23(3):e26646.
[11] Panda SP. Augmented and Virtual Reality in Intelligent Systems. Available at SSRN. 2021 Apr 16.
[12] Calvo-Rubio LM, Ufarte-Ruiz MJ. Artificial intelligence and journalism: Systematic review of scientific production in Web of Science and Scopus (2008-2019).
[13] Gandon F. Distributed Artificial Intelligence and Knowledge Management: ontologies and multi-agent systems for a corporate semantic web (Doctoral dissertation, Université Nice Sophia Antipolis).
[14] Kietzmann J, Paschen J, Treen E. Artificial intelligence in advertising: How marketers can leverage artificial intelligence along the consumer journey. Journal of Advertising Research. 2018 Sep 1;58(3):263-7.
[15] Almeida F, Simões J, Lopes S. Exploring the benefits of combining DevOps and agile. Future Internet. 2022 Feb 19;14(2):63.
[16] Hannousse A, Yahiouche S. Towards benchmark datasets for machine learning based website phishing detection: An experimental study. Engineering Applications of Artificial Intelligence. 2021 Sep 1;104:104347.
[17] Divya S, Indumathi V, Ishwarya S, Priyasankari M, Devi SK. A self-diagnosis medical chatbot using artificial intelligence. Journal of Web Development and Web Designing. 2018 Apr 7;3(1):1-7.
[18] Bonner E, Lege R, Frazier E. Large Language Model-Based Artificial Intelligence in the Language Classroom: Practical Ideas for Teaching. Teaching English with Technology. 2023;23(1):23-41.
[19] Rao AS, Vazquez JA. Identification of COVID-19 can be quicker through artificial intelligence framework using a mobile phone–based survey when cities and towns are under quarantine. Infection Control & Hospital Epidemiology. 2020 Jul;41(7):826-30.
[20] Shivadekar S, Halem M, Yeah Y, Vibhute S. Edge AI cosmos blockchain distributed network for precise ablh detection. Multimedia Tools and Applications. 2024 Aug;83(27):69083-109.
[21] De Silva D, Alahakoon D. An artificial intelligence life cycle: From conception to production. Patterns. 2022 Jun 10;3(6).
[22] Battina DS. DevOps, a new approach to cloud development & testing. International Journal of Emerging Technologies and Innovative Research. 2020 Aug;2349-5162.
[23] Maheshwari A. Digital transformation: Building intelligent enterprises. John Wiley & Sons; 2019 Sep 11.
[24] Subramanya R, Sierla S, Vyatkin V. From DevOps to MLOps: Overview and application to electricity market forecasting. Applied Sciences. 2022 Sep 30;12(19):9851.
[25] Dang Y, Lin Q, Huang P. Aiops: real-world challenges and research innovations. In 2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings. IEEE; 2019 May 25. p. 4-5.

[26] Liu J. Web Intelligence (WI): What makes wisdom web?. In IJCAI. 2003 Aug 9;3:1596-1601.
[27] Maddox TM, Rumsfeld JS, Payne PR. Questions for artificial intelligence in health care. JAMA. 2019 Jan 1;321(1):31-2.
[28] Paschen J, Kietzmann J, Kietzmann TC. Artificial intelligence (AI) and its implications for market knowledge in B2B marketing. Journal of Business & Industrial Marketing. 2019 Oct 7;34(7):1410-9.
[29] Sterne J. Artificial intelligence for marketing: practical applications. John Wiley & Sons; 2017 Aug 14.
[30] Burley SK, Bhikadiya C, Bi C, Bittrich S, Chao H, Chen L, Craig PA, Crichlow GV, Dalenberg K, Duarte JM, Dutta S. RCSB Protein Data Bank (RCSB.org): delivery of experimentally-determined PDB structures alongside one million computed structure models of proteins from artificial intelligence/machine learning. Nucleic Acids Research. 2023 Jan 6;51(D1):D488-508.
[31] Brusilovsky P, Peylo C. Adaptive and intelligent web-based educational systems. International Journal of Artificial Intelligence in Education. 2003 May;13(2-4):159-72.
[32] Battina DS. Ai-augmented automation for devops, a model-based framework for continuous development in cyber-physical systems. International Journal of Creative Research Thoughts. 2016 Sep 3;2320-882.
[33] Enemosah A. Enhancing DevOps efficiency through AI-driven predictive models for continuous integration and deployment pipelines. International Journal of Research Publication and Reviews. 2025 Jan;6(1):871-87.
[34] Kim G, Humble J, Debois P, Willis J, Forsgren N. The DevOps handbook: How to create world-class agility, reliability, & security in technology organizations. IT Revolution; 2021 Nov 30.
[35] Myllynen T, Kamau E, Mustapha SD, Babatunde GO, Collins A. Review of advances in AI-powered monitoring and diagnostics for CI/CD pipelines. International Journal of Multidisciplinary Research and Growth Evaluation. 2024 Jan;5(1):1119-30.
[36] Etzioni O, Cafarella M, Downey D, Popescu AM, Shaked T, Soderland S, Weld DS, Yates A. Unsupervised named-entity extraction from the web: An experimental study. Artificial Intelligence. 2005 Jun 1;165(1):91-134.

Deep Science Publishing, 2025
https://doi.org/10.70593/978-93-7185-060-5

DeepScience
Open Access Books

Chapter 7: Enhancing Security and DevSecOps Through Artificial Intelligence

1. Introduction to Security and DevSecOps

Security Security can be generally defined as the protection of assets from potential damage or loss. These assets can be represented by information, services, IT systems, or the users and the events of interest are defined as natural hazards, physical damage, cyber intrusion, data loss, theft, or fraud. Protection guarantees that assets will be available, kept private, and have their integrity maintained. The philosophy, culture and practice of integrating and automating security features and tools with DevOps processes is known as DevSecOps. Security policies need to mature along with systems.

Cyber security enables security of connected infrastructures, usually by means of defensive software: Firewalls and kinetic systems. To detect weaknesses, or possible vulnerabilities inherent in the systems, whether this is a security vulnerability or a design flaw scanning tools are used. This circular reviewing and enhancing of system is known as risk management. The functional evolution of risk is influenced by phases of the system development life cycle too.

Security controls are also evaluated through behavioral anomaly detection, beside vulnerability scanning. This pertains to controlling and administering the behavior of system users. This allows organizations to make decisions about whether user actions are valid and to recognize patterns of behavior that may threaten the security of the system."

2. The Role of AI in Cybersecurity

This paper presents an overview of two AI-driven methods of vulnerability scanning and behavioral anomaly detection and evaluates their usability in the context of DevSecOps. Vulnerability scanning is an integral part of risk analysis conducted prior to deploying software to identify possible security issues and to determine their scope. AI supersedes the traditional security approaches, which are cumbersome, using supervised and unsupervised machine learning models, which have been trained with the security classified data from the existing tools [1]. Behavioral anomaly detection acts as ongoing-risk determination technique by examining behavioural character which are generated by sequence of events. It detects and makes you aware of software abnormalities when it's doing something it's not supposed to do (e.g., zero-day exploits).

With the characteristic of software development, the development and operation are combined in a more effective way. Security is fundamentally important in the consolidated environment, and the recent studies aim to further develop DevOps practices with security-based knowledge. Support systems, such as machine learning models, help engineers cope with the additional work required. Additionally, major technology companies offer AI services, making it cost-efficient and relatively straightforward to implement during software development, operations, and maintenance. Due to the complexity and maturity of AI, it cannot replace current practices but can automate simple and routine tasks. Being aware of the associated risks enables better control of AI implementations.

3. AI-Driven Vulnerability Scanning

The goal of static application security testing (aka SAST) tools, including vulnerability scanning tools, is to automatically identify vulnerabilities and security weaknesses in source code, binaries, application services, and open source packages that are used by an application. The intent is to identify vulnerabilities early during the software development lifecycle to minimize risks and allow adequate time to plan and perform remediation actions. Although static application security testing tools are widely used, organizations often report poor results because of the massive number of false positives and inhibited business workflow.

Artificial intelligence techniques, specifically applied machine learning methods, are being exploited to improve vulnerability scanning processes and provide more meaningful results. In recent years, innovative ideas and approaches have been developed and put forward to improve static application security testing tools through machine learning techniques in order to reduce false positives and better prioritize vulnerabilities [1,2]. The ever-increasing numbers of real implementation examples deployed in industry, including published pieces of empirical research, have proved the superiority of these newly proposed approaches and methods compared to traditional static application security testing tools.

3.1. Overview of Vulnerability Scanning

Vulnerability scanning is a fundamental process for securing information systems. It involves identifying known vulnerabilities, shortcomings, misconfigurations, and deviations from best practices that may weaken a system and offer an entry point to attackers. While vulnerability scanning is not the only cybersecurity control required for comprehensive protection, it plays a pivotal role in effective risk management. A wide array of scanning tools are available to perform these examinations and assessments.

Vulnerability scanning and detection have become routine, recurrent, and automated activities. Provided a scanning tool can successfully probe a system and verify the existence or absence of a defined vulnerability, the process is sufficiently straightforward to lend itself well to automation and integration into the build pipeline. This integration aligns with a DevSecOps approach by delivering relevant information related to root causes, potential exploit paths, attack payloads, and mitigation measures. Such insights help the development and security teams allocate resources more effectively when implementing fixes, thereby reducing the exposure window.

3.2. AI Techniques for Vulnerability Detection

AI techniques used in vulnerability scanning build on pre-determined patterns. Vulnerability researchers curate large vulnerability databases, which can be processed with AI, especially natural language processing (NLP), and transformed into knowledge graphs. Edge AI makes use of these knowledge graphs and pattern information for fast identification of relevant vulnerability exploitation patterns within network traffic and other data sources.

AI methods complement signature-based vulnerability detection by incorporating pattern matching, behavioral analysis, classification, clustering, and anomaly detection. These techniques are implemented in the convolutional

layers of Convolutional Neural Networks for hierarchical feature learning. Practical applications demonstrate AI-powered continuations and extensions of traditional scanning approaches. Challenges associated with these applications, including data quality, model selection, privacy concerns, and integration with existing security frameworks, are also discussed.

3.3. Case Studies of AI-Driven Solutions

The following case studies illustrate the practical applicability of the aforementioned AI techniques. They demonstrate that the discussed risks represent significant areas warranting attention.

In a 2021 study by Deep Instinct exploiting AI in offensive attacks, Wasserblum et al. implemented an advance of persistent attack—Advanced Persistent Threat (APT)—using the Red-Black Lists technique. These lists act as solutions or signatures for both adversaries and security teams, respectively, embodied in a malicious APT program. This APT learns from an original dataset of victims—the Red List—and subsequently trains its AI module, enabling it to reach a set of victims from the Red List and avoid attacking them. The process of learning and avoiding these past victims diminishes the successful or compromised victim rate of the attack by 20%, clarifying the advantage to the adversaries.

A recent investigation by IIBH-Infosec into AI technology for SaaS products applied AI to reduce false positives around automated Quarantine non-intelligence (AINI). This process marks an email as safe in the quarantine application and teaches AI based on an array of confidence levels. The subcategory of Sensitive Phishing is the most prevalent phishing-related threat in the European Financial Services sector. To address this, a new AI Subcategory was developed to combat the negative impact of these emails, providing the ability to reduce the stress of false positives and implementing the learning aspect of AI. This reduction in false positives allows for the opening of additional features in Secure Email Gateways (SEGs).

3.4. Challenges in AI Vulnerability Scanning

AI-driven vulnerability scanning approaches introduce a number of risks, limitations, and open challenges. A primary difficulty lies in the lack of explainability of many AI algorithms, a problem commonly referred to as the black box problem. Explainability, or lack thereof, can substantially affect the adoption of new techniques both in industry and for research purposes. In the development of continuous-model–updating approaches, a reliable and automated feedback mechanism is required to assess the quality of predictions and stabilize the learning process. Such feedback can be applied not only to a

neural network model but also to reinforcement learning through appropriate design of rewards and penalties. In a real security environment, where actual attacks may be mixed with various network noises, the introduction of pseudo labels and the accuracy of pseudo labeling require special attention to prevent error propagation during training and testing [3-5].

Adopting an AI-driven approach in critical environments requires special care and well-defined policies for risk management to maximize the value of AI while minimizing the potential consequences associated with AI. Using the matrix model based on CRA, which is essential to prioritize, identify and treat potential security threats and vulnerabilities to adopt reasonable rules for constructing and applying the model to become compliant with existing guidelines.

4. Behavioral Anomaly Detection

Behavioral Anomaly Detection Behavioral anomaly detection techniques are a category of methods that detect deviations from normal behavior by, in example, monitoring unexpected activity or misuse. Unlike vulnerability scanning that scans for known flaws, intrusion detection concentrate on the "abnormalities" cause by exploiting weaknesses--and therefore an alert of abnormal activity might imply weakness. Supervised classification algorithms like Decision Trees (DT), Random Forests (RF), Support Vector machines (SVM), K-Nearest Neighbor (KNN) are the common base algorithms for anomaly detection. But data sets in such tasks are usually limited and may not be enough for overall training and testing. To overcome this problem, several approaches based on supervised, unsupervised or reinforcement learning techniques have been designed.

The common theme of using artificial intelligence (AI) techniques is automation of security tasks, decrease of the false-positive ratio and the detection of such attacks again by profiling of early risks it supports. This approach aligns with the goals outlined by "State of DevSecOps Report 2023: Securing Software Development in an AI-First World," which describes AI as a pivotal element in advancing security and DevSecOps.

4.1. Understanding Behavioral Anomalies

Security and DevSecOps with AI Behavioural anomaly An anomalous behaviour in software development, operations, or business processes is any observable behaviour that doesn't match the established normal behaviour [6,7]. These types of anomalous behaviour are often seen in such phenomena such as bursts of network traffic from a single IP or rapid acceleration of certain error classes

produced by an application. Unusual behaviours could be intentional weaponization of software services or software vulnerabilities. Not all anomalies are related to cyber- security vulnerabilities, but a large share of cybersecurity incidents feature anomalies, justifying the use of behavioural anomaly detection in the area.

Behavioural anomalies can be naturally thought of as pattern recognition and discovery of insights in large and very diverse data sets. Consequently, a large number of organizations employ artificial intelligence, which can learn certain patterns and correlations in datasets, for identifying anomalies in software systems. AI-supported behavioral anomaly detection diminishes the reliance on human experts, who often face enormous volumes of operational and application data, enabling faster responses to detected anomalies and threats. The initial focus rests on vulnerabilities—weaknesses in applications or infrastructure susceptible to exploitation—but the broader approach also encompasses misconfigurations that can be misused with insufficient controls.

4.2. AI Approaches to Anomaly Detection

This section discusses behavioral anomaly detection using AI. It begins with behavioral anomalies, followed by AI approaches for detection, implementation, and use cases.

System behavior originates in particular states and processes responding to changing environmental conditions [2,8-10]. In closed systems, cause–effect relationships govern operations in a linear manner, but in open systems behavior can be significantly non-linear. Most systems are inherently open and therefore non-linear. Environmental conditions can include factors indicating the state of other systems, for example, weather reports, road traffic information, and financial market indices. Consequently, when exception conditions occur, operation follows different rules and can be indicated by different behavior of the system. Anomalies appear in system behavior whenever inputs or the combination of inputs and context conditions reach abnormal values and when cause–effect relationships adapt according to exceptional circumstances. A relationship therefore exists between behavioral anomaly and risk, as when an anomaly is encountered also the risk distribution of the system at that moment is changed from the risk distribution encountered when operation is normal [1,11-12]. Many real-world applications in the security field can gain competitive advantages by implementing Detection Methodologies able to spot abnormal behavior of the system. Examples include credit card fraud detection, denial of service detection, and insider threat detection. Recent developments in Artificial Intelligence have allowed several approaches that can improve the performance

of behavioural anomaly detection to come to fruition. These include AO-DNN, a novel Anomaly-Detection framework for dynamic networks which transforms the dynamic network into a sequence of static snapshot graphs and directly tackles the node-level anomaly-detection task on dynamic networks using a combination of Graph Convolution Network and Long Short-Term Memory model; an on-line credit card fraud detection approach that, through a Recurrent Neural Network, is able to highlight which transactions are unusual and could be systematically checked by human analysts; a hybrid method made up of an LSTM model, able to learn normal behaviour of users from sequences of events, and one-class SVMs libraries, executed according to groups of similar features and capable of detecting anomalous behaviors; an insider threat detection framework whose effectiveness is implemented using RNN to learn data in time sequence, designed to dig deep into changing employee behavior in order to detect potential insider threats; and a framework that inoculates an Artificial Immune System with External Self Antigens to build up its tolerance, in order to reduce the false positive rate generated by behavioural anomaly detection systems based on the negative selection algorithm.

4.3. Implementation Strategies

The aforementioned AI techniques require proper superstructures for practical use. Automation is a natural consideration, as scanning is a tedious task for humans, and the speed of AI, which can scan multiple projects in a short amount of time, allows highly frequent scans.

In the context of DevSecOps, automation must be carefully approached, with the ultimate goal being the integration of scanning into projects that utilize continuous integration (CI). AI-based testing can play a major role in this space by conducting large-scale testing that is difficult, or even impossible, for humans. Nevertheless, the risks connected with AI testing must be managed, and experienced humans are still needed. One approach is enforced voluntary automation: when an AI model achieves a known accuracy level, it is granted access to the task it was built for [13-15]. This method restrains the system from uncontrolled testing and from performing actions that may be harmful to the project; instead, the AI model can operate only within the confines of the knowledge gained during training. Furthermore, the model has to be retrained at regular checkpoints, particularly when its task implies interacting with other elements that may evolve over time.

4.4. Real-World Applications

Artificial Intelligence driven Behavioral Anomaly Detection in personalized learning Artificial Intelligence based Behavioral Anomaly Detection in

personalized learning is a step forward towards educating in personal life. CJPR/ECOAB with the ChatGPT added management system processes the student's questions, recognizes feature vectors and part of speech tags to reply with related information serving as the virtual instructor. CLAGMetaEDU utilizes an AI chatbot with a knowledge graph to detect the abnormalities of students' behavior in LHP. Universities utilize Jovian, an intelligent medium driven by ChatGPT and context vectors, to take learners on a journey through convoluted concepts

Applied to e-commerce, AI-enabled behavioral anomaly detection can also streamline the shopping experience. Profiles for online trade are created and analyzed by Graph Above Ground in T-commerce, capturing trading preferences and behavior while identifying anomalies during e-shopping. SEAO effortlessly fetches suitable online e-commerce products using NLP and word embeddings. UBLAP's advanced user behavior-aware list-wise prediction algorithm anticipates preference transitions during e-shopping [16].

5. Integrating AI into DevSecOps Practices

DevSecOps represents the cultural change that places security at the center of both development and operations. Therefore, it requires the implementation of appropriate testing activities as part of a continuous integration scheme and the constant supervision of the software as it is deployed so that it is possible to respond promptly to any malfunction or breach. AI can provide DevSecOps practitioners with various valuable contributions fulfilling these needs.

Thanks to the continuous integration that allows the software to be always up to date, regular scanning for vulnerabilities can be easily automated and promptly run as soon as a new version is released [16,17]. AI can be included in the toolchain in order to perform behavioral anomaly detection, which makes trustworthiness assessment and advanced evaluation of the severity of vulnerabilities possible. Deployment processes might benefit from such an anomaly detection system, which can continuously monitor the running software and promptly react to dangerous on-the-fly behavioral changes. Moreover, AI can be employed not only for the development and operation stages but also in the process of identifying the vulnerabilities to be used for the automatic scanning itself.

5.1. DevSecOps Framework Overview

Security is essential to every aspect of everyday life. No longer do we live in a world where we can recognize a new email message from an unknown sender as something that almost certainly contains a virus. The Internet of Things (IoT), the Cloud, Social Networking sites, and the way that every aspect of our everyday lives is linked to the Internet are examples of how everything and everybody are connected and potentially vulnerable. The Cybersecurity and Infrastructure Security Agency (CISA) was created in 2018 to (i) identify and reduce the risks of physical and cyber incidents to the nation's critical infrastructure and (ii) provide timely, reliable, and actionable information to the President, the Secretary of Homeland Security, and the private sector, so that actions reducing risk and enhancing shared security and resiliency are well-informed and well-coordinated.

Artificial Intelligence (AI) is becoming a vital technique used to strengthen the protection of critical infrastructure. AI techniques are used to perform! vulnerability scanning and behavioral anomaly detection on links within an information system.

5.2. AI Tools for Continuous Integration

One powerful use case for integrating AI capabilities in DevSecOps is the automation of continuous integration processes. +Then comes a phase where developers submit their code to a central repository where AI-driven scanning tools examine the contributions to look for mistakes, vulnerabilities or other signs of trouble. These are tools that deliver the near-immediate feedback required to integrate the new code into the project, but they also test for somewhat idea-like entities, checking not just for bugs, but also where the new code falls regarding project's mission and its best practices. This AI-supported assessment allows quicker response times and lowers the potential of mistakes running forward in the innovation process. In signaling problems up front, AI tools help prevent the acceptance of suboptimal code before becoming more deeply embedded.

AI can also help with post-merger code scanning. While branching helps to minimize the effect of a single misbehaved line, the inclusion of bad code into the main branch of the project is still possible [12,18-20]. AI methods, such as machine learning (ML), can automate continuous integration scanning, delivering timely and actionable notifications. Additional testing backstops the process, but providing immediate feedback when mistakes are made offers invaluable support to developers. AI never tires, has access to vast knowledge bases, and harmonizes team understanding of project standards, facilitating a level of quality assurance not achievable alone. With adequate data, AI can

anticipate the types of problems likely to arise in a project, allowing for preemptive action.

5.3. Automating Security with AI

Security analysts use AI to automate parts of a DevSecOps pipeline. Many cloud vendors provide tools that help developers write secure code. Amazon CodeGuru Reviewer, for example, uses machine learning to detect and provide recommendations to address critical issues, security vulnerabilities, and hard-to-find bugs during application development and before code is deployed in production. CodeGuru Reviewer uses Amazon's internal Static Application Security Testing (SAST) tools and some models from Amazon CodeGuru Reviewer Research and creates a Unified Model trained on thousands of code reviews, bug detectors, and security analyzers.

Policy compliance checks are integral to software development. Automated Lighthouse and CloudFormation compliance checks identify various levels of risk in projects, managing budget, assets, and data. A vulnerability scanning policy triggers Snyk through Evident.io to detect, monitor, and notify vulnerabilities in Amazon EC2 instances on AWS. Snyk evaluates potential risks in code, dependencies, containers, and infrastructure as code. HackerOne creates vulnerabilities from GitHub and DevOps tools and dispatches tasks based on organizational policies, providing a seamless feedback loop to development and operational teams.

6. Risk Management in AI-Enhanced Security

The application of artificial intelligence (AI) techniques in cybersecurity inevitably introduces new risks that organizations need to manage. Some of these risks are common across various AI domains: adversarial examples that can fool AI-powered vulnerability scanners into missing certain vulnerabilities, known previously as evasion attacks; poisoned models trained on modified data intended to produce inaccurate outputs, also referred to as data poisoning attacks; exploitation of AI-specific biases; large resource consumption; and insufficient model training or validation, which consequently results in suboptimal performance [21-23]. Others are specific to the application context of AI-driven vulnerability scanning and behavioural anomaly detection. For example, in behavioural anomaly detection, a high rate of false positives—i.e., incorrectly flagged incidents—can lead to operational inefficiencies or oversight of real security breaches.

To mitigate these risks, organizations should establish effective risk management frameworks for the operation of AI-supported vulnerability scanning and behavioural anomaly detection. One potential mitigation strategy consists of internal and external red teaming to uncover weaknesses proactively. Additionally, mature process and technology risk management, coupled with regular validation of implemented models, helps safeguard operational stability. Finally, organizations must be prepared for regulatory developments that may hold them accountable for negligent or malicious use of AI-based tools in their cybersecurity operations.

6.1. Identifying Risks

Even with all their advantages, AI-driven cybersecurity tools also present certain risks that should be considered, understood, and mitigated. For example, in the case of vulnerability-scanning tools like the one described by de Lima et al. (2021), security teams need to understand the models behind the tool to perceive its weaknesses and limitations. Proper risk management also enables organizations to meet regulatory requirements. For instance, the European Union's AI Act includes provisions to ensure that AI applications do not endanger people.

Other examples of AI risks emerge in the context of behavioral anomaly detection, also known as behavior-based anomaly detection, which is the practice of identifying suspicious behavior displayed by a system or user. AI-empowered security systems learn, analyze, and predict behaviors associated with user accounts, network communications, enabled applications, executed commands, and other factors. As the capabilities of such systems grow, enterprises need to bear in mind the risks associated with their training, including incorporated biases. Figure 6.1 highlights some of the key risks.

6.2. Mitigation Strategies

Mitigation strategies should be implemented on the client side, cooperation side, platform side, and cooperation platform side according to both the corresponding mitigation strategy of the detected risks within the security operation framework and the related policies, laws, and regulations. At the most basic level within the DevSecOps infrastructure, establishing authentication and authorization for security scanning products and services is highly recommended [24,25]. Additionally, configuring an RBAC (role-based access control) model and assigning different roles and authority levels to user groups is advised. When integrating capabilities from commercial security scanning platforms or products, except for fully open-source products, adequate risk prevention and

control measures should be taken based on the vulnerability and risk characteristics of third-party products or APIs.

Vulnerability scanning can identify security loopholes in complex entities, network structures, and online components. A scanning plan and policy can be established based on the scanning scope and business needs to determine whether subsequent security testing is needed. Security personnel can publicly disclose the necessary security testing locations of online services for internal or external security testing, effectively controlling risks. Prevention and control still require corresponding test and emergency plans and disaster recovery capabilities. Integrating the scanning products and results with the enterprise management platform assists in completing the overall security management of the enterprises.

6.3. Compliance and Regulatory Considerations

Proper use of AI-driven vulnerability scanning and behavioral anomaly detection techniques helps organizations secure their services and products and adequately respond to incidents. Because those techniques rely heavily on the SGML process, risk management approaches may contribute to their maturity, i.e., the degree of certainty of their adoption. In addition, those approaches may be mandated by regulatory bodies that impose the use of cybersecurity assessment and risk management frameworks—particularly in regulated sectors.

Results achieved by applying AI-driven vulnerability scanning and behavioral anomaly detection should be documented for compliance purposes. As noted in Section 6.2, security requirements traceability is another compliance aspect further supported by AI.

7. Future Trends in AI and Security

Cybersecurity is reaching a new level through various techniques within the field of artificial intelligence (AI). Researchers constantly improve these techniques and discover new ones to enhance the capabilities of AI in enterprise security operations. Two aspects of AI application are AI-driven vulnerability scanning and behavioral anomaly detection. AI-driven vulnerability scanning involves training the AI on large datasets of attack patterns and targeting software. Implemented in DevSecOps solutions, it provides business managers, software development teams, and security teams with advisory content based on the detected vulnerabilities and recommends measures to avoid potential vulnerabilities. Behavioral anomaly detection also employs AI techniques applied to large-scale, complex ecosystems, innate patterns of business, and

typical activities of employees and customers. AI learns and measures such behavior and alerts unusual and suspicious activities, enabling security operations teams to take proper measures promptly.

Future AI-driven security measures will extend protection, detection, remediation, and reaction capabilities for organizations. An integrated DevSecOps platform, with AI and role-based dashboards that segregate duties and responsibilities between business managers, software development teams, and security operations teams, can transform security operation centers with real-time protection, detection, and progress. Furthermore, companies can utilize the AI-developed dashboards and techniques to alert anomalies in employees' behavior, locate where specific areas require further attention, capture real-time suspicious action by threat actors, and recommend immediate actions, considering aspects such as seriousness, risk, and sector.

7.1. Emerging Technologies

Cybercriminals are increasingly deploying artificial intelligence (AI) to infiltrate corporate networks and systems, exploiting vulnerabilities to compromise proprietary data and steal trade secrets. These AI-powered attack tools automate intrusions, deploy malware, bypass security controls, and exfiltrate data, diminishing the effectiveness of traditional, preventative security measures. Consequently, cybersecurity practitioners are responding in kind by developing advanced AI techniques to identify, defend against, and counteract these new forms of adversarial behavior.

Emerging AI-driven defensive measures incorporate vulnerability scanning systems capable of rapidly detecting deviations, anomalies, and novel attacks within digital infrastructures [26-28]. Advances in behavioral anomaly detection complement these systems by analyzing data patterns awaiting attackers' active exploits. For instance, examining the surge in behavioral deviations that preceded the recent surge in UK hospital system ransomware attacks could have informed early warning risk assessments. Proactive detection of attacker activity within enterprise environments effectively disrupts the malicious operation of AI attack tools, mitigating potential damages and enabling robust intelligence for law enforcement investigation.

7.2. Predictions for AI in Cybersecurity

In the final subsection of major predictions for AI in cybersecurity, recent developments and expert opinions forecast AI's future role in devsecops and cyberdefense. The discussion begins with automation and then advances to web vulnerabilities and machine learning. Automation will be central for businesses

once firewall AI is operational [29-30]. Its capability to scan the web for vulnerabilities and bugs, reminiscent of a cautious human tester navigating through suspicious code, will be invaluable.

As reported in a Forbes article, the deployment of AI in cybersecurity has surged. Research from Juniper Network estimates that automation powered by AI can cut cyberattack losses by approximately 30 percent. AI algorithms detect transaction patterns and alert institutions to any unusual client activity in real time, enhancing the rapid identification and management of suspicious or fraudulent behavior. Furthermore, Juniper predicts that by 2022, 50 percent of security budgets will be directed toward developing capabilities based on AI and machine learning that automate manual security processes.

8. Conclusion

The fields of security and DevSecOps are undergoing a transformational change due to the introduction of artificial intelligence techniques. Two of the most prominent of these are vulnerability scanning and behavioral anomaly detection. In the area of vulnerability scanning, some of the underlying principles and key concepts have been explored along with the role of AI in identifying and addressing potential vulnerabilities in complex systems and applications. Different scanning methods based on AI have been analysed and demonstrated with respect to practical implementations. Opportunities and risks of AI in this area were also investigated. From the viewpoint of behavioural anomaly detection, the principles and importance of this method for the detection of an arbitrary or suspicious behaviour that may portend from security sabotage and break were investigated. The role of AI in augmenting behavioural anomaly detection was then discussed. A number of techniques for detecting and classifying behavioural anomalies using AI were introduced, along with recommendations on how to apply and deploy them. Finally, real-world use cases are presented to demonstrate the Pros and Cons of AI for anomaly detection in real-life.

As AI continues to evolve at an unprecedented rate, what impact does AI have on DevSecOps? The analysis concluded that AI can be useful to also support some of the activities related to DevSecOps such as security risk management, automation of security tasks, security in CI/CD,. Collectively, these AI-driven approaches lead to successful automation of application development, security, and operational activities as part of DevSecOps.

References

[1] Buch VH, Ahmed I, Maruthappu M. Artificial intelligence in medicine: current trends and future possibilities. British Journal of General Practice. 2018 Mar;68(668):143.

[2] Ghosh A, Chakraborty D, Law A. Artificial intelligence in Internet of things. CAAI Transactions on Intelligence Technology. 2018 Dec;3(4):208-18.

[3] Raschka S, Mirjalili V. Python machine learning: Machine learning and deep learning with Python, scikit-learn, and TensorFlow 2. Packt Publishing; 2019 Dec 12.

[4] Panda SP. The Evolution and Defense Against Social Engineering and Phishing Attacks. International Journal of Science and Research (IJSR). 2025 Jan 1.

[5] Lo D. Trustworthy and synergistic artificial intelligence for software engineering: Vision and roadmaps. In 2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE). IEEE; 2023 May 14. p. 69-85.

[6] Celestin M, Vanitha N. AI vs Accountants: Will Artificial Intelligence Replace Human Number Crunchers. In Indo American Multidisciplinary Web Conference on Arts, Science, Engineering and Technology (IAMWCASET-2020). 2020. p. 117-124.

[7] Iliashenko O, Bikkulova Z, Dubgorn A. Opportunities and challenges of artificial intelligence in healthcare. In E3S Web of Conferences. Vol. 110. EDP Sciences; 2019. p. 02028.

[8] Lutz C. Digital inequalities in the age of artificial intelligence and big data. Human Behavior and Emerging Technologies. 2019 Apr;1(2):141-8. Panda SP. Augmented and Virtual Reality in Intelligent Systems. Available at SSRN. 2021 Apr 16.

[9] Karamitsos I, Albarhami S, Apostolopoulos C. Applying DevOps practices of continuous automation for machine learning. Information. 2020 Jul 13;11(7):363.

[10] Dragičević T, Wheeler P, Blaabjerg F. Artificial intelligence aided automated design for reliability of power electronic systems. IEEE Transactions on Power Electronics. 2018 Dec 20;34(8):7161-71.

[11] Panda SP. Securing 5G Critical Interfaces: A Zero Trust Approach for Next-Generation Network Resilience. In2025 12th International Conference on Information Technology (ICIT) 2025 May 27 (pp. 141-146). IEEE.

[12] Bello O, Holzmann J, Yaqoob T, Teodoriu C. Application of artificial intelligence methods in drilling system design and operations: a review of the state of the art. Journal of Artificial Intelligence and Soft Computing Research. 2015;5(2):121-39.

[13] Cheng Z, Jia X, Gao P, Wu S, Wang J. A framework for intelligent reliability centered maintenance analysis. Reliability Engineering & System Safety. 2008 Jun 1;93(6):806-14.

[14] Moreno-Guerrero AJ, López-Belmonte J, Marín-Marín JA, Soler-Costa R. Scientific development of educational artificial intelligence in Web of Science. Future Internet. 2020 Jul 24;12(8):124.

[15] Chen L, Chen P, Lin Z. Artificial intelligence in education: A review. IEEE Access. 2020 Apr 17;8:75264-78.

[16] Devedžić V. Web intelligence and artificial intelligence in education. Journal of Educational Technology & Society. 2004 Oct 1;7(4):29-39.

[17] Alam A, Alam S. Evolution of Artificial Intelligence in Revolutionising Web-Based and Online Intelligent Educational Systems. SGS-Engineering & Sciences. 2021 Sep 15;1(01).

[18] Maassen O, Fritsch S, Palm J, Deffge S, Kunze J, Marx G, Riedel M, Schuppert A, Bickenbach J. Future medical artificial intelligence application requirements and expectations of physicians in German university hospitals: web-based survey. Journal of Medical Internet Research. 2021 Mar 5;23(3):e26646.
[19] Swain P. The Artificial Intelligence and Machine Learning Blueprint: Foundations, Frameworks, and Real-World Applications. Deep Science Publishing; 2025 Aug 6.
[20] Gandon F. Distributed Artificial Intelligence and Knowledge Management: ontologies and multi-agent systems for a corporate semantic web (Doctoral dissertation, Université Nice Sophia Antipolis).
[21] Kietzmann J, Paschen J, Treen E. Artificial intelligence in advertising: How marketers can leverage artificial intelligence along the consumer journey. Journal of Advertising Research. 2018 Sep 1;58(3):263-7.
[22] Almeida F, Simões J, Lopes S. Exploring the benefits of combining DevOps and agile. Future Internet. 2022 Feb 19;14(2):63.
[23] Hannousse A, Yahiouche S. Towards benchmark datasets for machine learning based website phishing detection: An experimental study. Engineering Applications of Artificial Intelligence. 2021 Sep 1;104:104347.
[24] Divya S, Indumathi V, Ishwarya S, Priyasankari M, Devi SK. A self-diagnosis medical chatbot using artificial intelligence. Journal of Web Development and Web Designing. 2018 Apr 7;3(1):1-7.
[25] Sterne J. Artificial intelligence for marketing: practical applications. John Wiley & Sons; 2017 Aug 14.
[26] Burley SK, Bhikadiya C, Bi C, Bittrich S, Chao H, Chen L, Craig PA, Crichlow GV, Dalenberg K, Duarte JM, Dutta S. RCSB Protein Data Bank (RCSB.org): delivery of experimentally-determined PDB structures alongside one million computed structure models of proteins from artificial intelligence/machine learning. Nucleic Acids Research. 2023 Jan 6;51(D1):D488-508.
[27] Brusilovsky P, Peylo C. Adaptive and intelligent web-based educational systems. International Journal of Artificial Intelligence in Education. 2003 May;13(2-4):159-72.
[28] Battina DS. Ai-augmented automation for devops, a model-based framework for continuous development in cyber-physical systems. International Journal of Creative Research Thoughts. 2016 Sep 3;2320-882.
[29] Enemosah A. Enhancing DevOps efficiency through AI-driven predictive models for continuous integration and deployment pipelines. International Journal of Research Publication and Reviews. 2025 Jan;6(1):871-87.
[30] Kim G, Humble J, Debois P, Willis J, Forsgren N. The DevOps handbook: How to create world-class agility, reliability, & security in technology organizations. IT Revolution; 2021 Nov 30.

Deep Science Publishing, 2025
https://doi.org/10.70593/978-93-7185-060-5

DeepScience
Open Access Books

Chapter 8: Optimizing Software and ML Lifecycles Through MLOps–DevOps Convergence

1. Introduction to MLOps and DevOps

MLOps is one such CI/CD-style automated pipeline for Continuous Model Development Life Cycle (MDLC). MLOps is an abbreviation of the terms Machine Learning and Operations. MLOps can part of a closer movement between DEV and OPS.MLOps and ModelOps refers to the collaboration and communication between data scientists and operations to help manage the production machine learning lifecycle. Maintaining the health of a machine learning model in production involves considerable operational overhead of ensuring the Model Development Life Cycle (MDLC) of the model is efficient. It consists of various stages like data collection, model development, automated model verification, deployment, performance monitoring and retraining, and iterates perpetually in a cyclical manner.

DevOps is a combination of practices complete_bible that automates the processes between software development and IT complications. It seeks to reduce the system development life cycle and provide continuous delivery with high software quality. DevOps is about bringing developer and operations workflows together, with the main practices being continuous integration, continuous deployment, infrastructure as code, and monitoring/logging. It allows products and services to be iteratively improved via the use of automation and by having developers work in tandem with operations. Both MLOps and DevOps have similar objectives, but they target different requirements and challenges in bringing them together.

2. Understanding MLOps

MLOps (Machine Learning Operations) is a strategy, which leverages software engineering principles and practices, aiming at improving the quality and reducing the time and effort required to deliver, deploy and productionize machine learning models [1-3]. It achieves this by abstracting the workflow for serving machine learning models in production, such as model versioning and testing. Together, these quality and velocity enhancements make it faster to experiment with predictive models — diminishing the overall time taken to validate new research hypotheses that support the product direction of the business.

The MLOps lifecycle represents the ongoing process of machine learning models: data preparation, development, training, deployment and monitoring. Together with the real-time operation, continuous monitoring in GPU-EF allows for constant updates of the model supporting dynamic environments. This serves to cycle reload from the start by getting fresh features and labels which make the model remain discriminative and useful, making the model more robust to possible changes.

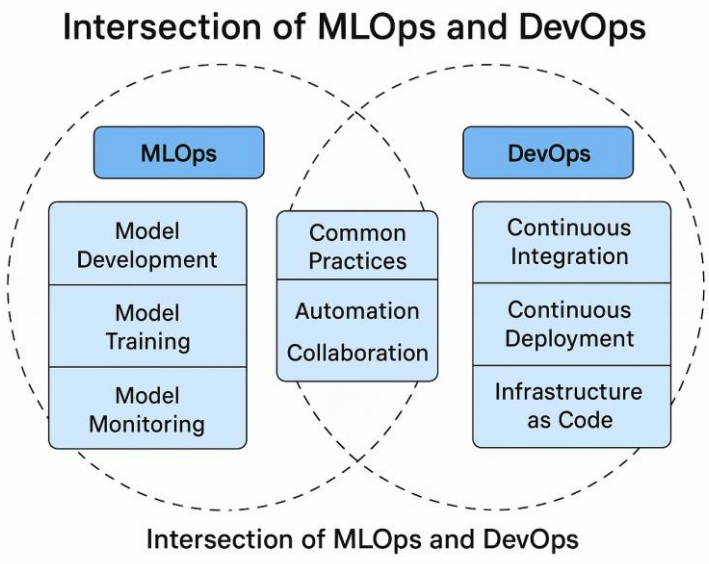

Fig 1.MLOps lifecycle depicts

2.1. Definition and Importance

MLOps, a portmanteau of machine learning and operations, encompasses a set of practices that allow for reliable and scalable machine learning model

deployment, monitoring, and management. It's helping breaking down the wall of confusion between machine learning and operations with a simplified, holistic view of the entire lifecycle of your ML models—everything from search, access, and experimentation to auditing and compliance. The process includes iterative steps like model validation and testing, release orchestration, and infrastructure monitoring. In this way, it enables the constant testing, upgrading, and retraining of ML models.

The point of MLOps is to bring the dev and operations parts of machine learning into closer cooperation but it covers a lot of the same ground as we're used to seeing in DevOps. One of the most difficult stages of the MLOps lifecycle is getting models into production. Tactics like model versioning, automated model testing, and continuous integration work together to speed up velocity and deployment frequency along with decreased risks of model failure.

Building CI/CD (Continuous Integration/Continuous Delivery) pipeline is one of the fundamental aspects of DevOps. We refer to their use in the context of data science. Integration of MLOps with the already established CI/CD pipelines, by extending with CI/CD for machine learning, encourage scrutinizing of machine learning components to continuously maintain consistency with the DevOps principles/PoCE on workflow.

2.2. Key Components of MLOps

The MLOps lifecycle consists of steps such as Automation, Change Management, Compliance and Audit, Versioning, Restoration, Model Monitoring, Continuous Integration, and Continuous Delivery. Automation aims to reduce manual effort and operational burden. Change Management helps identify and understand changes to the model or data [2].

The MLOps lifecycle includes: Automation, Change Management, Compliance and Audit, Versioning, Restoration, Model Monitoring, Continuous Integration, and Continuous Delivery. The purpose of automation is to minimize manual effort and decrease the operational overhead. Change Management assists in detecting and understanding model or data changes. Compliance and Audit capabilities provide traceability and transparency. Versioning allows for the storage and retrieval of all model versions. Restoration supports rollback of production models to earlier deployments. Model Monitoring keeps track of model performance and data drift. Continuous Integration emphasizes code quality and testing, triggering necessary downstream processes. Continuous

Delivery focuses on building, testing, and releasing every change automatically to production or staging.

2.3. MLOps Lifecycle

Key focus areas of the MLOps lifecycle are those that make ML production-ready. More specifically, activities include: (i) model training, (ii) model evaluation, (iii) model deployment, (iv) model monitoring in production, (v) model retraining, and (vi) model versioning.

Model training begins with an activity known as feature engineering, which consists of manipulating and transforming raw data into a set of features suitable for model training.In its essence, model evaluation asks a fundamental question: “Is the model good enough?”. Model deployment directs attention to the challenges presented by the model deployment mechanism, including questions related to continuous delivery/integration of models. Once the model is operationalised and making predictions in production, model inference results need to be monitored during production to detect undesirable scenarios, such as the data distribution shift. When such an event is identified, model retraining pipelines should be triggered to update the model, preventing performance degradation. Lastly, the MLOps lifecycle should also guarantee that every component created or modified is tracked and versioned as part of the model versioning.

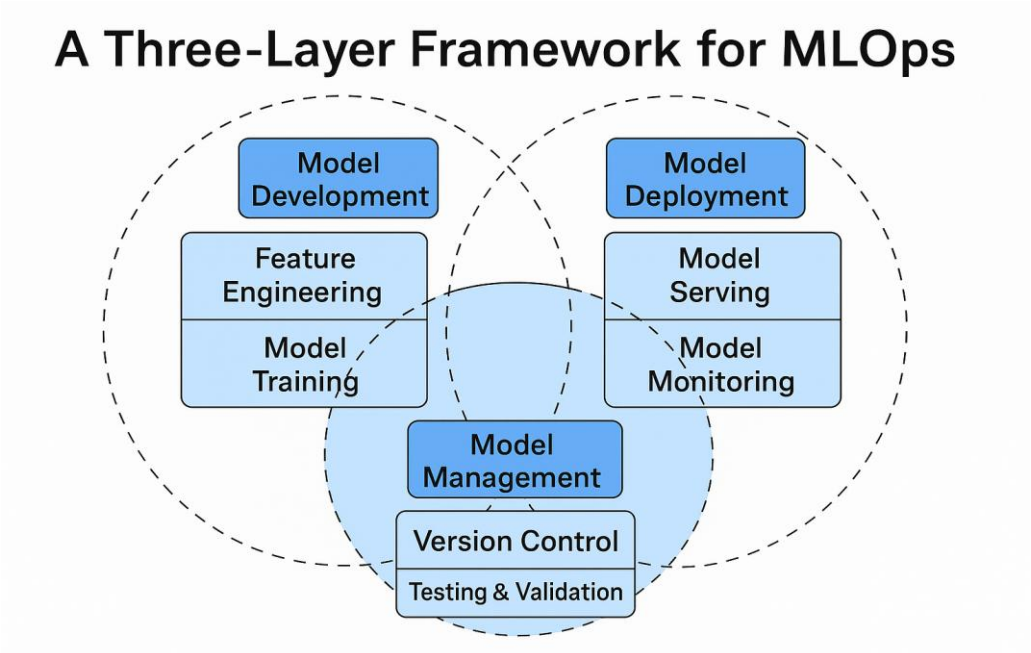

Fig 2. MLOps lifecycle

3. Understanding DevOps

DevOps evolved as a Software Development and Delivery approach with the primary objective of shortening the Development Life Cycle. Furthermore, adopting DevOps practices aims at delivering high-quality Software Continuously by constantly monitoring the Business, Quality, and Security Indicators. DevOps infuses Agility and Resiliency in Software Development, thereby significantly reducing delivery time and resources. It also helps identify and correct software defects as soon as they occur.

The focus of DevOps is on Automation, Continuous Monitoring, and Continuous Delivery of software components. The critical components in DevOps include Agile Development, Continuous Integration, Continuous Delivery, Infrastructure as Code, Release Orchestration, and Continuous Monitoring. Frequently used tools in a DevOps context comprise Jenkins, CircleCi, Bamboo, SonarQube, Selenium, Kubernetes, Prometheus, and PagerDuty.

3.1. Definition and Importance

DevOps is a software engineering methodology aimed at unifying software development (Dev) and software operations (Ops). The principal goal is to shorten development lifecycle and offer continuous delivery with high software quality.

DevOps enables development and operations teams to build better products and support their customers more effectively. It helps the development and operations teams to collaborate and work together throughout the entire software operation lifecycle [2,4]. DevOps combines people, process, and technology to deliver continuous value to end users.

DevOps consists of numerous essential components and features such as Continuous Integration and Continuous Delivery (CI/CD): an important software engineering method for development teams to integrate and deliver their product re-sources constantly—Application/Infrastructure Security, ChatOps, Microservices, Automation, Configuration Management, Continuous Monitoring, and Testing. Common industry tools that are used are Jenkins, Docker, Kubernetes, Puppet, Chef, Nagios, Selenium, Ansible, etc.

3.2. Key Components of DevOps

DevOps seeks to enable faster software releases and a quick turnaround on valuable feature requests received from the users. It is a practice that combines software development and software operations so that organizations can deliver

scalable software continuously and efficiently. The DevOps practice focuses on ensuring the software is built with a consistent and repeatable process, verified thoroughly, and is ready for deployment at any point in time. Further, DevOps encourages continuous feedback from all the stakeholders and users so that product experiences can be improved all the time.

DevOps is actually fairly clearly defined in the airgap [\url] operations in development and development in operations. The latter refers to baking of software operations tasks such as environment creation and provisioning into development. The former describes the organization or sum of all the guiding practices within the software development project (i.e., software development guidelines assemblage), and the latter teaches how to incorporate develop practices (for instance, configuration management and version control) into software operations within the company. These are collaboration initiatives aimed at breaking down barriers in silos between the Development and the Operation teams. Some practices such as the continuous integration and continuous delivery pipeline on built software is used to ensure that software releases are consistent and repeatable. The companies employ software tools including Jenkins and CircleCI to ensure that their development pace includes continuous integration. The operations procedures, such as configuration management and infrastructure deployments are automated with tooling such as Ansible and Terraform.

3.3. DevOps Practices and Tools

Practices to improve and streamline DevOps include the followings continuous integration, test automation, continuous delivery, continuous deployment and infrastructure as code. Adoption of those is based on a various assemblage of tools, where communication, team structuring and monitoring are critical to succeed. The DevOps tools chain facilitating these practices are Jenkins (automation), GitLab & GitHub (source control management), Maven (build tool), JIRA (issue tracking), Docker & Kubernetes (container management), Selenium (test automation) and many more [5].

As a practice that aims to align the development and operations arms of IT, DevOps attains its objectives by establishing a philosophy, culture and collection of tools meant to facilitate quicker deployment of new software. Its managing checklist, described in the corpus section Understanding MLOps, details the strategy required to develop and deploy software consistently and swiftly while maintaining quality, stability, and reliability. The practices and techniques described constitute a rough framework which these larger goals can fill.

4. The Intersection of MLOps and DevOps

MLOps as well as DevOps both enable Continuous Integration and Continuous Deployment (CI/CD). However, MLOps still has to face challenges such as model versioning, automated testing and other special features in the model deployment phase to achieve operational standardization like DevOps in software deployment. MLOps wants to accelerate the path of getting machine learning models into production, while DevOps wants to get new software functionality out quickly. Somehow, those compatible purposes include making fault detection and bottlenecks more visible earlier, resulting in better quality products sooner with shorter lead time for better products and services.

Why is MLOps interested in DevOps? CI is also one of the original DevOps paradigms which can be successfully applied in both data science and data engineering. By rapidly building and testing ML models, teams can ensure ML model quality. Although the eventual deployment mechanism and the possibility of continuous delivery can vary, the ability of CI to find problems earlier, and to distinguish them from changes in live business and customer envir onments, can enable data scientists and data engineers to under stand whether what you are doing, including model creation, creativity, and so on, is actually breaking.

4.1. Common Goals and Objectives

The convergence of MLOps and DevOps shares common goals and objectives that are absolutely necessary for efficiently building and deploying machine learning (ML) models [6-8]. They're both about getting to where you can deploy at high frequency with high reliability. MLOps' continuous integration pipelines: Model development and training is aided by new models being assembled in a fast and efficient way, while model testing is automated to prevent poor performing models from making it to production. These work together to enable ML work to easily be integrated into DevOps processes.

MLOps combined with CI/CD for pipelines additionally enables fast deployment of self-served production spaces and infrastructure maintenance. Versioning is another crucial aspect – in DevOps we kept versions of file repositories, VM / container images and artifacts for every release, but in MLOps we also need to account for versioning of models. This methodology keeps a record of modifications to ML models so that teams can systematically locate and replicate models built in production. The automated model testing also validates the model performance before deploying it.

4.2. Challenges in Integration

Many of the goals and principles of MLOps are similar to that of DevOps, yet there are unique challenges when it comes to integrating machine learning models into DevOps pipelines that do no exist for non machine learning algorithms or do exist and needs adaptation. When you start to deploy a model, for example, you need to consider what it means to use hosted APIs – that is, are you using REST APIs or gRPC-based frameworks, how you secure that in aggressive ways for production environments etc. Handling these considerations can be as complicated as choosing a cloud vendor.

Another critical concern lies in the maintenance of complex pipelines common in MLOps solutions. Although complex pipelines are often necessary to meet a project's needs, any deployment commitment should include continuous integration mechanisms that trigger the validation and testing of the model. These mechanisms maintain confidence over the operational statuses of both the pipeline and the model. Consequently, the foundational infrastructure should accommodate these continuous integration triggers, aligning with the philosophy of DevOps workflows.

5. Strategies for Effective Model Deployment

One of the biggest challenges of MLOps is model deployment. Here, the environment for training the model and validating its behavior can be different and cannot be fully replicated, especially for large ML models that interact with user data [9,10]. The evaluation of the model’s behavior is complex and requires real-time input from third parties like users, and hence is more time-consuming than software testing and quality check. Model deployment requires complex, rarely-used skills, is resource-intensive, and should be tracked continuously.

Model deployment refers to delivering the trained model into production and expecting it to perform on real-time data as per expectations. While the strategy varies across different ML models and business use-cases, many existing practices can be commonly adapted. Model versioning is a practice that involves versioning the training datasets, training codes, pipeline, model artifacts, and scores; it maintains a history of model changes, allows rollbacks to previous versions, and supports A/B testing in production. Automated model testing through unit testing, integration testing, and quality checks is critical for ensuring the model performs as expected. Automated testing also enables continuous integration, a DevOps practice where small code increments are integrated,

tested, and pushed to production rapidly and consistently using automation. This allows quality checks after every code push, resulting in faster and easier debugging.

5.1. Model Versioning and Management

An efficient model versioning and management process is a fundamental step towards achieving smooth MLOps and DevOps integration. These models, when developed, are usually stored using a version control system such as Git or GitHub. Just as the application's source code is stored in a repository, the models need to reside in another repository dedicated to storing the models. In this way, developers can track each model's development, examine the differences between models, and even share and communicate with colleagues in a decentralized manner.

A model stored with version control makes it fundamentally possible to develop an automated test process on the model. It is crucial to have the model and the application that call it stored in two separate repositories, since these elements should be tested in optimized ways: the control model cannot be cloned to the application repository or vice versa. For example, testing the model for classification accuracy requires considering accuracy, precision, recall, and F1 score, which can be measured only with a test dataset. Testing the application hosting inference requires methods such as stress-testing, where many users simultaneously call the API endpoint, or fuzz testing, where the model is called with nonstandard data inputs [11-13].

5.2. Automated Testing for Models

Automating model testing is a fundamental aspect of the MLOps lifecycle that helps maintain high-quality AI systems. Both manual and automated testing contribute to model test coverage, targeting scenarios that could jeopardize system reliability, performance, or behavior. Although it is impossible to guarantee a failure-free ML lifecycle through testing alone, thorough coverage of various failure scenarios significantly reduces such risks. Testing encompasses model accuracy using labeled datasets or real-world data through A/B or canary testing, new data bias testing by assessing confidence scores and data distributions, resource usage on end-user devices, service latency, and robustness against adversarial perturbations and sensitive feature influence.

Automated model tests support continuous delivery by providing an automatic pass/fail signal for model changes. Implementing these tests requires considerable engineering efforts and must be combined with other mechanisms, such as pipeline deployment conditions, to achieve a fully automated delivery

process. Different stages of the MLOps pipeline may be subject to varying automation levels; for example, a certified team can approve the configuration change that triggers pipeline execution, but subsequent testing and deployment should be fully automated to realize continuous delivery. As with any production system, well-defined monitoring and alerting strategies are essential during model serving to validate assumptions extracted from the training dataset in production settings.

5.3. Continuous Integration for Machine Learning

Machine-learning (ML) projects share substantial characteristics with software (SW) projects, yet present additional complexities. Projects that utilize the same SW components are generally affected by many of the same vulnerabilities and bugs. In ML projects, the same vulnerabilities and bugs could exist in workloads that share the same model and training dataset. Continuous integration practices for ML services require distinct testing practices beyond those used in continuous integration for SW services.

Machine-learning projects demand attentive deployment and integration approaches. Model versioning ensures that ML models are trackable and reproducible, facilitating rollbacks and debugging in production. Automated model testing can verify ML models against performance requirements and test for calibration, bias and fairness. CI4ML brings CI/CD to machine learning CI4ML applies CI/CD to machine learning workloads. Like continuous integration in SW development, CIs for ML need to be wired to the SW repository to kick off new training jobs whenever the relevant SW components are updated.

6. Integrating MLOps with CI/CD Pipelines

CI/CD are critical building blocks of the modern software development process, automating and monitoring the entire application lifecycle, from building and testing through to release and deployment and to operation. By enabling seamless, automated and rapid code testing, CI/CD pipelines bridge the gap between development and operations, and provide a streamlined movement into production.

MLOps practices are flexible and can dovetail with CI/CD to support development [2,14-17]. Model deployment strategies—such as model versioning, automated model testing, and continuous integration of machine learning models—form an integral part of this integration. Integrating MLOps

into CI/CD pipelines means following best practices and selecting the right tools and technologies from the many available in the DevOps ecosystem.

6.1. Overview of CI/CD Pipelines

Development teams use Continuous Integration and Continuous Delivery (CI/CD) pipelines to automate testing and deployment tasks. A CI/CD pipeline tests every commit and uses automation to create software releases. Using automation significantly enhances software quality and alleviates the challenges of manual deployment. Many models fail to reach production because data science teams struggle to put the models in a usable format, and ongoing retraining requires repeated manual effort. For machine learning, a CI/CD pipeline automates model deployment.

CI/CD pipelines commonly employ several best practices. Using a “one click” deployment strategy enables continuous delivery and continuous deployment, because the deployment can occur at any time, removing the dependency on manual intervention and reducing human error during the deployment process. Deployment can be done either with a virtual machine or container. The No Downtime design ensures that deployments don’t cause an outage or slow response from the web application or API [15-16]. Logging and Monitoring capabilities allow the developer to confirm the deployment of the new version and detect any bugs introduced during the deployment. Rollback capability provides the ability to revert to the previous production model that is known to be working when a bug is detected. Continuous Integration ensures any changes introduced through the deployment process are functional and maintain a high level of software quality. Secondly, the CI/CD pipeline is also expected to allow future scalability of the deployment process and should reduce the risk of an inefficient deployment procedure.

6.2. Best Practices for Integration

The two most important obstacles which we are trying to address, that prevent CI of the models, are the necessity to deploy the models (which is very different than microservice deployment) and CI context, for which special practices must be put in place in order to support model in CI pipeline [17]. Deployment Management Many other aspects are related to deployment management, e.g., model versioning, automated model testing, model monitoring. Model versioning involves assigning a unique identifier to each version of the model so that changes from one version to another can be efficiently tracked and the model may be easily rolled back to previous stages, if needed. Model testing means running the performance test and then automatically resetting the model when its

performance deteriorates. Model monitoring follows KPIs overtime to check that all seems in order.

Including model deployment in a CI pipeline demands a few practices that are still being neglected for it to run in a sufficiently smooth mode [18]. The integration of these practices in a CI pipeline is a prerequisite for a smooth integration of DevOps–MLOps. These may consist of model testing, model registry, and model metadata management, among other approaches. Model testing adds a stage in which a model, trained or machine learned beforehand, is tested before deployment to validate its performance. The model registry is a central store that stores and manages versions of registered models as well as their metadata and the artifacts they point to. Model metadata is the meta information of the model, and includes the model name, version number, framework that was used, and training data set information.

6.3. Tools and Technologies for CI/CD in MLOps

MLOps tools and technologies are developed to support continuous integration so that MLOps and DevOps can be integrated well. Of all of these facets of MLOps, there are two considerations that we should also focus on to help improve Model Deployment and seamlessly integrate into CI/CD:

1. Model Versioning: A fundamental point of the seamless integration between MLOps and DevOps is to make models versions manageable and shareable.

2. Automated Model Testing: Automatic assessment of the quality of a developed model improves the quality of the implemented models, and is yet another important cornerstone of integrating MLOps and DevOps[19-22].

7. Case Studies and Real-World Applications

Implementing MLOps within existing DevOps pipelines presents challenges, yet these difficulties can be mitigated by employing automated model training and testing capabilities before merging code into the main branch. Managing different model versions remains a significant aspect of MLOps. The sections on model-deployment strategies and MLOps-CI/CD integration best practices demonstrate how MLOps can be harmonized with a DevOps workflow by incorporating continuous integration.

Broader MLOps pipelines share much in common with DevOps pipelines. Both seek to automate the process from development to production for machine learning and traditional software development, respectively. Best practices for integrating MLOps encompass model versioning within source control systems, automated model testing prior to production integration, and continuous integration mechanisms designed for machine-learning environments. These guidelines are further elaborated in the subsequent discussion on CI/CD integration [23].

7.1. Successful Integrations of MLOps and DevOps

The concepts MLOps and DevOps aim for similar goals and share many aspects and core principles, but for achieving swift put into production of ML models not only the MLOps practices and integrated solutions towards model creation are needed, but also an adaptation of current DevOps processes towards the needs of ML is essential. DevOps is a software development practice that unifies software development (Dev) and software operation (Ops). The main goal of DevOps is to shorten the system development process while delivering high-quality software continuously. Infrastructure as Code (IaC), continuous integration (CI), continuous testing (CT), continuous delivery (CD), and continuous monitoring (CM) are common general practices of DevOps. Further practices are security integration and automated alerting among others. There are many tools that simplify and fully automatize DevOps processes. Among the most popular tools for the crucial CI/CD phases are Jenkins, GitLab, Jenkins X, Tekton, Drone and Spinnaker. It is a common approach to provisioning and managing infrastructure as well as packaging and releasing software to use Kubernetes-based solutions. Kubernetes (also called k8s) is a platform for the automatic provisioning, scaling, and managing applications that are containerized. This functionality is provided directly by Kubernetes or additional tools and services such as Helm and Kustomize.

7.2. Lessons Learned from Industry Leaders

DevOps teams still strive to extend their practice to machine learning and AI models. Besides companies like Google, Uber, Facebook, Amazon, and Microsoft, who explained their workflows and extracted lessons from implementing MLOps in production, other influencers and ambitious companies also shared their experience [24-26]. Christian Falch from Microsoft Denmark wanted to show the top three lessons learned from launching their MLOps practice within Microsoft for the first time, considering the audio intelligence domain. Engaging in a real-time use case that involved automating audio content moderation for industrial-scale messages within Microsoft Teams, resulted in an

automated objectionable content moderation (OCM) in Microsoft Teams called Talking Cleaner. The framework was first made available in preview and was in the process of being launched in Mule.

8. Future Trends in MLOps and DevOps

Analyzing the future trends of MLOps is another way to solve the model deployment challenge. MLOps is the discipline of continuously creating and improving machine learning models in production. The perfect MLOps pipelines will handle all possible use-cases and scenarios and generate a trajectory of the different test-cases. Develop MLOps pipelines like DevOps pipelines to avoid production issues on machine learning models and for easier deployment. In MLOps, the risk of failure is--fer with CI.

DevOps is a culture and practice in which development and operations (ops) work together whole-cloth so that development and operation teams are involved throughout the product lifecycle in order to expedite the process from inception to realization. It includes software development, testing, and operations and is supported by automation. When you use a DevOps pipeline for model deployment, you create an MLOps pipeline, which incorporates CI/CD into MLOps flows. Using models in a correct way and implementing a CI catering to Machine learning methodologies plays a vital role to have continuous flow.

8.1. Emerging Technologies and Innovations

In recent years we have witnessed great advances in MLOP, DevOps and CI/CD tools which are crucial for the automation of slow and error-prone manual product deployment processes. CI/CD allows a company to stay current with software updates, as it automates the majority of the work.

As an increasing number of Intelligent Systems have been evolving based on Machine Learning (ML), MLOps has become trendy. Current MLOps practices have a CICD pipeline and model monitoring mechanisms. These methodologies inspired by DevOps workflow are designed to unify the operations between different departments involved in ML development. Predictive and preventive maintenance also garnered interest; nonetheless, in ML projects, continuous integration plays a pivotal role in model stability and drift control, facilitating the model DevOps phases[27].

8.2. Predictions for the Future

The latest developments in cloud computing further enable the integration of MLOps and DevOps, opening the door to greater efficiencies and more finely tuned ML models [20]. For many developers, these growing possibilities generate excitement, but also raise a crucial question: When will MLOps finally be integrated into DevOps to support continuous integration for ML—just as DevOps supports continuous integration for software?

Predictions in the market are beginning to emerge. As the list of Level 3 and Level 4 solutions on the MLOps Landscape continues to grow, so do the expectations—and exigences—placed on these controllers and orchestration layers. Advances in these highly specialized solutions will soon permit data scientists, data engineers, and ML engineers to manage an increasing number of common DevOps workflow patterns within their MLOps tooling. Embracing the shared philosophy of continuous integration, ModelOps is expected to become just another flavor of DevOps, with the many related best practices suitably adapted for AI/ML/DS projects.

9. Conclusion

MLOPs, frequently described as the "DevOps of ML," focuses on deploying and maintaining machine learning models in production reliably and efficiently. DevOps extends beyond development and operations to include managing infrastructure and controlling every aspect of the software environment. While DevOps concentrates on delivering production-ready software to end-users, MLOps focuses on continuously training and delivering the best possible machine learning model to the production system. Model deployment in ML requires a different approach because, unlike software development, the model isn't ready at the outset; it is continuously evolving. Each trained model varies in accuracy, f1 score, recall, and precision, and the best model must be delivered to production. Automated backbone support is crucial, early in the development cycle, to compare models and deploy the best one.

Model deployment can be streamlined using principles such as model versioning, automated model testing, and continuous integration. Continuous integration provides the backbone for a robust MLOps pipeline and allows for smoother integration with standard DevOps practices. Integrating MLOps within a high-performance CI/CD pipeline enables a robust training pipeline and endorses seamless model deployment. The objectives of MLOps and DevOps are aligned;

training a model or developing a new feature are both code changes aimed at enhancing the product. Machine learning projects will increasingly rely on effective methodologies for model deployment and continuous integration/continuous deployment.

References

[1] Bonner E, Lege R, Frazier E. Large Language Model-Based Artificial Intelligence in the Language Classroom: Practical Ideas for Teaching. Teaching English with Technology. 2023;23(1):23-41.

[2] Rao AS, Vazquez JA. Identification of COVID-19 can be quicker through artificial intelligence framework using a mobile phone–based survey when cities and towns are under quarantine. Infection Control & Hospital Epidemiology. 2020 Jul;41(7):826-30.

[3] Shivadekar S, Halem M, Yeah Y, Vibhute S. Edge AI cosmos blockchain distributed network for precise ablh detection. Multimedia Tools and Applications. 2024 Aug;83(27):69083-109.

[4] De Silva D, Alahakoon D. An artificial intelligence life cycle: From conception to production. Patterns. 2022 Jun 10;3(6).

[5] Battina DS. DevOps, a new approach to cloud development & testing. International Journal of Emerging Technologies and Innovative Research. 2020 Aug;2349-5162.

[6] Maheshwari A. Digital transformation: Building intelligent enterprises. John Wiley & Sons; 2019 Sep 11.

[7] Subramanya R, Sierla S, Vyatkin V. From DevOps to MLOps: Overview and application to electricity market forecasting. Applied Sciences. 2022 Sep 30;12(19):9851.

[8] Dang Y, Lin Q, Huang P. Aiops: real-world challenges and research innovations. In 2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings. IEEE; 2019 May 25. p. 4-5.

[9] Liu J. Web Intelligence (WI): What makes wisdom web?. In IJCAI. 2003 Aug 9;3:1596-1601.

[10] Maddox TM, Rumsfeld JS, Payne PR. Questions for artificial intelligence in health care. JAMA. 2019 Jan 1;321(1):31-2.

[11] Paschen J, Kietzmann J, Kietzmann TC. Artificial intelligence (AI) and its implications for market knowledge in B2B marketing. Journal of Business & Industrial Marketing. 2019 Oct 7;34(7):1410-9.

[12] Myllynen T, Kamau E, Mustapha SD, Babatunde GO, Collins A. Review of advances in AI-powered monitoring and diagnostics for CI/CD pipelines. International Journal of Multidisciplinary Research and Growth Evaluation. 2024 Jan;5(1):1119-30.

[13] Etzioni O, Cafarella M, Downey D, Popescu AM, Shaked T, Soderland S, Weld DS, Yates A. Unsupervised named-entity extraction from the web: An experimental study. Artificial Intelligence. 2005 Jun 1;165(1):91-134.

[14] De Silva D, Alahakoon D. An artificial intelligence life cycle: From conception to production. Patterns. 2022 Jun 10;3(6).

[15] Battina DS. DevOps, a new approach to cloud development & testing. International Journal of Emerging Technologies and Innovative Research (www. jetir. org), ISSN. 2020 Aug 8:2349-5162.
[16] Maheshwari A. Digital transformation: Building intelligent enterprises. John Wiley & Sons; 2019 Sep 11.
[17] Subramanya R, Sierla S, Vyatkin V. From DevOps to MLOps: Overview and application to electricity market forecasting. Applied Sciences. 2022 Sep 30;12(19):9851.
[18] Dang Y, Lin Q, Huang P. Aiops: real-world challenges and research innovations. In2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings (ICSE-Companion) 2019 May 25 (pp. 4-5). IEEE.
[19] Fitsilis P, Tsoutsa P, Gerogiannis V. Industry 4.0: Required personnel competences. Industry 4.0. 2018;3(3):130-3.
[20] Mohapatra PS. Artificial Intelligence and Machine Learning for Test Engineers: Concepts in Software Quality Assurance. Intelligent Assurance: Artificial Intelligence-Powered Software Testing in the Modern Development Lifecycle. 2025 Jul 27:17.
[21] Erich FM, Amrit C, Daneva M. A qualitative study of DevOps usage in practice. Journal of software: Evolution and Process. 2017 Jun;29(6):e1885.
[22] Qumer Gill A, Loumish A, Riyat I, Han S. DevOps for information management systems. VINE Journal of Information and Knowledge Management Systems. 2018 Feb 12;48(1):122-39.
[23] Panda S. Scalable Artificial Intelligence Systems: Cloud-Native, Edge-AI, MLOps, and Governance for Real-World Deployment. Deep Science Publishing; 2025 Jul 28.
[24] Almeida F, Simões J, Lopes S. Exploring the benefits of combining devops and agile. Future Internet. 2022 Feb 19;14(2):63.
[25] Battina DS. An intelligent devops platform research and design based on machine learning. training. 2019 Mar;6(3).
[26] Battina DS. Ai-augmented automation for devops, a model-based framework for continuous development in cyber-physical systems. International Journal of Creative Research Thoughts (IJCRT), ISSN. 2016 Sep 3:2320-882.
[27] Enemosah A. Enhancing DevOps efficiency through AI-driven predictive models for continuous integration and deployment pipelines. International Journal of Research Publication and Reviews. 2025 Jan;6(1):871-87.

Deep Science Publishing, 2025
https://doi.org/10.70593/978-93-7185-060-5

DeepScience
Open Access Books

Chapter 9: Harnessing Artificial Intelligence to Advance Site Reliability Engineering

1. Introduction

Several prominent companies are using Artificial Intelligence (AI) tools in their Site Reliability Engineering (SRE) efforts because these tools facilitate and automate the detection of performance degradation and production incidents. The support of these tools with AI can help to reduce costs and increase site reliability. The essential characteristics of Site Reliability Engineering are: an engineering discipline aimed at managing IT systems—especially large-scale operation and maintenance—maintaining reliability, and building automation tools in operational services. SRE is a relatively new technology that emerged after the founding of Google in 1998. The rapid development of AI has also benefited SRE. As a result, multiple related tools have been combined with Artificial Intelligence, forming what is called AI for Site Reliability Engineering.

The comparative study of Artificial Intelligence tools for SRE examines four different tools employed by four well-known companies. It addresses the question: «What are the key Artificial Intelligence tools for Site Reliability Engineering and how are they currently utilized?» The goal is to follow the continuous development of AI technologies within SRE over the long term. The criteria for selecting the tools focus on their AI capabilities, recent integration with Site Reliability Engineering, practical application, and accessibility of public information. Four leading companies that are actively involved in recent AI developments in SRE satisfy these requirements and are therefore included in the discussion.

2. Background of Site Reliability Engineering

Thorough and efficient code-development is of utmost importance as it forms the foundation of compelling applications. Focusing on speed and scope has become integral in software development, along with monitoring the development process and ensuring quality to deliver a good product. Both tech giants and startups invest heavily to track the health and performance of their service, and to find out when something goes wrong, what happened, and why. Monitoring can be outsourced, to save both time and menial work, adding in a few man-months to the development process while keeping the costs down [1-3].

Site Reliability Engineering (SRE) is fundamentally doing work on the operation of software systems. SRE forecasts system and service performance on the basis of real time and historical data, which becomes cheaper in the long run. The impact of AI tools in SRE: automated forecasting of system outages The use of Artificial Intelligence in SRE has accelerated the pace of automation of predicting system outages and has left a lasting impression on the industry. There are two goals that can be achieved by the use of AI tools to predict service degradation or outage and cost savings. The good performance of these tools in handling outages, incidents, or flaks has led the study to be deep further d, including: the background of SRE and AI, key AI tools of several companies, and a real case study on how some leading companies use them.

2.1. Definition and Importance

Site Reliability Engineering (SRE) is a software development discipline focused on building large scale, high availability software systems. SRE teams work behind the scenes to balance system reliability with the pace of new developments. Without their contributions, costly outages, dissatisfied users, and damaged reputations would result. A study of 3,500 IT decision makers in companies with 1,000+ employees revealed that more than half (54%) of organisations are concerned about the impact of IT operations on their bottom line. A third (35%) worry about how incident management affects customer satisfaction and retention. Additionally, 41% are concerned about the technical expertise gap, and 35% cite insufficient resources for incident management as a major issue.

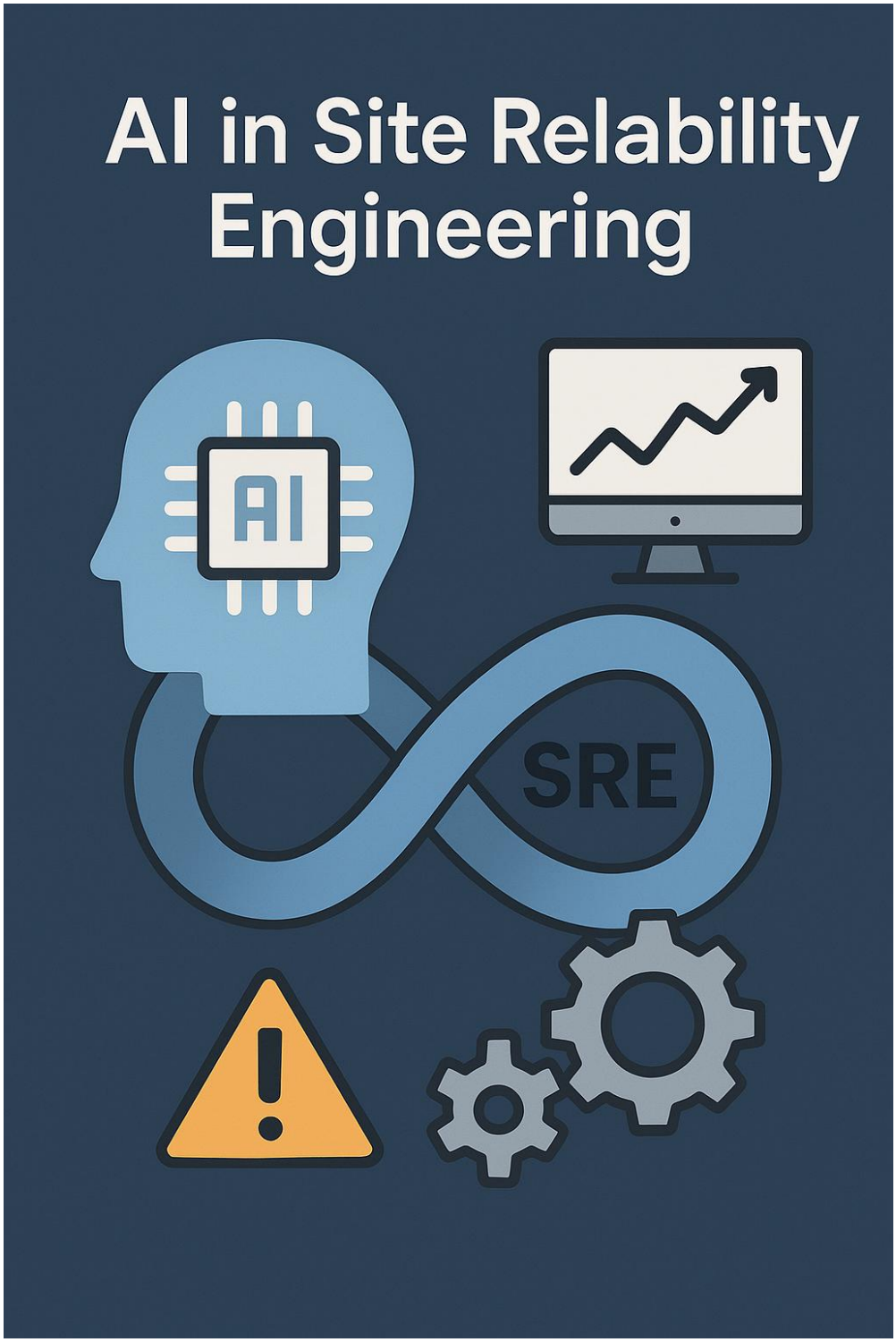

Fig 1. AI in Site Reliability Engineering

In 2022, the global AI in IT Operations (AIOps) market size reached approximately USD 11.58 billion and is expected to expand at a compound annual growth rate (CAGR) of 30.2% from 2023 to 2030. When implemented properly, AI tools can enhance site reliability by reducing downtime, enabling complex system management without constant supervision, and significantly cutting operational costs. One organisation successfully deployed AI models internally for incident management, achieving a 99% accuracy rate in incident assignment and a 70% reduction in service incident resolution time, thereby freeing the SRE team to concentrate on higher-value tasks.

2.2. Evolution of SRE Practices

The evolution of SRE practices reveals a natural progression driven by technological advances alongside a focus on efficiency, quality, reliability, and cost-effectiveness. In 2016, Keyvan Rahbari Akbari et al. observed that efficient IT operation is a major challenge within the industry and that: "It is believed that automation is one of the promising emerging technological approaches for future IT Operations that can provide a number of benefits such as improving system performance, reducing operational costs and shortening time to repair." Moreover, Tan D. Phu et al. pointed out that while SRE is a proven way to create stable and reliable cloud services, new growth in cloud infrastructure and services brings new challenges in handling incidents cost effectively. They concluded that processing operational incidents can be substantially automated through deep neural networks, which offer benefits such as improving system performance, reducing incident processing costs, and shortening time to resolve.

SRE is closely related to the broad topics of IT automation and IT operations. Although there may be many areas where automation can be used, intelligent automation is considered the final stage of automating IT operations, addressing the complex nature of IT Operations. Denise Yu et al. confirmed that Google relies heavily on SRE—the discipline of applying software engineering principles to operations—and uses automation to keep production and development running smoothly [2]. They found that "Automation is the final stage of IT operations automation, where IT can be largely self-managed and managed proactively," and that such intelligent automation is able to handle tasks that are repetitive, routine, and rule-based, as well as those that require a greater degree of intelligence, such as incident handling and resolution.

3. Artificial Intelligence in Engineering

Artificial Intelligence (AI) denotes a general method of engineering that contributes, streamlines, and improves many other engineering tasks. More recently, many Site Reliability Engineering (SRE) teams have started to integrate some AI features into their workflows; however, little is known about the level of adoption nor the selection process for these tools. Therefore, a comparative investigation of AI tools in SRE has been conducted. Ten companies that have developed a tool with reported Site Reliability Engineering use cases were identified, and relevant features were extracted. The analysis reveals the impact of AI on Site Reliability Engineering in mitigating operational costs, improving system performance, and facilitating incident management. Moreover, the study systematizes key aspects before and during the setup of AI features and explores future trends associated with these developments.

The background of Site Reliability Engineering covers categories of aspects that any SRE practice need to consider, including: the transition from traditional operations to SRE; structured analysis and problem-solving; monitoring strategy; demand forecasting; capacity planning and management; workload management; cost control outcome review; business impact evaluation; post-incident review; and the incident lifecycle. Furthermore, it highlights the importance of preparing the team and achieving cultural consensus. The discussion of Artificial Intelligence in Engineering complements the comparative analysis and case studies of implementation in leading companies.

3.1. Overview of AI Technologies

Artificial Intelligence (AI) encompasses technologies within the engineering sector that work to solve problems with the aid of corresponding techniques. The technologies identified as part of AI include Deep Learning, Machine Learning, and Big Data Analytics. AI can also be characterized as a branch of computer science designed to generate systems with human characteristics.

AI is designed with a primary objective of enhancing efficiency in processes, including engineering operations. Services that utilize AI technologies strive to operate at an Unreliable Cost Point (UCP), aiming for maximized services while minimizing expenses [2,4,5]. A Real Self Cost Point (RSCP) is identified where services can be optimized by lowering costs, increasing performance above the base-case cost point, or balancing both. The strategy for site reliability companies that employ AI begins with personnel training, subsequently progressing to testing, implementation, monitoring, and continuous maintenance upon successful deployment.

3.2. Benefits of AI in Engineering

While dangers such as algorithmic bias clearly exist, AI offers a large supply of benefits. Firstly, the general use of AI allows people to focus more on complex tasks. More specifically, in the world of engineering, AI can be used in a variety of engineering fields to decrease cost, save time, improve quality, and enable more intelligent decision-making.

The examples listed above in the areas of electrical engineering, mechanical engineering, civil engineering, and software engineering showcase how AI technologies are applied in engineering, thus mitigating the risk of people focusing too much on repetitive, complex, and boring tasks.

4. Comparative Analysis of AI Tools

A selection of AI tools implemented within the engineering domain is reviewed, followed by a comparative discussion of the most pertinent technologies for introduction into Site Reliability Engineering. SRE employs software to automate manual activities performed by operations engineers. Techniques for automating rules and heuristics have existed since the 1980s, but current widespread adoption is partly due to the progress of AI. The advent of Deep Learning, capable of generating models from experience without explicit human guidance, is instrumental in this development [6-8]. Hoori et al. are among the pioneers assessing the impact of Deep Learning in SRE and demonstrating how it can significantly expedite the technology's evolution in this sector.

Informed by categorisation criteria extracted from the literature, a comparative analysis of AI tools is developed to select those most appropriate for SRE. At present, several prominent companies have publicised their implementations of SRE systems based on artificial intelligence. However, the admission documents for specialist positions indicate that the field is still in its infancy, as evidenced by the limited range of cases on which conclusions have been drawn. Nevertheless, the results achieved are highly promising. Assessing the effects of applying AI in SRE reveals enhancements in cost reduction, system performance, and the simplification of incident resolution and service restoration.

4.1. Criteria for Tool Selection

Before comparing the leading AI tools in Site Reliability Engineering (SRE), it is important to understand the selection criteria used for identifying current AI

tools. A selection of tools offers an explanation for the criteria employed. The requirements will subsequently be used for a comparison of AI tools for SRE.

A SRE working definition proposes the problem space in which AI can support the SRE efforts. Ultimately, AI has the potential to transform multiple facets of SRE. Some of the signs already present about the potential of AI are areas of applications of AI and current success stories and existing challenges.

4.2. Leading AI Tools in SRE

Site Reliability Engineering (SRE) focuses on tasks such as maintaining systems, balancing reliability and latency, and minimizing costs, with incident management playing a crucial role. Exploiting artificial intelligence in these key areas can provide significant advantages, a potential now acknowledged by the majority of leading companies. Nevertheless, implementing adequate AI tools remains challenging, prompting an examination of the tools companies are currently employing. A comparative analysis of two leading AI tools applied in SRE provides further illumination.

The selection of these tools rested primarily on their usage by Google's Site Reliability Engineers. Originating from an internal Google project, the first tool requests Slack workspace access to retrieve conversations and search for answers, invariably replying based on the available informatio [9,10]n. The second tool addresses both virtual and physical server troubleshooting. In case of unavailability, it accepts analysis requests and launches investigation jobs for deeper examination. Both ChatOps AI tools leverage the company's conversational database to enhance the incident management process.

5. Case Studies of Leading Companies

Suspicion that AI can change roles and responsibilities has boosted the implementation of AI tools for IT operations. Site reliability engineers, typically burdened with examining alert logs, analyzing key performance indicators, and performing on-call duties, now explore AI tools to alleviate these redundant tasks. With proper implementation, AI-assisted systems promise improved system availability, minimized alert volumes, reduced operational costs, and accelerated incident resolution.

A clear understanding of AI’s capabilities, the currently tested AI tools in engineering, and a comparison of these tools with future developments are necessary for effective adoption. SRE is the practice of engineering for IT

operations, while artificial intelligence is a collection of advanced technologies that can perform functions generally requiring human intelligence [11-13]. AI's role in engineering is to aid in automating routine, tiresome tasks. Analyses of AI technologies and their benefits in engineering reveal that several companies have integrated AI into their SRE operations to achieve higher SLO attainment and enhanced incident management. A study of these companies and the AI tools they presently employ underscores both the advantages of AI-assisted systems and their future growth trajectories.

5.1. Company A: Implementation of AI Tools

Site Reliability Engineering is a practice that incorporates aspects of software engineering and applies that to operations. The main goals of an SRE team are that services that are delivered to the customers are highly reliable, highly available, and highly efficient, among other things. Services could range from running Frontend APIs that are accessible to the client to Interaction with Databases to fetch or push data, to any other feature. Artificial intelligence refers to the simulation of human intelligence in machines. The credibility and usefulness of predictions in the field determine the degree to which these predictions are self-fulfilling. In selection functions, as in many other engineering tasks linked to the development of new products attaching AI, benefits are readily apparent.

Selected AI tools for implementing SRE have been briefly analyzed and examined. Finally, based on a set of criteria, a comparison of leading tools in the market for easy implementation by any company is made. Every sector in the 21st century is leaning towards automation or the use of Artificial Intelligence to lessen the burden. Site reliability, crucial for maintaining product functionality across services, also needs to abide by these tenets of automation [2,14-17]. SRE, or Site Reliability Engineering, is a sophisticated solution that helps reduce downtime, mitigate adverse consequences during outages, and maintain an efficient infrastructure while working in the ground-level framework. AI implementation in SRE has already begun in companies. A few of these companies along with the AI tools chosen, look as follows.

5.2. Company B: Challenges and Solutions

One of the main challenges facing the team in Company B is that AI tools cannot detect subtle conditions such as latent warnings that indicate an imminent disaster. Detection is limited by the set of labeled data submitted to the AI model. In production environments, a human engineer is better able to perceive the faint symptoms of an imminent disaster even when the tool does not generate a warning about it. Another limitation of the AI tools is the generation of false

positives. Anything that is labeled as an incident must be thoroughly analyzed and deselected if it is a false positive. The analysis process is resource-intensive. The impact of incident management tools also poses a limitation for Company B: the cost of the tooling, especially during an incident, can easily exceed the cost of hiring site reliability engineers. Because incident management tools generate additional logs to monitor the efficiency of the incident, the higher volume requires more infrastructure resources which, in turn, increase costs. The efficiency gained during the incident does not always generate a positive cost/benefit relationship for the company.

5.3. Company C: Achievements and Metrics

Company C discovers indicators of high-severity conditions early in the escalation pathway by detecting unexplained application latency through analytics. When possible, associated services are isolated in new deployment environments to facilitate problem resolution.

Particular attention is directed to the generation of effective performance and availability metrics. The combination of machine learning and statistically based pattern recognition enables the identification of early signals relevant to availability, permitting hierarchical grouping for at-risk/service-level-violating services. Although capacity optimization during high-impact events remains under development, significant attention is devoted to analyzing root causes of outliers. Early traffic routing actions have yielded mixed results, yet considerable value is realized by optimizing on-call scheduling to reduce costs.

6. Impact of AI on Site Reliability

Artificial Intelligence (AI) is widely recognized for its beneficial impact on many engineering areas. The ability to assist in predicting system behavior, planning solutions, forecasting costs, and analyzing complex problems using deep learning algorithms is highly valuable in Site Reliability Engineering (SRE). AI can enhance system performance, reduce maintenance costs, and accelerate the development of needed technology. By enabling companies to use human resources more efficiently, AI assists in managing systems and high-risk incidents occurring 24/7 and in processing substantial data volumes.

Many companies, including Google, Microsoft, Meta, Splunk, Moogsoft, and Datadog, invest in AI-based tools to support their Site Reliability Engineers. An overview of AI tools implemented in these companies and organizations, along

with their functionalities and capabilities, provides insight into the influence of artificial intelligence in Site Reliability.

6.1. Performance Improvements

Artificial intelligence is evolving rapidly, transforming numerous fields, including engineering [9,18-21]. Its application in Site Reliability Engineering (SRE) promises significant performance improvements in user experience and cost reduction. Leading companies are already using various AI tools and practices in their SRE-related activities.

SRE applies software-engineering approaches to IT operations. A comparative study between the main AI tools employed in SRE reveals considerable benefits and opportunities, as well as challenges remaining to be resolved. Several case studies illustrate how leading companies have adopted these tools to tackle different SRE tasks.

6.2. Cost Efficiency

Artificial Intelligence can assist automation in the three disciplines of SRE. Moreover, it can make existing tools more intelligent, enabling the reduction of operational costs in IT. Operational costs represent a significant portion of IT budgets, and companies consequently seek methods to optimize their spending. Cloud providers facilitate this optimization by offering on-demand machine provisioning, scaling CPU, memory, and bandwidth resources according to predefined and customizable rules [22,23].

Cloud providers such as Amazon, Microsoft, and Google furnish their customers with tools for managing and provisioning machines. However, these tools require supervision to avoid errors and to identify any opportunities for cost reduction. In the context of SRE, these aspects become critical.

6.3. Incident Response Times

Once the system moves to incident handling, the system identifies the priority of the problem and collects the incident data and history. It may notify the responsible SRE team and select the action based on the predefined policies and the previous actions taken in similar situations.

By analyzing the incident history and the available playbooks, the system may suggest fixing actions to the SRE team.

7. Future Trends in AI for SRE

Following the trends of recent years, thanks to breakthroughs in AI, the adoption of AI technologies in Site Reliability Engineering seems unstoppable. Recent approaches to Enterprise AI have argued that AI technologies will evolve over the next decade.

Despite the impressive results achieved recently, AI technologies have not been integrated fully into IT operations. Recent research has divided the evolution of SRE into three eras, from a focus on cost reduction to incident prevention, and finally to smaller models that enhance user experience. By analyzing successful experiences of top companies, some challenges remain, along with recommendations for their resolution.

7.1. Emerging Technologies

Artificial intelligence researchers and developers have been making significant advances in Generative AI related technologies and products. These advances lead researchers to consider other sectors of engineering and associated novel uses of AI. Site Reliability Engineering (SRE) is a critical discipline that focuses on developing and deploying a highly scalable and reliable online IT infrastructure. Over the years, SRE teams in organizations have adopted automation and AI tools for various aspects such as requirements elicitation, design, development, testing, operations, and production support. The exploration of the use of emerging generation AI tools for SRE is a natural evolution of this process.

Emerging technologies that are useful in the field of Artificial Intelligence (AI) are creating several benefits for SRE operations. Some of these enabling technologies that impact the use of AI for Site Reliability Engineering include big data technologies, high power and low cost computing platforms, emerging AI product tools, and innovative solutions offered by major SRE organizations such as Google, Microsoft Azure, and SRE communities like SRE Consoles and DevOps. The promising opportunities are evident from operations and performance successes at leading companies. These companies have successfully implemented AI tools and demonstrated tangible benefits such as cost reduction, better incident reporting and analysis, appropriate alerts, and timely resolution of possible or real incidents [24-26].

7.2. Predictions for the Next Decade

Machine learning (ML) algorithms are progressively enabling the automation of all routine and costly processes in organizations. The rapid progress in SRE

remains evident as AI continues to pave new pathways and redefine organizational practices. The future evolution of AI, particularly for SRE applications, is poised to unfold in an exciting manner. Understanding these future trends aids practitioners in discovering an even broader range of applications in the next decade. AI technologies—machine learning, computer vision, natural language processing, and data mining—are continuously offering new ideas and solutions to improve the SRE process.

The surveyed companies have also identified some future challenges in the use of AI technologies and in their effective deployment in the real world. AI-based tool which suggests workload placement and better fault management in accordance with the importance of a given service. But when you go to put an AI tool into production it requires rigorous validation and testing to assess their impact on service availability and to mitigate accordingly." Regularly check AI tools and logs to reduce the risk of downtime caused by incorrect recommendations made by AI [27,28]. Furthermore, training squads to decipher AI operators and their output also helps interpret results correctly. Human judgement is still essential for detecting and halting system failures that are not anticipated by the system (cf. When technical staff have no domain experience and have been trained to use AI tools in a certain way, it can result in risks, highlighting the necessity of human involvement in incident response. The AI services should be available for use in SRE in an open mannerWith AI technology, the use of AI technology in SRE is a new thing, and there is no clear method in how to develop service operation and find the human resources, and how to develop user demand and update the system periodically.

8. Challenges in Implementing AI Tools

Deployment of AI applications into CI presents novel challenges and threats. The issues of staffing are one of the main impediments in this regard. Because SRE can be an extremely high-stress role as-is, there’s real danger of burnout. Favouring a proactive culture, organisations need to invest in staff training and understanding to ensure the tools are maximised. Instituted employees are central to AI deployment: They are the most fit to plan holistic monitoring schemes that decrease accidents with AI in use, and to periodically assess tools’ efficiency during and after integration.

In addition, a backlog of follow-ups adds significant exposure. If the operation of AI tools repeatedly generates surplus tasks—such as alert remediations or incident resolutions—organizations may incur negative workload consequences. Whether AI tools ultimately enhance or diminish operational capacity depends on the scope of their deployment and the organizational strategies employed to address such challenges.

8.1. Technical Limitations

Technical Limitations

Artificial intelligence has simplified modern life in numerous ways. For example, when systems collect metrics such as Internet traffic, coverage of an AI-based system can be wider and more comprehensive than that of humans. However, AI tools also present limitations. There is no method to determine 100 percent whether the results generated by machine learning algorithms are precisely accurate. The AI techniques also come bundled with a sample generation bias, thereby skewing the results. To maintain the power of such tools beyond accident prediction, it is crucial to explore the best approach for determining the accident stage even before the crash occurs. SRE teams face challenges in designing and implementing AI tools capable of generating alerts about potential future incidents to enhance site reliability.

Engineering relies heavily on data; the absence of data impedes the development of new solutions through AI. Although AI has been utilized in various other industries, its integration in SRE remains limited. Nevertheless, leading companies such as Google, Microsoft, Meta, and Oracle have made significant progress in developing AI solutions to realize the benefits of AI in SRE. They have successfully addressed numerous challenges, and their tools demonstrate a positive impact on cost savings and site reliability.

Exploring the best AI tools for SRE and understanding their practical impact on site reliability involves examining several pivotal questions. The selected AI tools can be categorized based on their underlying AI technologies—for instance, employing machine learning, deep learning, or natural language processing in AI operations. Analysing tools from the four companies reveals implementation strategies for AI and the inherent challenges encountered during deployment. Furthermore, the overall impact of these tools on site reliability, as experienced by the companies, sheds light on the advantages and remaining obstacles of AI integration in SRE [19,29].

8.2. Organizational Resistance

Soon, only a small fraction of the SRE companies in the world will be able to take advantage of the benefits of AI in Site Reliability Engineering because organizations are typically slow to adopt new technologies and processes in mission-critical areas such as IT operations and Site Reliability Engineering. For decades, IT organizations have used ad-hoc automation to reduce the toll of "boring and repetitive work," to improve availability, and to manage cost, but repeating the same processes or coding similar scripts has been far more efficient than redesigning organizations and operations.

Implementing AI-powered tools can be a solution, but cultural barriers exist in these organizations. Implementing such tools requires shifting the role of an SRE or administrator from capacity management or firefighting for a project to training automated tools for these responsibilities. This organizational change requires dedicated staff who know how to train AI tools and may encourage organizations to pursue increased automation only with their existing teams.

8.3. Ethical Considerations

The ethical considerations of artificial intelligence in engineering are of paramount importance. As designers of engineering systems, engineers must anticipate and understand the ethical consequences of those systems. Grady Booch has summarized it succinctly: so much of our modern society rests on the infrastructure and processes that engineers build that nearly every human on the planet is at the mercy of the engineers, whether they realize it or not. Engineering systems are rarely failures of designed systems; most often they are failures of the processes that govern the design and implementation of those systems. By incorporating such ethical considerations, SRE practitioners can go beyond the purely technological aspects of site reliability engineering. This viewpoint recognizes the suite of technologies—automation, machine learning, artificial intelligence, cloud computing, and big data—that continue to evolve. It acknowledges the significant impacts that AI-based applications have had on our society and culture with machine vision, machine translation, artificial creativity, and board game–playing. Additionally, it recognizes that deploying AI in SRE can profoundly affect the cost and reliability of the services, the culture of the operations staff, the speed of incident detection and resolution, and the level of automation of the entire incident management lifecycle.

Ethical considerations in AI for engineering encompass the future evolution of capabilities, the application in specific real-world domains, and the means to achieve the benefits without causing harm. The ongoing success of a company in highly competitive markets will rely more heavily on the application of artificial

intelligence than ever before. Ethical responsibilities must be addressed through training, staffing, governance, methods, tools, and technologies. Training must instil an understanding of AI's capabilities and limitations. Staffing must ensure sufficient skills and expertise to design, develop, deploy, and maintain AI-based solutions responsibly. Governance must consist of appropriate controls, oversight, and monitoring to manage ethical risks. Approaches should offer repeatable and predictable methods of engineering trustful AI apps. Tools need in-built presets/settings so that something not-safe-that-could-kill-you doesn't convert accidentally. Resilience Technologies should be fail-safe and fail-operational, that is ensure acceptable systematic behaviour, even in face of faults. These guiding principles are the key to successful enterprise roll-out of AI-powered applications for SRE teams.

9. Best Practices for AI Integration

Best practices vary from technology aspect to organizational and human aspects. Many authors claim that staff performing SRE should be certified in AI technologies and have experience in operation of Devops tasks, to make correct use of AI tools. Also key will be predicting the next AI and machine learning trends. Recommendation: Leverage Your AI Tools Effectively Including AI in the SRE toolkit can help optimize your system's performance, save money, and decrease the number of incidents. AI in SRE should be implemented following best practices in order to mitigate risks and pitfalls during implementing and deployment phase.

The rise of AI technology has transformed some engineering disciplines by providing automation of numerous engineering activities and tasks. AI is establishing organizations that learn faster, and are more productive, effective, and competitive. As AI systems become more advanced, many new experts are entering the field, creating new knowledge and experience. Site Reliability Engineering (SRE) applies software engineering techniques to information technology operations problems. Being responsible for long-term business effect, a company focuses on ensuring new technologies improve operations. Using machine learning in business can predict the future and help anticipate costs and build better road maps. Leading companies, such as Google, Netflix, American Express, and Capital One, have developed AI tools to simplify SRE processes.

9.1. Training and Development

Even with the assistance of advanced AI systems, continuous training for SRE personnel remains essential. Proper preparation enables staff to make appropriate use of the technology and obtain the best results. Newly hired and junior

engineers will likely require ongoing training to become fully operational and effective within the organization's specific environment and processes. Experienced engineers should also receive continued training to stay updated on procedural modifications and technological updates.

Companies should develop sound training programs for current and incoming engineers[23-25]. Additionally, all employees should be encouraged to participate in such sessions."Training and development are among the main factors leading to high levels of job satisfaction. Employees identified that confidence levels and career prospects can be increased by working for an organization that takes employee development seriously" (Institute of Leadership & Management, 2016).

9.2. Monitoring and Evaluation

Assessment of AI tool implementation progress in Site Reliability Engineering requires continuous monitoring and evaluation of system-outcome performance. First, selected metrics, Key Performance Indicators (KPIs), and target values need to be defined, corresponding to the system's optimization aims. Second, the system's current state is compared against the predefined target values, enabling the quantification of cost–benefit ratios. Such analyses facilitate the quantification of SRE service improvements due to AI implementation. The defined metrics may also serve as benchmarks for future implementations of alternative AI-based SRE tools.

To monitor the status and evaluate the impact of AI tool implementations, the development of new evaluation methods and the modification of existing ones are necessary[27-29]. The principal purpose of these methods is to correlate technical system states or process data with the broader business context. As the state of the art in ML-field-level monitoring advances, it is expected that the evaluation of AI tool implementation in SRE will progress accordingly.

10. Conclusion

The research focuses on insight into the use of AI tools in Site Reliability Engineering (SRE). It develops an understanding the scope of SRE jobs and tasks and the influence AI will have on the quest to build highly available, resilient, and fault-tolerant systems. A comparative study on the AI tools offered by a few leading companies has been undertaken.

SRE is of great importance in the complex world of IT management and operations. Artificial Intelligence in Engineering discusses some of the AI technologies and the benefits they bring to engineering. The Selection and Comparison of AI Tools looks at the criteria used to select AI tools and provides descriptions of some of the leading AI tools. Case Studies on Leading Companies shows how the companies have implemented and used these AI tools for the purpose of SRE, illustrated by some real examples. The Impact of Site Reliability on AI discusses how AI has effected site reliability, what it has been able to do in terms of driving down costs and being proactive around incidents. Future Trends in AI for SRE: Discusses the future of AI for SRE in the next 5 – 10 years. Finally, the Challenges in Implementing AI Tools discusses the difficulties in integrating, adopting and using AI tools and the Best Practices for AI Integration discusses a few best practices to integrate AI tools in terms of SRE that are recommended.

References

[1] Lo D. Trustworthy and synergistic artificial intelligence for software engineering: Vision and roadmaps. In2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE) 2023 May 14 (pp. 69-85). IEEE.

[2] Swain P. The Artificial Intelligence and Machine Learning Blueprint: Foundations, Frameworks, and Real-World Applications. Deep Science Publishing; 2025 Aug 6.

[3] Panda SP. Securing 5G Critical Interfaces: A Zero Trust Approach for Next-Generation Network Resilience. In2025 12th International Conference on Information Technology (ICIT) 2025 May 27 (pp. 141-146). IEEE.

[4] Enemosah A. Enhancing DevOps efficiency through AI-driven predictive models for continuous integration and deployment pipelines. International Journal of Research Publication and Reviews. 2025 Jan;6(1):871-87.

[5] Panda SP. The Evolution and Defense Against Social Engineering and Phishing Attacks. International Journal of Science and Research (IJSR). 2025 Jan 1.

[6] Battina DS. DevOps, a new approach to cloud development & testing. International Journal of Emerging Technologies and Innovative Research. 2020 Aug;2349-5162.

[7] Maheshwari A. Digital transformation: Building intelligent enterprises. John Wiley & Sons; 2019 Sep 11.

[8] Subramanya R, Sierla S, Vyatkin V. From DevOps to MLOps: Overview and application to electricity market forecasting. Applied Sciences. 2022 Sep 30;12(19):9851.

[9] Dang Y, Lin Q, Huang P. Aiops: real-world challenges and research innovations. In 2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings. IEEE; 2019 May 25. p. 4-5.

[10] Liu J. Web Intelligence (WI): What makes wisdom web?. In IJCAI. 2003 Aug 9;3:1596-1601.

[11] Maddox TM, Rumsfeld JS, Payne PR. Questions for artificial intelligence in health care. JAMA. 2019 Jan 1;321(1):31-2.
[12] Paschen J, Kietzmann J, Kietzmann TC. Artificial intelligence (AI) and its implications for market knowledge in B2B marketing. Journal of Business & Industrial Marketing. 2019 Oct 7;34(7):1410-9.
[13] Myllynen T, Kamau E, Mustapha SD, Babatunde GO, Collins A. Review of advances in AI-powered monitoring and diagnostics for CI/CD pipelines. International Journal of Multidisciplinary Research and Growth Evaluation. 2024 Jan;5(1):1119-30.
[14] Etzioni O, Cafarella M, Downey D, Popescu AM, Shaked T, Soderland S, Weld DS, Yates A. Unsupervised named-entity extraction from the web: An experimental study. Artificial Intelligence. 2005 Jun 1;165(1):91-134.
[15] Bonner E, Lege R, Frazier E. Large Language Model-Based Artificial Intelligence in the Language Classroom: Practical Ideas for Teaching. Teaching English with Technology. 2023;23(1):23-41.
[16] Rao AS, Vazquez JA. Identification of COVID-19 can be quicker through artificial intelligence framework using a mobile phone–based survey when cities and towns are under quarantine. Infection Control & Hospital Epidemiology. 2020 Jul;41(7):826-30.
[17] Shivadekar S, Halem M, Yeah Y, Vibhute S. Edge AI cosmos blockchain distributed network for precise ablh detection. Multimedia Tools and Applications. 2024 Aug;83(27):69083-109.
[18] De Silva D, Alahakoon D. An artificial intelligence life cycle: From conception to production. Patterns. 2022 Jun 10;3(6).
[19] Kietzmann J, Paschen J, Treen E. Artificial intelligence in advertising: How marketers can leverage artificial intelligence along the consumer journey. Journal of Advertising Research. 2018 Sep 1;58(3):263-7.
[20] Panda SP. Augmented and Virtual Reality in Intelligent Systems. Available at SSRN. 2021 Apr 16
[21] Karamitsos I, Albarhami S, Apostolopoulos C. Applying DevOps practices of continuous automation for machine learning. Information. 2020 Jul 13;11(7):363.
[22] Sterne J. Artificial intelligence for marketing: practical applications. John Wiley & Sons; 2017 Aug 14.
[23] Brusilovsky P, Peylo C. Adaptive and intelligent web-based educational systems. International journal of artificial intelligence in education. 2003 May;13(2-4):159-72.
[24] Buch VH, Ahmed I, Maruthappu M. Artificial intelligence in medicine: current trends and future possibilities. British Journal of General Practice. 2018 Mar;68(668):143.
[25] Ghosh A, Chakraborty D, Law A. Artificial intelligence in Internet of things. CAAI Transactions on Intelligence Technology. 2018 Dec;3(4):208-18.
[26] Raschka S, Mirjalili V. Python machine learning: Machine learning and deep learning with Python, scikit-learn, and TensorFlow 2. Packt Publishing; 2019 Dec 12.
[27] Panda SP. The Evolution and Defense Against Social Engineering and Phishing Attacks. International Journal of Science and Research (IJSR). 2025 Jan 1.
[28] Subramanya R, Sierla S, Vyatkin V. From DevOps to MLOps: Overview and application to electricity market forecasting. Applied Sciences. 2022 Sep 30;12(19):9851.

[29] Dang Y, Lin Q, Huang P. Aiops: real-world challenges and research innovations. In 2019 IEEE/ACM 41st International Conference on Software Engineering: Companion Proceedings. IEEE; 2019 May 25. p. 4-5.

Deep Science Publishing, 2025
https://doi.org/10.70593/978-93-7185-060-5

DeepScience
Open Access Books

Chapter 10: Shaping the Future of Autonomous DevOps and Site Reliability Engineering

1 Introduction to Autonomous DevOps

Autonomous DevOps is the next stage in the automation journey of DevOps, where things are not only automated to reduce toil, but to a stage where it can operate almost without human intervention. In this mature stage, automatic operations are performed on a routine basis, human intervention may not be needed, and AI and machine learning models are central, for providing active support to mission-critical operational decisions [1-3]. By design, Site Reliability Engineering (SRE) bakes in its principles and practices at every step of the way, even in writing the automation itself. The self-healing nature of infrastructure and the rolling nature of autonomous DevOps are visible stripes of the same zebra and facilitated by technology like AI-based monitoring. We are beginning to see early successes from institutions leading the charge in these capabilities. The morality of AI is just as important as the mechanics of automation. Sticking to core AI ethics principles, organizations are able to reverse over-automation decisions before performance hits the skids. Human-centered design for AI increases the efficiency of diverse teams to understand and work together in running complex systems, so long as we have adequate descriptive explainability that accompanies the use of AI models and technologies.

2. The Role of Site Reliability Engineering

Site Reliability Engineering - SRE Site Reliability Engineering (SRE) is a discipline that incorporates aspects of software engineering and applies them to

IT operations problems. It assists in balancing feature development velocity with live product stability. For some companies, including Google, the adoption of SRE is due to the realization that simply automating traditional IT operations work will not cut it. SRE presents a new approach to IT operations, one that has been proven to help Google and other Internet companies make drastic improvements to key processes through a focus on change management as opposed to crisis management.

From these principles, Autonomous DevOps is something that just follows'. A lot of companies are working toward greater automation in the build/deploy/test pipeline, and its inevitable that theyll also want to automate many operational tasks [1-3]. But following in the steps of Autonomous DevOps, AI ethics and explainability is a topic in need of careful consideration. With increasing machinized decisions and actions of AI-backed SRE and DevOps jobs, It is important to control the approval of these decisions to be those that are ethical and attributed.

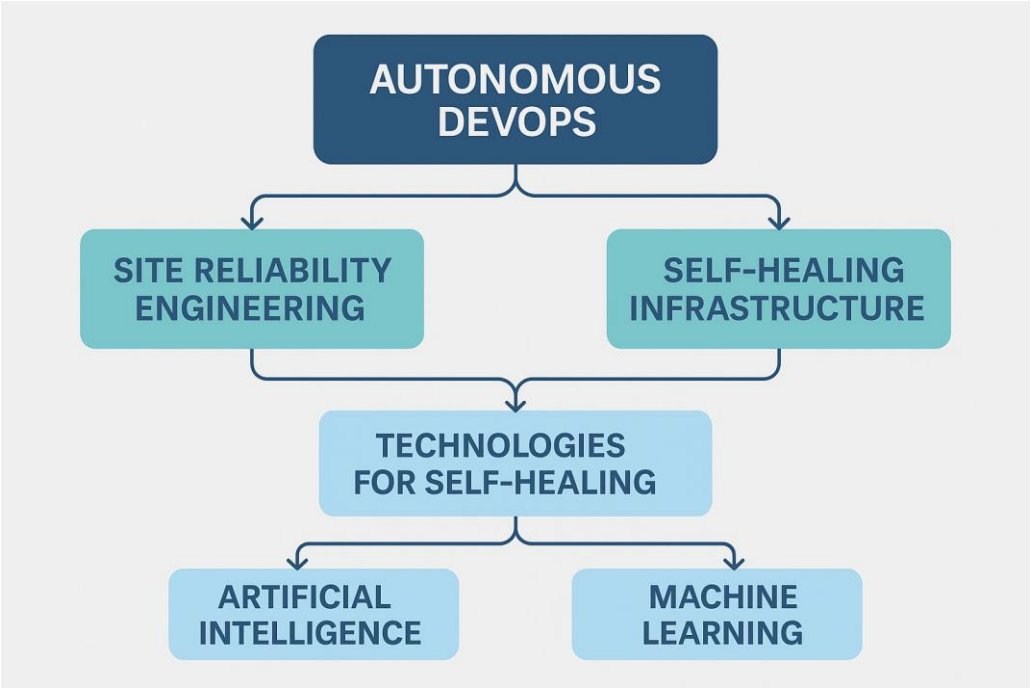

Fig 1. Autonomous DevOps

3. Self-Healing Infrastructure

Self-healing infrastructure refers to computer systems which can automatically recover from disruption to their services without human intervention. This is the power behind self-driving DevOps and site reliability engineering (SRE) for faster and efficient IT and cloud operations. Fully autonomous organizations require layering Web 3.0 technologies (including AI and ML) with advanced orchestration and monitoring.

Several real-world instances demonstrate the application of self-healing infrastructure that will lead operations to a mature level of autonomy in the coming years. Some use cases: Beneath them, Cisco Systems uses AI tools to automate IT processes and tasks. **2. Amazon Web Services (AWS)** incorporates AI into all its functions through a central machine learning center of excellence, extending models and practices that speed automation. HCL Technologies uses AI and ML to automate the development lifecycle and infrastructure management and to enable self-service automation in an evolutionary fashion. All of these practices demonstrate the importance of self-healing systems in the evolution of DevOps and SRE practices.

3.1. Definition and Importance

The phrase autonomous DevOps has been popping up in conversations regarding the future of DevOps and site reliability engineering (SRE), indicating a greater level of automation [2]. The concept of a self-healing platform—a system that can repair itself without human intervention to ensure its own availability—has long been seen as a "holy grail." In this scenario, autonomous DevOps is supported by self-healing infrastructure, enabling both DevOps and SRE to deliver on their responsibility to maintain the availability of applications and services.

Autonomous DevOps is quite a forward-facing term but the origins of SRE date back over ten years. In spite of that, self-healing still remains as a critically important issue. Self-healing infrastructure/In order to engage with the ethical and explainability aspects of AI, Autonomous DevOps lives in the meeting point between these. Taken together, these views help to guide the journey from automated assistance for DevOps and SRE to a completely autonomous system.

3.2. Technologies Enabling Self-Healing

Self-healing is a indispensable feature in the autonomous DevOps and SRE, and we analyzed the supporting techniques for this feature. As automation is integral for any self-healing system, tools in this area also have to be present in a survey. The technologies facilitating autonomous DevOps and SRE therefore include not

only self-healing enabling technologies, but also separable yet important derivatives of artificial intelligence and machine learning. The inherent automation of autonomous DevOps and SRE creates a natural affinity with principles of artificial intelligence and machine learning, raising questions of AI ethics and the social-cultural aspects of the practice. Among these areas, three particularly intriguing dimensions emerge: self-healing, AI ethics, and explainability [2,4,5].

AI ethics and explainability occupy separate sections, but the notion of self-healing infrastructure and operations represents a core idea that transcends a single aspect of automation. Self-healing technology hinges on integrative monitoring, analytic, planning, and execution capabilities. Fully autonomous infrastructure becomes capable not only of advanced self-diagnosis but also of conceiving and implementing remedial measures without human intervention. Numerous commercial projects explore self-healing concepts by orchestrating diverse analytic tools, enabling automatic remediation—such as self-eviction of faulty containers in Kubernetes or initiating autoscaling actions in red-queen scenarios—and automatically shutting down business services impacted by detected attacks.

3.3. Case Studies of Self-Healing Systems

Research and industry projects demonstrate that the concept of self-healing infrastructure is neither new nor futuristic. Google's Site Reliability Engineering (SRE) team applies Error Budget Policies, a real-world example of self-healing infrastructure that enables a self-regulating environment to simultaneously deliver innovation, features, and system reliability. Another case in point is Uber's Michelangelo, an internal machine-learning platform that lays the groundwork for self-healing infrastructure. It enables automatic detection of model quality degradation and dynamic retraining and deployment of models to higher environments. Qualys transforms private cloud environments with "autonomous vulnerability management"; this system identifies and implements the best solutions to maintain an entire private cloud environment free of vulnerabilities.

These cases express the real business value of self-healing infrastructure: autonomy, self-regulation, and a high degree of automation. Taken to the highest extremes, this levy can still fall back on humans. The self-healing vision, accompanied by its own set of design principles and methodologies, would then require humans to operate at strategic and supervisory levels rather than at operational and tactical levels [6-8]. Several recent technologies reflect this trend. Emerging real-time packet analytics technology focuses on live packet

processing to identify outlying information and offers solutions to neutralize or clean the packet in real-time without human intervention. Google's Site Reliability Engineering (SRE) book, The Site Reliability Workbook, presents an entire chapter on automation to enhance reliability and even deliver self-healing infrastructure.

4. AI Ethics in Operations

Artificial Intelligence (AI) ethics are concerned with the moral behavior of AI itself and with the moral behavior of people creating, deploying, and operating AI. AI ethics defines how and when AI should be applied, so that it helps and empowers people, without harming them. For AI governance to be effective, it needs to be implemented at every management level, including at the very beginning, when executive management decides if and where AI-based automation should be applied in the first place. Individuals who create and operate AI solutions have to ensure they do so with respect of AI ethics principles. Ethics are also especially important when autonomous DevOps or Site Reliability Engineering (SRE) are considered: when automation is capable of taking control over production systems, a clear set of AI ethics principles must be defined.

Explainability goes together with ethics, helping AI operators to understand the decision process of an AI model, providing them with extra knowledge and safeguards [9,10]. Explainability therefore makes the model's behavior more explicit and justifiable. This in turn makes decisions more transparent and accountable, assists compliance, and helps build trust.

4.1. Understanding AI Ethics

Artificial intelligence can empower Autonomous DevOps and site reliability engineering (SRE) to tackle significantly more demanding tasks, realize more ambitious goals, and do so in increasingly complex and dynamic environments. However, this is sustainable only if AI algorithms and models are designed, developed, and deployed in accordance with agreed-upon ethical principles. A fundamental set of AI ethics principles is thus among the three emerging foundations of Autonomous DevOps and SRE. Just as the concept of introducing progressively higher levels of automation in the self-driving car has led to an expanded understanding of both self-driving cars and automation, so too has the range and growing societal implications of AI's deployment led to more general principles of AI ethics. In the context of Autonomous DevOps and SRE, this

demands that AI technology be applied as the means to an end, rather than the end itself [11-13].

The recent implementation of a self-driving car capable of detouring by a smart and autonomous re-routing of the vehicle showcases the importance of explainability in the use of AI. Birhane et al. have further stressed the imperative that computer scientists, AI researchers, and practitioners act ethically in the application of AI. The author of that implementation described, discussed, and analyzed the self-driving car's self-healing capability in relation to self-healing infrastructure. A coherent body of work is beginning to emerge.

4.2. Implications for DevOps and SRE

Automation is an invaluable asset in the continuous progress of DevOps and SRE, but misuse can lead to negative outcomes. Without a measure of indirect personality aligning each job, automation can provoke distress and unhappiness. It follows that developers who take over bank branches may be unhappy if their indirect personality measures MSc are not considered in an automated reassignment process. Parallel can be drawn from SRE developers taking over operational functions similarly.

All autonomous jobs should therefore be assigned in accordance with MSc. Doing so balances the workload across all logical and physical servers. This balance is as desirable for developers as it is for the servers and it ensures a more rational use of the collective talent pool. By doing so, operational tasks should become a source of learning about the product, broadening expertise and increasing interest in operational tasks and their associated job functions. These operations activities tend to be dull and repetitive, so introspection is necessary to prevent individuals from being encouraged into roles fulfilling their non-indirect personality measures—writing code in the case of a bank that was previously a branch manager or an SRE developer. Caution is warranted in other operations settings that are repetitive and devoid of challenge. These and other ethical considerations may emerge as AI-assisted automation tools advance. Similarly, the requirement for explainability in AI operations aims to enhance transparency and adjustability to operational decision-making.

4.3. Frameworks for Ethical AI Implementation

The implementation of any AI system must consider its ethical, social, and legal effects. These effects cannot be considered as an afterthought and need to be factored into the design process before capabilities are deployed in production [2,14-17]. A survey of ethical AI frameworks reveals a number of common principles covering aspects such as transparency, justice and fairness, non-

maleficence, responsibility, privacy, beneficence, freedom and autonomy, trust, dignity, sustainability, and solidarity.

Ensuring that general AI principles are part of autonomous DevOps foundations helps with responsible automation. These principles influence the design of automation capabilities, the rules that govern operational decision-making, and the wider cultural impacts of automating more and more aspects of the systems.

5. Explainability in AI Operations

Explainability is a chief concern in the pursuit of autonomous DevOps and SRE. It establishes the trustworthiness of artificial intelligence (AI) and machine learning (ML) models, products, and applications. By explaining individual or global decisions, the models bestow various benefits upon multiple stakeholders and facilitate the identification and mitigation of model bias that may impact other stakeholders.

AI-enabled DevOps and SRE products control software deployment and operation, workload management, infrastructure tiering, and proactive self-healing of IT systems. The ever-growing impact of self-healing infrastructure on business and individual users further increases the demand for explainability in these products. Several frameworks guide the adoption of just, ethical, and trustworthy AI. Queries such as “Why did the software error and failure prediction model predict that?” or “How did the model arrive at that particular prediction?” have crucial implications. Questions of this nature determine whether a self-healing operation should be executed, who may be legally responsible for mistakes or errors made by an AI model, and the level of autonomy that the model should be permitted to exercise.

5.1. The Need for Explainability

Even for the human experts who developed the models powering Autonomous DevOps (ADO) and Site Reliability Engineering (SRE) operations, the inner workings of these models are often a black box. Decisions can only be questioned if their underlying process and rationale are transparent and understandable. Unless decisions can be interpreted and understood, the capacity to question, argue against, or hold operators responsible is missing. Explainability ensures that actions taken by AI systems are comprehensible, thus fostering trust and facilitating accountability [9,18-21].

Key stakeholders—customers, end users, and regulators—are increasingly demanding explanations and justifications for decisions and actions taken by AI systems. Providing insight into the models' operations is a complex undertaking, yet one that is rapidly becoming unavoidable. Both ethical mandates and practical necessities will drive explainability to the forefront of AI applications in DevOps and SRE, requiring technologies and processes that demystify automated decision-making.

5.2. Techniques for Enhancing Explainability

Research on explainability (sometimes also called interpretable machine learning or machine learning + X) focuses on understanding machine learning models and their behavior. A concise summary of techniques for enhancing explainability follows. These techniques enable machine learning models and other AI systems to provide explanations for their outputs, correct their behavior, or enable corrections by an external agent (e.g., an SRE).

Explainability methods often operate in conjunction with machine learning models that have been trained on labeled data.

Explanation of model mistakes using counterfactuals . Exception-based explanations describe a data point in relation to exceptions or outliers for a specified property—either a matched set of points possessing or lacking the property. Counterfactuals are a subset of exception-based explanations, where the property of interest is typically the predicted label. For example, the prosthetic leg might not be deployed because the delivery date is incorrectly set to a date in the past.

Feature feedback labeling . If a model is clearly incorrect, the explanation could be used to identify whether the model has assigned importance to erroneous features. Such a finding would enable the SRE to retrain the model with a special label indicating that those features are irrelevant.

Explanation of model parameters . Rule-based models have an inherent explanation; similarly, parameters of statistical models have inherent meanings that establish explanations.

Explanation of model behavior . Techniques that focus on visualizing (e.g., heatmaps) or summarizing the behavior of any machine learning model help provide insights into the model's decisions.

Diagnostics for model improvement . Metrics and diagnostics are additional explainability methods that can indicate that the model is performing well in

general but might require some improvements (e.g., detection of bias or poor performance on certain feature subsets).

5.3. Challenges in Achieving Explainability

Explainability is a fundamental prerequisite in many cases for the responsible automation of DevOps and SRE operations; however, achieving it can be challenging. First, the trained-for automation needs to be encapsulated in explainable outputs. Generating a valid, actionable incident countermeasure is not sufficient in itself, because any responsible automation framework has to differentiate between the low-risk changes that can be executed autonomously and the high-risk case where human intervention must be triggered. For the latter, an explainable rationale must be provided to decision-makers, implying that automation cannot be based on some inscrutable inner workings of a black-box system. Second, from the inner workings perspective, the requirement of data explainability also cuts in a different direction [22,23]. Although explainable methods (such as decision trees, random forests, and explainable variations of deep neural networks) exist, they are typically less powerful than their purely predictive black-box equivalents. The challenge of balancing explainability with predictive power embodies the manifesto balance of ModelOps, which states that the ‘‘integration and automation of models across the entire AI production lifecycle must find a tradeoff between business relevance, risk, and cost.’»

Addressing explainability requires comprehensive training of all members of the DevOps and SRE teams, ensuring that they understand and can thoroughly question the outputs of explainable automation. It is essential that teams are not pressured into blind reliance on automation simply because of the sheer scale of information involved in operating the modern data-driven IT environment. Beyond internal training, it is important to use an appropriate automation tool that produces explainable outputs [24-26]. The related fields of Explainable AI and Explainable Machine Learning promise to provide automation tools that can assist the DevOps and SRE team, augmenting their capabilities to maintain and operate excellent service quality despite the increasing cloud-native infrastructure complexity.

6. Automation Tools and Technologies

Numerous automated tools already support standard DevOps and SRE operational functions. Some of the more advanced tools designed to move organizations toward Autonomous DevOps include operations AI platforms, AI-

based monitoring tools, and self-healing tools. These tools will be enhanced rapidly over the next few years.

A comprehensive set of training courses is also necessary to prepare developers and operators for a future with Autonomous DevOps and SRE. The focus of these courses includes the impact of AI Ethics and Explainability on automation process, procedures, and tools. In addition to discussing the course material presented in other sections, the training material also covers how Autonomous DevOps will affect jobs.

6.1. Overview of Current Tools

Various automation tools facilitate autonomous DevOps and Site Reliability Engineering (SRE). In the area of monitoring, examples include pre-built dashboards (SolarWinds Orion NPM), customizable self-hosted dashboards (Grafana), event management (Moogsoft), bidirectional integrations with established ITSM systems (PagerDuty), and CloudOps options such as AWS CloudTrail and CloudWatch. For provisioning, tools available are Terraform, the Serverless Framework, and AWS CloudFormation [27,28].

As for the orchestration of self-healing across any one of Terraform, Serverless Framework, or AWS CloudFormation, customization typically occurs via Python, PowerShell, or Node.js; in the world of AWS, these serverless applications are characterized as Lambda functions. More generally, serverless is a cloud model proposed to inherently achieve higher levels of self-healing. Solutions such as Ansible Tower already incorporate basic remediations tied to event management alerts.

6.2. Future Trends in Automation

Explosive growth of automation capabilities will continue to accelerate changes in how DevOps and SRE teams work. While the self-healing infrastructure use case remains critically important, two additional aspects will have an enormous impact of how people collaborate [19,29-31]. The first aspect is responsible use of automation, which calls for incorporating ethics into the algorithms that drive self-healing. The second is explainability, which enables DevOps and SREs to better comprehend moves suggested or executed by any self-healing mechanism.

Automation will never be able to completely replace human operators; therefore, it remains important to foster the existing culture of collaboration between DevOps and SREs. When teams achieve this goal, automation elevates human roles, enabling people to work more efficiently with fewer errors. Organizations may find that the need for upskilling is less about acquiring hard technical skills

and more about adjusting cultural mindsets. In particular, training on responsible use of automation and awareness of its limitations should be key considerations.

7. Cultural Shifts in DevOps and SRE

Creating an autonomous, explainable, and ethical company culture is the most challenging factor for dominion in a variety of industries. Changing the ways in which DevOps and SRE professionals have been working for years is no small Read more. This cultural transformation is significantly shaped by the way in which automated systems are used [32,33]. One could think of the accumulated DevOps team combined with SageMaker model that sleeps platform patches in automated way with less a bit of the human effort and allowing practitiAioners to focus instead of fixing the warning roof peak to build even more value towards the enterprise. This culture thrives on training and education. If the ethical and explainability components of these autonomous activities can be understood by practitioners, AI and ML operations at work are less of a black box, leading to trust in AI.

Without this focus on ethics and explainability, the company risks using automation tools that do not act for the benefit of the business and its customers. The same ethos explaining why the practice of Site Reliability Engineering should be incorporated into an autonomous world also clarifies why a greater measure of ethical consciousness should be imbued whenever artificial intelligence is plugged into operations executions. A robust ethical principle framework for artificial intelligence was established by the European Commission in 2021 and serves as a foundational guide for this aspect of autonomous DevOps and SRE.

7.1. The Importance of a Collaborative Culture

Although the ultimate goal of autonomous DevOps is to eliminate repetitive toil operations, humans will not be taken out of the equation altogether. Thus, the emerging ethical considerations demand training and nurturing the human operators and support staff to be able to understand that AI decisions can be flawed and must be re-examined. This underlines the importance of cultivating a DevOps culture that encourages collaboration, with practitioners supporting each other in conducting quality control decisions [34-36].

Similarly, AI-based operations must be explainable to humans and actors in the software supply chain, as well as to auditors and compliance officers. Training

DevOps practitioners to provide explanations for the decisions made by AI is as crucial as preparing their human colleagues to request clarifications.

7.2. Training and Development for Teams

Despite or because of all the planned, scripted and automated DevOps functions that reduce human operational stress and human error, ever more training is needed.10 A further implication of AI ethics and explainability in autonomous DevOps and SRE is cultural: no DevOps or SRE team should be allowed to use these advanced autonomous capabilities without first having had training on their responsible use. For example, as explained in Section 7.4, templates and workflows should be established before undertaking any AI operations that use closed-loop automated additions to alert systems or closed-loop remediation. Such training ensures that the output of AI engines is visible to the organization and identified as AI-generated when embedded in alerts, recommendations and code.

8. Impact of Autonomous Systems on Job Roles

Fully autonomous DevOps and SRE do not imply complete removal of humans from DevOps and SRE job roles. Nevertheless, autonomous DevOps reduces mundane DevOps and SRE tasks. DevOps teams are rapidly adopting tools, frameworks, AI engines, and automation scripts to support Site Reliability Engineering functions to deploy the infrastructure, deploy the application, provide service discovery, perform log analysis, identify downtime by monitoring, escalate bugs, perform Auto-remediation, patch, collect postmortem report details, and more. This approach to Devops also goes by the name Auto DevOps or Autonomous DevOps. It must also be understood that DevOps and SRE are cultural methods of working. Simply introducing autonomous automation for operations is not enough. Eventually support personnel would shift their training and skill development to deploy these tools and perform audits on their use, and failover and risk mitigation plan development.

Additionally, by leveraging AI to take over tasks and decision-making, the scope and risks of errors and failure modes can dramatically increase [37-40]. The lack of attention by DevOps and SRE personnel to develop an understanding about AI Ethics can lead to a very flawed and erroneous Full AI approach for the team. It limits the adoption of these tools and automations for decision-support or decision making. Lack of awareness about the need for AI Explainability for Transparent AI also results in the operator’s discomfort to use these automation

capabilities. It reduces confidence in using the tool for decision-making because any wrong decision made by the automation can cast a negative impact on the career or the reputation of the employee.

8.1. Evolving Job Descriptions

A natural follow-on topic from the rise of autonomous DevOps (ADO) and Site Reliability Engineering (SRE) is how the underlying job descriptions are likely to evolve. Autonomous DevOps and SRE will shift job responsibilities from executing change and incident processes and fulfilling on-call items to supporting change and incident processes by providing the automation tooling that executes the bulk of that work. Similarly, firefighting will no longer be the primary task for SRE but will instead focus on developing the tooling that performs the firefighting.

Expanding on these shifts uncovers the need for emphasis in two additional areas: AI ethics and explainability. The ethical use of AI hinges on the development of tools that perform tasks in an explainable manner. Organizations that encourage an ethical use of AI within their operational teams should proactively embed responsible AI and XAI (explainable AI) training into their programs to facilitate the use of autonomous SRE tooling. Conversely, organizations that overlook these requirements may face difficulties in producing the necessary tooling or experience resistance to its adoption, thereby missing out on the benefits of ADO and SRE Automation.

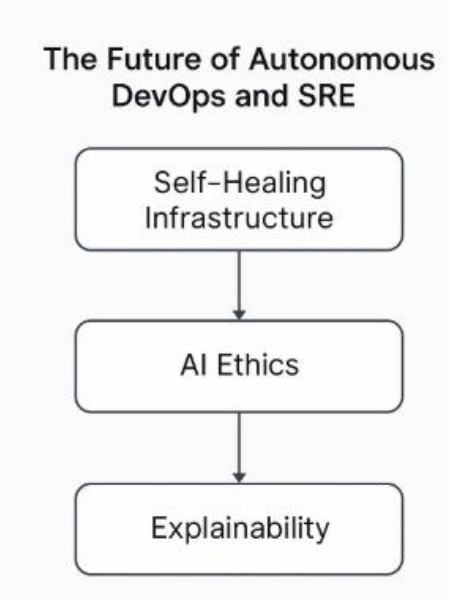

Fig 2. Autonomous DevOps (ADO) and Site Reliability Engineering (SRE)

8.2. Upskilling and Reskilling Needs

As DevOps and Site Reliability Engineering (SRE) teams change over to managing self-driving systems that keep the applications and IT infrastructure in a good health, what they do is also shifting. Even as complex and repetitive tasks begin to bend to the will of automation, the weight of responsibility is growing higher, especially in risk management and AI ethics. Teams need to be ready to address these issues, which requires the changes in organizational culture, specialized training, the adoption of responsible DevOps and SRE automation tools, and commitment to governance frameworks. Without those, and automation's potential will be in vain, with big ethical risks coming at us full throttle.

DevOps and SRE staff need to be trained in ethical considerations and associated end user operations, along with knowing what responsible automation tools are capable of [41-43]. Companies rolling out self-contained DevOps and SRE systems should apply AI ethics principles—either tap into their company-wide AI guidelines or opt for those laid out by ethics consortia. Furthermore, a thorough understanding of extensible explainability in the context of Autonomous DevOps and SRE augments the teams' ability to decode AI reasoning, thereby supporting the operationalization of AI ethics principles. Such comprehensive training fosters a culture of responsibility and ensures that deployed tools effectively safeguard the infrastructure.

9. The Future Landscape of DevOps and SRE

SRE and DevOps will evolve into autonomous SRE and autonomous DevOps by leveraging increasingly autonomous technologies like artificial intelligence (AI). Many organizations get stuck at partial automation, and the existence of multiple legacy applications, logs, and alerts keeps them mired in the previous era. The future of DevOps and SRE envisions a more comprehensive automation of processes to bring greater simplicity, agility, and efficiency to operations teams.

The availability of increasingly advanced tools is enabling organizations to develop and maintain self-healing infrastructure. SRE and DevOps teams bear responsibility for maintaining automation and managing the impact of self-healing on the cultural aspects of continual collaboration, shared responsibilities, and upskilling. However, the adoption of AI introduces concerns about ethics, bias, and explainability that warrant attention. It is so important to do the responsible thing, not only for technical reasons but also because tooling is being

developed at a breakneck speed. There is also a need for cultural change to confront the loss of jobs and the withering of human stewardship.

9.1. Predictions for the Next Decade

Introduction to Autonomous DevOps and the Future of SRE Autonomous DevOps is the (disciplined) removal of manual work from IT operations. One thing that serves as the cornerstone is a tendency toward the eventual, complete automation of all things with DevOps and SRE. This transformation is expected to lead to increased operational efficiency, lowering of human error and the prevention of unplanned downtime and security risks. Process efficiency improvement also drives culture development by increasing interactivity, collaboration, shared ownership, instrumented experience and know-how. The impact of automation on corporate infrastructure and skills An Autonomous DevOps journey cannot be undertaken without acknowledging the wider effects that automation is having (and will continue to have) on corporate resources and skills.

Forward Looking Perspectives and Predictions Despite perhaps sounding too specific, the notion of autonomous DevOps actually spans a broad range of viewpoints and proffered ideas about the future. The artificial intelligence topic is broad; it spans from which technologies will get automation sup port, to what is the cultural impact for DevOps and SRE teams, going through what is the dream of self-healing infrastructures, to the exposure to risk of running AI Technology. It also questions the ethical and the practical implications of IT's future directions. From these varied views, they coalesce to provide predictions for the revolution that is slated to occur over the next decade and more with autonomous DevOps for DevOps and SREs operations management [44].

9.2. Potential Challenges Ahead

The autonomous DevOps and SRE systems automating common decisions in the context of largescale IT operations are a big leap toward greater operational efficiency. Support for operations workers is provided with specific automation tooling, and scaling of autonomous systems allows for a broader range and more complex automation. But The ability to create and act on operational decisions also raises questions about AI Ethics and Explainability. These Responsible AI concepts cover ethical approval, scrutiny and understanding of algorithms. In the IT operations area, they try to figure out how much to cede decision-making to

an algorithm, under what conditions, and what will be the comprehension over the foundation for such decisions.

These conversations are things that have to be nurtured in DevOps and SRE cultures, and that's hard. While there remains a natural symbiosis between seasoned operations workers and well-tuned AI systems, the increasing complexity of automation technologies is also impacts cultural perspectives. Collaboration is a key principle of both DevOps and SRE, but what it looks like in practice is changing. Automation becomes more about building and managing tools than about being responsible for daily IT operations. It therefore also becomes necessary to train for responsible AI ethics and explainability, including the limits to which autonomous systems can be trusted.

10. Conclusion

Finally, an ending view on the future of autonomous DevOps and SRE brings up three critical bits: self-healing infrastructure, AI’s ethics and Explainability: It is this aspect that needs addressing to make the promise of more automation work.

Self-healing infrastructure is a fundamental aspect of autonomous DevOps and SRE. Observability, release orchestration and AI-based automation combined enable systems to predict common problems, pinpoint their root causes, recommend remediation actions and validate that remediation was successful. It is crucial to further these mechanisms beyond a pilot stage to achieve substantial enhancements in efficiency of IT operations. Principles of AI ethics safeguard against the irresponsible and unethical use of AI and ML within IT operations. Addressing underlying questions—such as whether an action is right, who should benefit or be harmed, and the responsibilities toward AI-enabled tools—requires also considering explainability to establish trust in autonomous capabilities. Explainability in AI operations enhance stakeholder understanding of automated actions and recommendations, bolstering confidence in the technology. The future landscape of DevOps and SRE is poised to leverage autonomy, AI, and machine learning comprehensively. Organizations will thus be compelled to cultivate a culture that promotes collaboration, shared responsibility, and continuous learning.

References

[1] Karamitsos I, Albarhami S, Apostolopoulos C. Applying DevOps practices of continuous automation for machine learning. Information. 2020 Jul 13;11(7):363.

[2] Dragičević T, Wheeler P, Blaabjerg F. Artificial intelligence aided automated design for reliability of power electronic systems. IEEE Transactions on Power Electronics. 2018 Dec 20;34(8):7161-71.

[3] Moayedi H, Mosallanezhad M, Rashid AS, Jusoh WA, Muazu MA. A systematic review and meta-analysis of artificial neural network application in geotechnical engineering: theory and applications. Neural Computing and Applications. 2020 Jan;32(2):495-518.

[4] Bello O, Holzmann J, Yaqoob T, Teodoriu C. Application of artificial intelligence methods in drilling system design and operations: a review of the state of the art. Journal of Artificial Intelligence and Soft Computing Research. 2015;5(2):121-39.

[5] Cheng Z, Jia X, Gao P, Wu S, Wang J. A framework for intelligent reliability centered maintenance analysis. Reliability Engineering & System Safety. 2008 Jun 1;93(6):806-14.

[6] Moreno-Guerrero AJ, López-Belmonte J, Marín-Marín JA, Soler-Costa R. Scientific development of educational artificial intelligence in Web of Science. Future Internet. 2020 Jul 24;12(8):124.

[7] Chen L, Chen P, Lin Z. Artificial intelligence in education: A review. IEEE Access. 2020 Apr 17;8:75264-78.

[8] Devedžić V. Web intelligence and artificial intelligence in education. Journal of Educational Technology & Society. 2004 Oct 1;7(4):29-39.

[9] Alam A, Alam S. Evolution of Artificial Intelligence in Revolutionising Web-Based and Online Intelligent Educational Systems. SGS-Engineering & Sciences. 2021 Sep 15;1(01).

[10] Maassen O, Fritsch S, Palm J, Deffge S, Kunze J, Marx G, Riedel M, Schuppert A, Bickenbach J. Future medical artificial intelligence application requirements and expectations of physicians in German university hospitals: web-based survey. Journal of Medical Internet Research. 2021 Mar 5;23(3):e26646.

[11] Calvo-Rubio LM, Ufarte-Ruiz MJ. Artificial intelligence and journalism: Systematic review of scientific production in Web of Science and Scopus (2008-2019).

[12] Gandon F. Distributed Artificial Intelligence and Knowledge Management: ontologies and multi-agent systems for a corporate semantic web (Doctoral dissertation, Université Nice Sophia Antipolis).

[13] Kietzmann J, Paschen J, Treen E. Artificial intelligence in advertising: How marketers can leverage artificial intelligence along the consumer journey. Journal of Advertising Research. 2018 Sep 1;58(3):263-7.

[14] Almeida F, Simões J, Lopes S. Exploring the benefits of combining DevOps and agile. Future Internet. 2022 Feb 19;14(2):63.

[15] Hannousse A, Yahiouche S. Towards benchmark datasets for machine learning based website phishing detection: An experimental study. Engineering Applications of Artificial Intelligence. 2021 Sep 1;104:104347.

[16] Divya S, Indumathi V, Ishwarya S, Priyasankari M, Devi SK. A self-diagnosis medical chatbot using artificial intelligence. Journal of Web Development and Web Designing. 2018 Apr 7;3(1):1-7.

[17] Bonner E, Lege R, Frazier E. Large Language Model-Based Artificial Intelligence in the Language Classroom: Practical Ideas for Teaching. Teaching English with Technology. 2023;23(1):23-41.
[18] Rao AS, Vazquez JA. Identification of COVID-19 can be quicker through artificial intelligence framework using a mobile phone–based survey when cities and towns are under quarantine. Infection Control & Hospital Epidemiology. 2020 Jul;41(7):826-30.
[19] Panda SP. Augmented and Virtual Reality in Intelligent Systems. Available at SSRN. 2021 Apr 16.
[20] Shivadekar S, Halem M, Yeah Y, Vibhute S. Edge AI cosmos blockchain distributed network for precise ablh detection. Multimedia Tools and Applications. 2024 Aug;83(27):69083-109.
[21] De Silva D, Alahakoon D. An artificial intelligence life cycle: From conception to production. Patterns. 2022 Jun 10;3(6).
[22] Battina DS. DevOps, a new approach to cloud development & testing. International Journal of Emerging Technologies and Innovative Research. 2020 Aug;2349-5162.
[23] Maheshwari A. Digital transformation: Building intelligent enterprises. John Wiley & Sons; 2019 Sep 11.
[24] Subramanya R, Sierla S, Vyatkin V. From DevOps to MLOps: Overview and application to electricity market forecasting. Applied Sciences. 2022 Sep 30;12(19):9851.
[25] Swain P. The Artificial Intelligence and Machine Learning Blueprint: Foundations, Frameworks, and Real-World Applications. Deep Science Publishing; 2025 Aug 6.
[26] Liu J. Web Intelligence (WI): What makes wisdom web?. In IJCAI. 2003 Aug 9;3:1596-1601.
[27] Maddox TM, Rumsfeld JS, Payne PR. Questions for artificial intelligence in health care. JAMA. 2019 Jan 1;321(1):31-2.
[28] Paschen J, Kietzmann J, Kietzmann TC. Artificial intelligence (AI) and its implications for market knowledge in B2B marketing. Journal of Business & Industrial Marketing. 2019 Oct 7;34(7):1410-9.
[29] Sterne J. Artificial intelligence for marketing: practical applications. John Wiley & Sons; 2017 Aug 14.
[30] Burley SK, Bhikadiya C, Bi C, Bittrich S, Chao H, Chen L, Craig PA, Crichlow GV, Dalenberg K, Duarte JM, Dutta S. RCSB Protein Data Bank (RCSB.org): delivery of experimentally-determined PDB structures alongside one million computed structure models of proteins from artificial intelligence/machine learning. Nucleic Acids Research. 2023 Jan 6;51(D1):D488-508.
[31] Brusilovsky P, Peylo C. Adaptive and intelligent web-based educational systems. International Journal of Artificial Intelligence in Education. 2003 May;13(2-4):159-72.
[32] Battina DS. Ai-augmented automation for devops, a model-based framework for continuous development in cyber-physical systems. International Journal of Creative Research Thoughts. 2016 Sep 3;2320-882.
[33] Enemosah A. Enhancing DevOps efficiency through AI-driven predictive models for continuous integration and deployment pipelines. International Journal of Research Publication and Reviews. 2025 Jan;6(1):871-87.

[34] Kim G, Humble J, Debois P, Willis J, Forsgren N. The DevOps handbook: How to create world-class agility, reliability, & security in technology organizations. IT Revolution; 2021 Nov 30.
[35] Panda SP. Securing 5G Critical Interfaces: A Zero Trust Approach for Next-Generation Network Resilience. In2025 12th International Conference on Information Technology (ICIT) 2025 May 27 (pp. 141-146). IEEE.
[36] Etzioni O, Cafarella M, Downey D, Popescu AM, Shaked T, Soderland S, Weld DS, Yates A. Unsupervised named-entity extraction from the web: An experimental study. Artificial Intelligence. 2005 Jun 1;165(1):91-134.
[37] Buch VH, Ahmed I, Maruthappu M. Artificial intelligence in medicine: current trends and future possibilities. British Journal of General Practice. 2018 Mar;68(668):143.
[38] Ghosh A, Chakraborty D, Law A. Artificial intelligence in Internet of things. CAAI Transactions on Intelligence Technology. 2018 Dec;3(4):208-18.
[39] Raschka S, Mirjalili V. Python machine learning: Machine learning and deep learning with Python, scikit-learn, and TensorFlow 2. Packt Publishing; 2019 Dec 12.
[40] Panda SP. The Evolution and Defense Against Social Engineering and Phishing Attacks. International Journal of Science and Research (IJSR). 2025 Jan 1.
[41] Lo D. Trustworthy and synergistic artificial intelligence for software engineering: Vision and roadmaps. In 2023 IEEE/ACM International Conference on Software Engineering: Future of Software Engineering (ICSE-FoSE). IEEE; 2023 May 14. p. 69-85.
[42] Celestin M, Vanitha N. AI vs Accountants: Will Artificial Intelligence Replace Human Number Crunchers. In Indo American Multidisciplinary Web Conference on Arts, Science, Engineering and Technology (IAMWCASET-2020). 2020. p. 117-124.
[43] Iliashenko O, Bikkulova Z, Dubgorn A. Opportunities and challenges of artificial intelligence in healthcare. In E3S Web of Conferences. Vol. 110. EDP Sciences; 2019. p. 02028.
[44] Lutz C. Digital inequalities in the age of artificial intelligence and big data. Human Behavior and Emerging Technologies. 2019 Apr;1(2):141-8.