YRKESHÖGSKOLAN
NOVIA

Exploring AI-driven Use Cases in Integration Platform Red Hat

Mari Lintula

Degree Thesis for a Master of Engineering (UAS) - degree

Degree Program: Industrial Management and Engineering

Place and year: Vaasa 2024

DEGREE THESIS

Author: Mari Lintula

Engineering, M.Eng., Vaasa Supervisor: Ray Pörn, Novia UAS, Vaasa, Arto Kvist, DigiFinland Oy

Title: **Exploring AI-driven use cases in integration platform Red Hat**

Date: 27.5.2024 Number of pages: 57 Appendices: 0

Abstract

In today's rapidly evolving digital landscape, integrating artificial intelligence (AI) technologies into platform development is crucial for enhancing efficiency and fostering innovation.

This research delves into AI-driven use cases within the Red Hat integration platforms development tools and identifies practical applications that optimise development processes and business models. It provides examples such as code generation and support by ChatGPT and introduces AI agents for intelligent error handling, monitoring, and coordination in the development process.

Utilising qualitative research and case study methodology, the study explores historical aspects, technological background, and real-world scenarios to offer actionable insights and recommendations. By examining AI's impact on programming processes and business models, the research highlights the potential of AI technologies to streamline workflows, optimise resource allocation, and expedite project timelines within integration platforms.

The findings underscore the significance of strategic planning, collaboration, and iterative data analysis in successfully integrating AI into development processes. Overall, the research aims to provide valuable insights for professionals in the field and demonstrate the benefits of AI-driven solutions in revolutionising integration platforms such as Red Hat OpenShift.

Language: English
Key Words: Artificial intelligence, machine learning, integration platform, development tools, development frameworks

Table of Contents

TABLE OF FIGURES

ABBREVIATIONS

Abbreviation	Explanation
AGI	Artificial general intelligence
AI	Artificial Intelligence
API	Application Programming Interface
Applitools	AI-driven test tool
Ardublockly	Low code programming tool
AutoML	Automation of machine learning
Bag-of-Words	Representation model
BERT	Bidirectional Encoder Representations from Transformers
ChatGPT	Chat Generative Pre-trained Transformer
Copilot	Microsoft's AI assistant
CSV	Comma-Separated Values
DevOps	Philosophies, practices, and tools for SW development
DL	Deep Learning
Docker	Open-source platform for applications using containers
DSL	Domain-Specific Language
EAI	Enterprise Application Integration
Ernest	Low code programming tool
FTP	File Transfer Protocol
GDPR	General Data Protection Regulation
GenAI	General Artificial Intelligence
GitHub	SW development platform
GPT-4	Natural Language Model from Open AI
GRAFANA	Graphite and Grafana
Groovy	Readable programming language
HTTP	HyperText Transfer Protocol
IoE	Internet of Everything
iPaaS	Integration Platform as a Service
Java	Object-oriented programming language
JMX	Java Management Extensions
JSON	JavaScript Object Notation
Kotlin	Readable programming language
Kubernetes	Open-source system for automating tasks on containerised applications
LLM	Large Language Model
M.Eng.	Master of Engineering
ML	Machine Learning
MLOps	Machine Learning Operations
NLM	Natural Language Model
NLP	Natural Language Processing
NN	Neural Networks
Node-RED	Low code programming tool
NPL	Natural Language Processing

OOP	Object-Oriented Programming
OSS	Open-Source Software
POC	Proof of concept/proof of principle
Prometheus	Monitoring agent
Python	Programming language
PyTorch	Open-source machine learning framework
Q-learning	A machine learning algorithm
Red Hat	Enterprise open-source solution
REST	Representational State Transfer
REST	Representational State Transfer
RNN	Recurrent Neural Networks
ROI	Return on investment
Scratch	Low code programming tool
SOAP	Simple Object Access Protocol
Spring	Application framework for Java platform
SW	Software
TensorFlow	Open-Source Platform
Testim	An AI-driven test tool
TF-IDF	Term Frequency-Inverse Document Frequency
UAS	University of Applied Sciences
URI	Uniform Resource Identifier
XML	Extensible Markup Language
YAML	Human-friendly data serialisation language

1 Introduction

In today's rapidly advancing technological landscape, Artificial Intelligence (AI) integration has become a transformative power across multiple industries. As businesses strive to stay competitive in an increasingly digital landscape, adopting AI technologies has become imperative for driving innovation, enhancing efficiency, and unlocking new growth opportunities.

Industry 4.0 is an extension of the Third Industrial Revolution, also known as the digital revolution, which lasted from the 1950s to the early 2000s. This new era surpasses previous technological advancements with four game-changing disruptive technologies. (IBM, 2024)

Also, in 2016, Gartner explored the profound impact of AI on businesses, as Laurence Goasduff conveyed a powerful message. Industrie 4.0 aims to enhance industrial competitiveness through digital value chains. It requires a transformation emphasising strategic goals, digitalising core competencies, and change management. (Goasduff, 2016)

Today, in 2024, we witness the realisation of these predictions as businesses undergo strategic transformations to embrace the opportunities presented by AI in Industry 4.0.

This thesis examines the potential benefits of implementing AI technology beyond technological advancements. The study investigates the integration of AI technology for DigiFinland Oy and its potential impact on business models. The research aims to identify 1-2 applicable use cases within a specific field and propose relevant enhancements to the business plan. These insights align with those provided by well-known AI-based tools such as OpenAI ChatGPT and AI Agents.

1.1 The client

The client for the master's thesis is DigiFinland Oy, a development company partly owned by the government and recently founded welfare areas. The government owns 33.4% of the company, and the other owners are the welfare areas, Helsinki, and the Joint Authority of the Helsinki and Uusimaa Hospital District. The company has an "in-house" status with regard to the organisations mentioned above.

DigiFinland Oy develops national digital solutions that improve the impact of social welfare and healthcare. One of the company's products, known as 116117, is a healthcare information and guidance service that includes 97% of the Finnish population as of 2021. The company also provides digital solutions for rescue services, planning, and built environment planning (Antila Mirva, 2023).

1.2 Background and problem statement

DigiFinland Oy is confronted with the ongoing challenge of optimising its software development processes to achieve cost savings while maintaining efficiency and innovation.

After evaluating multiple technological solutions, the client has decided to adopt the open-source technology Red Hat, an open-source platform product. After thoroughly analysing and considering various factors, this technology was deemed the most suitable option for the client's needs. With a focus on programming within the Red Hat OpenShift platform, the company seeks to explore the potential benefits of integrating AI technologies to streamline development workflows and reduce programming hours, thus driving development costs.

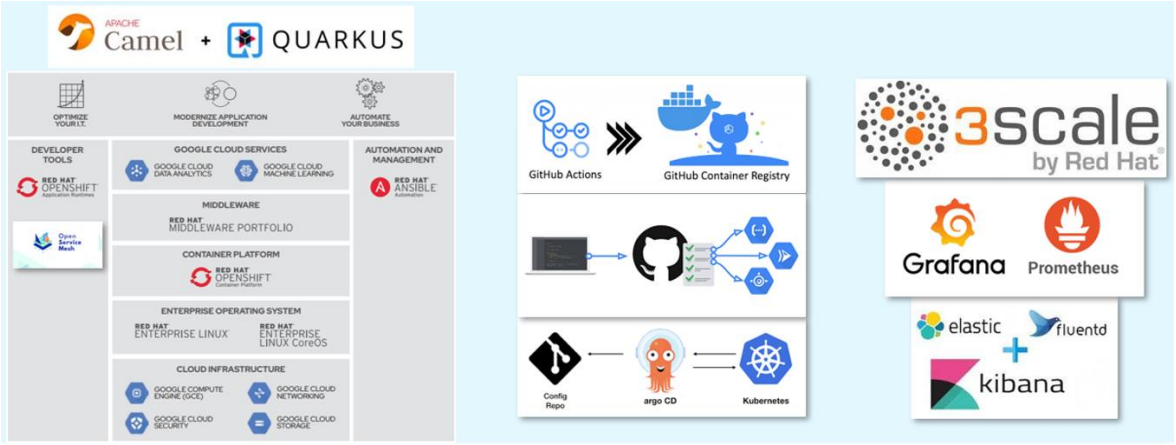

Figure 1 Technology landscape chosen by the client

Furthermore, as the company evaluates the adoption of AI capabilities, it aims to assess the broader implications for its business model. Understanding how the integration of AI technology can not only enhance software development practices but also reshape the

company's value proposition and revenue streams is essential for strategic decision-making and long-term sustainability.

This research attempts to identify and analyse concrete use cases where implementing AI-driven solutions on the Red Hat OpenShitft platform can lead to cost savings in software development for the client. By examining the intersection of AI technology, software development efficiency, and business model innovation, this study aims to provide actionable insights that enable the client to harness the full potential of AI while navigating the evolving landscape of digital transformation in social welfare and healthcare.

Integration is connecting different software components, applications, and data sources. Integration platforms are software tools that facilitate integration solutions. Red Hat OpenShift is a leading integration platform that leverages AI and machine learning to provide intelligent features for integration tasks. This research focuses on the Red Hat OpenShift platform, the client's chosen development tool, especially in the Quarkus and Apache Camel frameworks.

Figure 2 Scope defined for the research and thesis: Red Hat OpenShift and development frameworks Quarkus and Apache Camel.

The development work on the integration platform is partly done with in-house resources and primarily through purchased consulting services.

1.3 Aim of study

Through a comprehensive analysis of real-world scenarios and practical applications, the intention is to provide actionable insights and recommendations to optimise the efficiency and effectiveness of integration development processes within the organisation.

In addition to identifying use cases, another crucial aspect of this research is to evaluate the impact of integrating AI technologies on the organisation's business model. The approach is to comprehend the broader implications for the organisation's operations, strategic planning, and revenue streams by evaluating how AI-driven integration development can optimise resource allocation, streamline workflows, and accelerate project timelines.

This study provides valuable insights and practical recommendations to empower the organisation to harness AI's transformative potential in integration platform development. By strategically leveraging AI technologies, the organisation can augment its competitive edge, drive innovation, and achieve sustainable growth in the rapidly evolving digital landscape.

2 Methods

The research method chosen for this thesis is qualitative and will use a case study methodology. This methodology aims to offer descriptive information and propose theoretical significance. Detailed descriptions allow for a deeper and more refined comprehension of the case.

2.1 Description of the research

The research will analyse AI's impact on programming processes and the business model, exploring its potential to optimise resources and accelerate project timelines. The findings of this study will provide insights into AI's impact on integration development processes and support the client in making informed decisions about implementing AI-based integration platforms.

In the digital era, integration platforms like Red Hat OpenShift are pivotal in connecting diverse applications. This study investigates how AI technologies can enhance such

platforms to uncover practical use cases, optimise development processes, and improve cost efficiency and productivity. Analysing AI's impact on programming and business models offers valuable insights to support informed decision-making in implementing AI-powered integration.

2.2 Case studies methodology

When conducting research, choosing the right research design that best suits your objectives is essential. A case study is one type of design often suitable for obtaining solid, situational, and detailed information about a specific real-world subject.

A case study is a qualitative research method that involves an in-depth examination of a single event, person, group, or phenomenon. It allows the researcher to explore the case's key characteristics, meanings, and implications in detail. This approach is considered valuable when the researcher seeks a thorough understanding of a complex and multifaceted subject.

This thesis was conducted by using efforts to gather information from a wide range of reliable sources covering AI, ML, DevOps, Red Hat, and related technologies. Using a qualitative research approach, case study methodology informed the investigation, enabling systematic data analysis and theory development. Deep engagement in literature and iterative data processing aimed to reveal findings and generate empirical knowledge. Through careful inquiry, valuable contributions to academia and practical insights for industry practitioners are expected, enhancing understanding and advancing research in technology implementations.

Case studies provide a focused and manageable research project when the researcher does not have the time or resources to conduct large-scale studies. By analysing a single case, the researcher can gain valuable insights into the subject matter that can be used to develop theories, models, or frameworks.

In summary, a case study is an appropriate research design when a researcher wants to comprehensively understand a specific real-world subject. It allows for exploring key case characteristics, meanings, and implications and is useful when resources or time are limited.

2.3 Key research questions

The research method for the technical part of this thesis work was mainly based on case studies of several proof-of-concept implementations related to the Red Hat platform. The following research questions can be investigated:

- ***What AI-driven techniques*** are best suited for integration platforms like Red Hat OpenShift, and how can they support the company's operations and objectives?
- ***To what extent*** does the use of AI affect the amount of work in development processes compared to traditional methods?
- ***How will the adoption*** of AI in programming tasks impact the company's existing business models, and what benefits and challenges are expected in specific areas of developing the integration platform?

The research methodology involves analysing primary and secondary sources, focusing on AI's impact on programming processes and business models.

3 Theoretical framework

This section will discuss AI's technical concepts, principles and applications, such as natural language models, ChatGPT, neural networks, and deep learning. These concepts are essential for understanding how AI can enhance the user experience of chatbots in integration platform ecosystems. The theoretical framework will also discuss other technologies and tools that are important and used by the client.

3.1 Artificial Intelligence from a historical perspective

The early history of AI and ML is covered in this section, and how these technologies have evolved is explored. The two approaches to developing AI are discussed, and how hybrid systems have been created by combining the strengths of both approaches is explained.

3.1.1 First AI models

Artificial Intelligence and machine learning have deep historical roots dating back to the 1950s when the first models in this field were developed. Today, AI is established as a branch of computer science (Hauck et al., 2019, p. 143110).

As Bodgan M. Wilamowski and J. David Irvin stated 2018 that the oldest intelligent systems, solving problems using AI methods, were so-called Neural Networks (NN), which could solve practical problems using learning. (Bogdan M. Wilamowski & J. David Irwin, 2018, pt. 1.3)

In autumn 2022, the public gained access to ChatGPT, an innovative product from OpenAI that marked a significant milestone in AI technology. ChatGPT is a robust large language model (LLM) capable of generating human-like text through conversational prompts and context, utilising statistical language patterns derived from a vast collection of online text. LLM AI is a remarkable advancement with the potential to draft full scientific manuscripts.(Macdonald et al., 2023, p. 219)

3.1.2 Symbolic and connectionist AI

In the past, AI development has been driven by two approaches: symbolic AI and connectionist AI. Symbolic AI uses logic and rules to solve problems, while connectionist AI relies on neural networks to process information. Connectionist AI models learn through training and operate like the brain. (Shavlik, 1994, pp. 321–322)

While symbolic AI systems were seen as more rule-based and straightforward, connectionist AI models were viewed as more complex but with more potential for *learning and adaptability*. (Russel & Norvig, 2022, p. 42).

Over time, when technological advancement, such as improvements in computer power, enabled researchers to develop more sophisticated AI systems. By recognising the value of both approaches, the researchers have unlocked the potential of artificial intelligence. By creating hybrid AI systems, they have found a way to combine the strengths of each approach, resulting in a more robust and efficient system. Factors that further influenced the development of AI at this time were the growing amount of available data, research, and different types of experiments conducted by researchers. (Shavlik, 1994, p. 323)

Despite their differences, symbolic and connectionist AI have made impressive contributions to the field. These two approaches grew together over time, and in the 1990s, they were merged into a hybrid AI design, as described by Towell and Shavlik in the article Knowledge-based Artificial Intelligence 1994. (Towell G. Geoffrey & Shavlik W. Jude, 1994).

ChatGPT, the conversational interface, was created via early AI technology advancements, vast research data, and 1990s research-based algorithms. It uses a robust language model with statistical language patterns from online texts, making it an exceptional tool for various settings.

3.2 Artificial intelligence and machine learning – an overview

AI is an area of computer science that aims to develop machines capable of performing tasks that require human-like intelligence. Natural language models are mathematical models that generate or understand natural language based on statistical methods. ChatGPT-type interfaces (prompts) include computer programs that use natural language to interact with humans via text or speech. Neural networks are models that copy the structure and function of biological neurons and learn from data. Deep learning uses multiple layers of these models to complete complex tasks.

3.2.1 Artificial intelligence

What is Artificial Intelligence? AI is the replication of human cognitive abilities in computer systems. These processes include learning, which involves acquiring information and rules for utilising it; reasoning, which involves using rules and algorithms to arrive at approximate or definite conclusions; and self-correction.

AI applications include expert systems, speech recognition in language models, and machine vision. AI can be divided into narrow AI, designed for a specific task, and general AI (GenAI), which displays human-like intelligence and can perform any intellectual task that a human can.

Russell and Norvig (2022) describe the process of defining and blueprinting an AI program as including the following human-like capabilities: natural language processing, knowledge representation, automated reasoning, machine learning, computer vision, speech

recognition, and robotics. In addition, the program must be able to behave according to a human cognitive model. In other words, the program should be able to act in a human way. (Russel & Norvig, 2022, pp. 19–22)

3.2.2 Machine learning

While AI encompasses a broad spectrum of techniques and approaches, machine learning (ML) involves analysing data with algorithms, learning from it, and ultimately making decisions or predictions. In ML, existing data (experience) is used to generate knowledge. The available data are assigned to classes. Examples are differentiating e-mails into spam and non-spam or deciding whether a football player should be hired. (Hauck et al., 2019, p. 143110)

Primary ML typically operates on cloud-based computing resources, accessing vast datasets stored in remote servers. It meticulously iterates through the information, constructing a model based on the most optimal outcome it achieves during the process. The resulting model then gives the user predictions for various scenarios. (Russel & Norvig, 2022, pp. 669–670)

3.2.3 ChatGPT and language processing

ChatGPT is a user interface that uses advanced natural language models (NLMs) to perform various language-related tasks, such as answering questions, generating text, and even composing poems. These NLMs are trained on massive amounts of data, making them competent and efficient (Mattas, 2023).

ChatGPT offers human-like text generation, image processing, audio capabilities, and video support. It uses natural language models to process and generate information, providing users with an efficient tool for diverse content generation and robust search capabilities.

3.3 Integration platforms and development tools

Integration Platforms have been widely used for decades to facilitate seamless communication and data flow between systems and applications. Integrating systems and applications through platforms is critical in reducing maintenance costs and accelerating

the launch of various development tasks. This approach is crucial for minimising time to market, which is essential for achieving business goals. (Linthicum David S., 2003, pp. 1–4).

3.3.1 Development and maintenance

Developing new features and performing maintenance work on integration platforms requires a good understanding of various technologies and programming languages. In 2019, the Environmental Campus Birkenfeld's Institute for Software Systems researched the challenges associated with developing new features and maintaining integration platforms, focusing on the Internet of Things (IoT).

Their findings highlighted a considerable demand for expertise across various domains, including protocols, embedded programming, distributed systems, and development tools. Additionally, their investigation emphasised the potential of AI as a cost-effective solution in this field, offering promising opportunities for savings and progress. (Hauck et al., 2019).

3.3.2 Open-source solutions

Open-source software (OSS) is developed collaboratively, with the source code accessible to all users. Unlike proprietary software, where only the original authors can modify the code, OSS encourages community contributions and improvements. Examples of well-known OSS applications are the Firefox and Chrome browsers and Android operating systems. (Synopsys, 2024)

OSS tools and frameworks are essential in software development, especially in today's fast-changing technological landscape. In his *'Beginners Guide to Open-Source Innovation,'* Ingo Boering outlines how OSS development can be split into three periods. The first one starts as far back as the mid-80s with OSS Operating Systems, with Linux, one of the most prominent, moving over to OSS databases, web servers, and compilers. The last period can be seen now when AI and ML tools are incorporated into the software (SW) development processes.

These tools enable developers to collaborate, innovate, and deploy solutions more effectively. The programming language Python, combined with the TensorFlow OSS platform and ML capabilities from PyTorch, as highlighted by Boering, plays an essential role in the SW industry, particularly with the increasing prominence of AI and ML

technologies. Open-source platforms have made AI technologies more accessible and flexible for everyone. (Boering, 2021)

The primary motivation for developers and businesses to select OSS in their SW development programs is free access to the code and design files in the chosen frameworks and toolkits. Other benefits, such as community support and customisation options, shall also be mentioned when discussing the benefits of OSS. The array of available OSS frameworks and toolkits in contemporary software development is extensive and diverse. Developers use open-source resources like Python, TensorFlow, and PyTorch to create, test, and implement AI in their SW solutions.

Some interesting examples where AI and ML tools have been successfully utilised in SW development are OpenAI's Codex and GitHub's Copilot, which use AI to assist developers in generating code snippets and suggesting solutions based on natural language descriptions. If looking into the critical part of testing, AI-driven tools like Applitools and Testim utilise machine learning algorithms to automate test case generation, execution, and analysis. And finally, the commonly known ChatGPT. (White et al., 2019, pp. 2–3)

The open-source technologies the client has selected for the SW development are briefly introduced in this section before moving on to the next one. *RedHat Openshift is a cloud-based platform* that enables developers to build, deploy, and manage containerised applications based on Kubernetes, an open-source system for orchestrating containers across multiple hosts. *Apache Camel is an open-source integration framework* offering components and patterns for connecting various systems and services, supporting multiple protocols and data formats, such as HTTP, REST, SOAP, FTP, CSV, JSON, and XML.

Quarkus is an open-source library set that optimises Java applications for Kubernetes and serverless environments. It offers a fast startup time, low memory usage, and live reload capabilities. Developers can use these tools and frameworks to create secure, scalable, and resilient software solutions that utilise AI's power.

In conclusion, integrating open-source tools and frameworks has become indispensable in modern software development, revolutionising how developers collaborate, innovate, and deploy solutions. From the emergence of OSS operating systems in the mid-80s to the current era of AI and ML technologies, open-source platforms have continuously evolved to meet the dynamic demands of the industry.

3.3.3 Integration Platform As A Service

Integration Platform as a Service (iPaaS) has become essential to modern software architectures. It enables effortless communication and data exchange between different applications and systems. iPaaS provides cloud-based integration solutions that function as centralised, application-independent systems connecting applications within any enterprise.

Around the turn of the millennium, the term ‘‘enterprise application integration’’ (EAI) gained popularity among IT departments. Until then, silo applications – e.g. custom-developed or packaged, host- or client–server-based—were connected through numerous point-to-point interfaces within system landscapes. The transition from traditional EAI, characterised by complex point-to-point connections, to iPaaS, marked a shift towards more streamlined and efficient integration solutions. (Ebert et al., 2017, pp. 375–376)

iPaaS solutions provide various benefits in today's business landscape. They streamline integration processes, enhance operational efficiency, and improve agility in responding to changing business requirements. By providing cloud-based integration capabilities, iPaaS platforms enable organisations to adapt and scale their integration infrastructure more effectively, which is particularly important in today's constantly changing business environment.(Ebert et al., 2017, p. 376)

Furthermore, iPaaS platforms hold the potential for synergy with AI technologies. By harnessing the power of AI, integration platforms can automate routine tasks, optimise workflows, and provide intelligent insights into data integration processes. Integrating AI capabilities into iPaaS solutions enhances their functionality and enables organisations to leverage data more effectively for informed decision-making and strategic planning. (Dhruv, 2023)

3.3.4 Red Hat OpenShift framework

OpenShift is a container application platform developed by Red Hat that leverages technologies such as Docker and Kubernetes. Docker provides a standardised way to package and distribute applications and their dependencies into lightweight containers, enabling consistent deployment across different environments. Kubernetes is an open-source container orchestration platform that automates containerised applications'

deployment, scaling, and management. Together, these technologies form the foundation of OpenShift, allowing organisations to streamline their application development and deployment processes with efficiency and scalability. (Red Hat, 2023)

By providing a self-service environment for on-demand application creation and deployment, OpenShift reduces development time and accelerates the release cycle. Additionally, OpenShift simplifies the process of composing applications by focusing on service composition rather than individual application objects. This approach makes it easy to integrate services, such as web containers reusing databases or exposing databases directly to the network edge. (Cisco, 2019)

In conclusion, the Red Hat OpenShift Container Platform offers a robust solution for organisations leveraging container technologies for application development and deployment. By combining the strengths of Docker and Kubernetes, OpenShift provides a user-friendly environment for managing containerised services, promoting agility, scalability, and efficiency in modern IT infrastructures.

3.3.5 Apache CAMEL integration framework

Apache Camel is an open-source integration framework designed to help developers build resilient and scalable applications. One key feature is its ability to support various Domain-Specific Languages (DSLs) such as Java, XML, Groovy, Kotlin, and YAML. This feature ensures developers can write code in a language they are comfortable with, improving productivity and reducing the learning curve.

Additionally, Apache Camel is based on established Enterprise Integration Patterns. This means that it provides a set of predefined patterns that developers can use to define routing and mediation rules. This feature ensures that developers can rapidly and easily create complex integration solutions without starting from scratch.

Apache Camel also uses Uniform Resource Identifiers (URIs) to work with different transports or messaging models. This feature makes working with other systems, including protocols such as HTTP, FTP, JMS, and similar systems, easier. Furthermore, Apache Camel offers pluggable Components and Data Format options. This means developers can use different data formats and components depending on their needs.

Finally, Apache Camel provides seamless integration with frameworks such as Spring. This ensures that the integration works smoothly and that the application is stable. Furthermore, Apache Camel has extensive support for unit testing routes. This feature ensures developers can thoroughly test their integration solutions and catch any issues early in development. (2004-2024 The Apache Software Foundation, 2024)

3.3.6 Quarkus Java framework

Quarkus is a modern framework for Java developers working in a cloud-native environment. Its Kubernetes-native architecture makes it a powerful toolset for building robust applications with containers and microservices. (Quarkus, 2023)

As developed, Quarkus comprises high-quality Java libraries that conform to industry standards. Its mission is to establish Java as the primary platform for Kubernetes and serverless environments while empowering developers to address diverse distributed application architectures.(Oh, 2021)

Java developers can benefit from Quarkus's range of practical advantages. Its reactive and event-driven nature allows for the creation of scalable and efficient applications. In addition, Quarkus includes various features that enable developers to optimise their applications for resource utilisation, leading to more efficient performance on cloud platforms.(Oh, 2021)

In addition to its many features, Quarkus includes an extensive set of extensions that developers can use to add new functionality to their applications without writing additional code. This allows developers to focus on building the core logic of their applications instead of spending time on boilerplate code.

Modern SW architectures rely heavily on integration platforms, facilitating smooth communication and data flow between systems and applications. iPaaS – solutions offer cloud-based integration solutions that enhance operational efficiency and improve agility. When combined with AI, iPaaS presents exciting prospects for automation and optimisation. Thanks to its efficiency and scalability, the Red Hat OpenShift Container Platform is an excellent tool for streamlining application development and deployment. By leveraging integration platforms, organisations can cut costs, speed development, and

successfully achieve their business goals. The following sections will discuss potential AI and other technological solutions that can be used to improve the integration platform's effectiveness.

3.4 AI and ML in integration platforms

This chapter will explore the intersection of artificial AI and ML technologies with integration platforms. Although integration platforms are significant, they pose challenges and expenses in their development and maintenance. The upcoming sections will cover crucial technical aspects.

3.4.1 Foundational concepts

The following section covers foundational concepts in machine learning, including discussions of deep understanding, supervised and reinforcement learning, and backpropagation. It also introduces OpenAI, a non-profit organisation that researches and creates safe and beneficial AI systems.

The technologies discussed in the following sections are essential and represent the backbone of AI and machine learning technologies. These revolutionise various fields by automating complex data processing and feature extraction. AI's capability to handle diverse tasks, learn from vast data, and adapt through reinforcement learning fosters innovation and efficiency, transforming industries and unlocking new possibilities. These are processes and technologies behind AI tools, such as ChatGPT and Agents, which are discussed in the use cases of this research.

3.4.2 Machine learning methods

When discussing AI and ML, particularly *deep learning (DL)* techniques, some prominent researchers, such as Geoffrey Hinton, Yann LeCun, and Yoshua Bengio, cannot be overlooked. The representation learning model and deep learning field have greatly

benefited from the work of these researchers, who have been instrumental in their advancement. Additionally, they have developed deep learning techniques in areas such as medical image analysis, drug discovery and genetic data to understand the genetic determinants of diseases. They are also behind the development of AI and ML in the transportation field.

In the past, ML technology became an essential tool in trading and sales. It was used in various web types of searches, such as content filtering on social networks and e-commerce recommendations. Additionally, it became highly efficient in trading for consumer products like home electronics. ML technologies have been gradually applied across various domains, including healthcare, finance, and transportation, based on their specific needs and technological advancements.

Traditional machine learning techniques required careful engineering to design feature *extractors*, such as *methods and algorithms*, that could transform raw data into a suitable internal representation for pattern detection or classification. This method is referred to as the *representation learning* model. Furthermore, Hinton, LeCun, and Bengio describe in their research that deep learning is *a type of representation learning* where multiple layers of non-linear transformations are used to learn hierarchical representations of data. Utilising DL, automatic extractors of hierarchical features, scalability with increased computational resources, strong generalisation capabilities, and ongoing algorithmic advancements have made representation learning easier due to their ability to extract features automatically, understand multiple levels of abstraction, scale with computational power and data, generalise well to unseen examples, and benefit from ongoing algorithmic advancements. (Hinton et al., 2015, pp. 436–437)

After explaining representation learning in the preceding section, the focus will be exploring *supervised learning* and *backpropagation models*. Supervised learning is ML, where the model is trained on labelled data; the data is paired in input data with corresponding and correct output data. (Hinton et al., 2015, p. 439) One interesting way of performing supervised learning is when a ‘free-ware application’ asks the user the question ‘Are you a robot?’ followed by the question 'Please select all the pictures *where a bicycle can be seen’*.

Additionally, reinforcement learning is machine learning. In conclusion, the role of Natural Language Processing is to maximise some notion of cumulative reward. This approach is well-suited for dynamic and uncertain environments, making it applicable to various tasks, including autonomous decision-making and control. Reinforcement learning algorithms, such as Q-learning and policy gradient methods, can be applied to train *agents* to make intelligent decisions over time based on trial and error and environmental feedback. (Shavlik, 1994)

Backpropagation is an algorithm for training neural networks. Backpropagation can be utilised for image classification and training deep neural networks. A specific branch of backpropagation is known as recurrent neural networks (also referred to as RNNs), which incorporate an analysis of sequential inputs. This has been very useful when speech and language models have been developed. One of the most promising applications is natural language processing NLP, which is the ability of machines to understand and generate natural languages. NLP can enable many valuable tasks, such as machine translation, text summarisation, sentiment analysis, question answering, and conversational agents. (Shavlik, 1994)

3.4.3 Open AI

OpenAI is a non-profit organisation committed to ensuring that artificial general intelligence (AGI) benefits humanity. The organisation strives to create safe and beneficial AI systems, conduct innovative research, and revolutionise work and creativity through AI. OpenAI focuses on research on generative models and aligning them with human values. The ultimate goal is to create AGI that is both safe and beneficial for society.

OpenAI offers an API platform that provides access to its latest models, including ChatGPT. The organisation is dedicated to understanding the potential risks and benefits of AI. It emphasises safety and responsible development to mitigate any negative impact on society.

The team at OpenAI comprises leading researchers and developers committed to pushing the boundaries of AI research and development. OpenAI's work is focused on creating advanced AI technologies capable of transforming industries and enhancing human lives.

With a strong focus on safety and responsible development, OpenAI is shaping the future of AI in a way that benefits all of humanity (OpenAI, 2024).

3.4.4 Natural language processing

Natural Language Processing (NLP) background technology involves various techniques and methodologies for analysing and understanding human language using computational methods.

Several methodologies are available for NPL, including machine learning and deep learning, which were discussed in previous sections. Data preprocessing, feature extraction, topic modelling, sentiment analysis, named entity recognition, and text summarisation are fundamental in extracting insights, identifying patterns, and deriving meaning from textual information. (Shaik et al., 2022, p. 56722)

Data preprocessing involves cleaning and transforming raw text data into a format suitable for analysis by removing special characters, tokenising text into words or sentences, handling capitalisation, and applying techniques like stemming or lemmatisation. Feature extraction and topic modelling are essential techniques in NLP for extracting meaningful information from textual data. Feature extraction involves converting text data into numerical representations, such as Bag-of-Words, TF-IDF, or word embeddings, to enable machine learning algorithms to process the data effectively.

Another method is topic modelling, which automatically identifies hidden topics or themes within a collection of documents, aiding in organising and summarising textual content. By combining feature extraction and topic modelling, NLP systems can efficiently analyse and categorise text data, uncovering key patterns and insights for various applications such as information retrieval, document clustering, and content summarization. (Shaik et al., 2022, pp. 56732–56733)

Sentiment analysis, also known as opinion mining, determines the sentiment or emotional tone expressed in a text, such as whether it is positive or negative. Named Entity Recognition is a technique used to identify and classify named entities mentioned in text into predefined categories such as names of persons, organisations, locations, dates, and

more. Text summarisation generates a concise and coherent summary of a longer text while retaining the most important information. (Shaik et al., 2022, pp. 56720, 56729)

These methodologies form the backbone of NLP applications, facilitating tasks ranging from information retrieval and sentiment analysis to machine translation and text summarization. By leveraging these methodologies, NLP systems can effectively process and interpret natural language data, opening many possibilities for enhancing human-computer interaction, automating tasks, and deriving valuable insights from textual content.

In conclusion, the role of NLP technologies in modern computing systems cannot be overstated. These technologies enable computers to understand, interpret, and generate human language, paving the way for the development of assistants or co-pilots that assist in more demanding tasks such as text generation, for example, in different programming tasks or demanding analyses.

3.4.5 AI in programming tasks

This section discusses ChatGPT's roles in programming and development tasks. It discusses ChatGPT's various capabilities, such as code completion, correction, snippet prediction, error fixing, optimisation, and technical query answering, along with its potential to enhance code quality and troubleshoot effectively. Additionally, the emerging field of MLOps will be introduced, highlighting its fundamental principles and benefits for enabling organisations to leverage the power of machine learning.

3.4.6 Boosting programming efficiency with ChatGPT and NLP

In the introduction of the article *Role of ChatGPT in Computer Programming*, Biswas states that computer programming is a complex field that requires a deep understanding of programming languages, algorithms, and data structures. (Biswas, 2023)

The article also discusses ChatGPT’s role in computer programming, highlighting its capabilities such as code *completion, correction, snippet prediction, error fixing, optimisation, and technical query answering*. Keeping up with the latest technologies and best practices can be challenging for developers. It requires staying on top of industry trends, identifying potential areas of improvement, and implementing new strategies to

drive success. However, with ChatGPT, developers have access to a powerful tool that can help them stay informed, efficient, and successful. By leveraging ChatGPT, developers can receive valuable guidance on the latest trends and best practices and assistance in identifying and fixing technical issues. With ChatGPT's expert assistance, developers can work more productively, innovate more effectively, and succeed tremendously.

ChatGPT has wide-ranging roles in programming tasks, such as code completion and correction, code prediction and suggestion, automatic fixing of syntax errors and common mistakes, and code optimisation and refactoring suggestions. (Biswas, 2023, pp. 9–12).

Guides exist for using ChatCPT with frameworks like Quarkus. They cover creating a new application, configuring it to talk to ChatGPT, and defining the REST API (Application Programming Interface) with annotations in Java interfaces. By integrating AI capabilities, development and problem-solving processes are streamlined. ChatGPT is a valuable aid, empowering developers to enhance their code quality and troubleshoot effectively. Its versatility allows it to undertake diverse tasks, from answering queries to generating code snippets. (Bueno, 2023)

Developers may find themselves stuck and unable to proceed when encountering a code problem. During such circumstances, ChatGPT can offer priceless aid to developers. ChatGPT can provide viable solutions or guidance through a simple explanation of the situation. This service can potentially save critical time and prevent common errors from occurring. Silkin also highlights the limitations of *relying on ChatGPT*, such as the possible drawbacks of receiving incorrect or outdated information. ChatGPT and other Copilot-type solutions should be seen as assistants, not replacements for skilled human resources. (Silkin, 2023)

These ChatGPT roles can also make a practical and helpful base for improving efficiency in programming, development, and maintenance tasks for the client.

3.4.7 Machine Learning Operations (MLOps)

DevOps is a software development practice emphasising communication, collaboration, and automation between development and operations teams. By streamlining the

software delivery process, DevOps aims to reduce the time it takes to develop and deploy software. (Beattie et al., 2021)

MLOps, or *Machine Learning Operations*, is a relatively new and exciting domain closely related to DevOps. It refers to the set of methods and tools that are used to deploy and manage machine learning models in, for example, software production. Unlike traditional software development, machine learning involves both code and data that are constantly changing. This poses new challenges for testing, monitoring, debugging, and updating the models. MLOps seeks to bridge the *gap between data science and engineering* by establishing best practices and tools that enable faster and more reliable delivery of machine learning solutions.

Some fundamental principles of MLOps include version control, continuous integration and delivery, and automated testing and monitoring. Version control enables data scientists to track changes in the models, code, and data over time, while continuous integration and delivery will allow them to deploy updates quickly and efficiently to the models. Automated testing and monitoring help to ensure that the models are accurate, reliable, and performing as expected. Overall, MLOps is an emerging field that holds great promise for enabling organisations to leverage the power of machine learning to solve complex business problems. By incorporating MLOps into their DevOps processes, organisations can accelerate their time-to-market, reduce costs, and deliver more innovative and impactful solutions. (Symeonidis et al., 2022)

3.4.8 AI In SW Development Process

One of the pioneer studies on this topic was done by Kulkarni and Padmanabham (2017). These researchers presented a *framework for incorporating AI activities into various stages of software development*, such as requirement analysis, design, coding, testing, and maintenance. They also proposed metrics to assess the impact of AI on different stages of software development. These metrics evaluate the achievement of usability goals within software applications, such as user satisfaction, efficiency, learnability, and error rates. Other metrics assess the level of integration of AI activities within the software Development process. It may encompass factors such as the extent of AI utilisation across

different development stages, the depth of integration, and the impact on overall development efficiency and effectiveness.

Earlier, the industry primarily utilised the waterfall model for software development, but the landscape has shifted with the emergence of agile methodologies. Researchers like Kulkarni and Padmanabham (2017) have explored how AI frameworks can be tailored to accommodate both models, providing a reassuring adaptability. This adaptability is crucial as modern software development, including DevOps practices highlighted in Machine Learning Operations (MLOps), increasingly leans towards agile methodologies for their flexibility and iterative approach.(Kulkarni & Padmanabham, 2017, p. 18)

The researchers used AI to support the programming process with *intelligent agents*. These agents were used to automate repetitive tasks, summarise complex data, and provide recommendations, aiding the software development process. As ML was trained, AI allowed computers to learn without being explicitly programmed. AI was utilised to find *cost-effective choices* and achieve high performance within given constraints, which is important for planning and scheduling in software development.

Today, AI plays a significant role in automating repetitive tasks in software development, simplifying the building of custom machine-learning models. Automation of machine learning (AutoML) streamlines tasks such as model selection, hyperparameter tuning, and training, relieving developers of the manual effort required and significantly speeding up the development process. (Beres, 2023)

AI integration in SW development processes offers meaningful benefits to developers. Developers can receive assistance and support through AI technologies throughout the programming process. AI tools can provide valuable assistance by offering examples and identifying errors the human eye may overlook. This enhances the efficiency of development workflows, improves code quality, and reduces the likelihood of mistakes. AI integration empowers developers to streamline their processes and produce higher-quality software products. (Kulkarni & Padmanabham, 2017, pp. 19–20)

3.4.9 AI In Distributed Systems

Node and block-based development tools have emerged as invaluable resources for constructing distributed systems integrated with AI applications. These tools offer intuitive

visual interfaces, facilitating the design, deployment, and management of intricate systems while harnessing the power of AI technologies. *Node-RED, Ardublockly, Scratch, and Ernest* are examples of node and block-based development tools used for developing applications for various purposes, including distributed systems and AI-based business models. (Hauck et al., 2019, pp. 43110–143111)

These development environments facilitate the seamless data flow between visual interfaces, allowing users to define computational tasks and specify data transformations easily. Rather than laboriously writing code to articulate procedures and manage communication channels, users can visually connect blocks or nodes, each representing distinct operations or data transformations. (Hauck et al., 2019, pp. 143109–143110)

The development of visual programming approaches represents a departure from traditional coding methodologies like object-oriented programming (OOP). OOP, first developed by Norwegian scientists in the late 1960s, has significantly democratised software development by enabling code reuse, modular design, and abstraction.(Black, 2013, pp. 5–7)

From a development and programming perspective, one of the tasks of this research is to find ways for technical resources without deep programming experience to contribute to complex tasks such as building and maintaining integration platforms. Another area of interest is to find methods that enable individuals with less advanced programming skills to participate in more demanding programming tasks.

3.5 AI-driven Business Model

The integration of AI technologies has revolutionised the business landscape, with notable examples such as Airbnb, Uber, and Amazon leading the charge. These companies use AI to fuel innovation and gain a competitive advantage. Shrutika Mishra and A. R. Tripath's discussion in their journal *'AI Business Model: An Integrative Business Approach'* (2023) underscores the foundational role of AI and ML in these commonly known platforms. AI's role in automating processes, deriving insights from data analysis, and engaging with Customers and employees ultimately leads to innovative business dynamics and enhanced decision-making capabilities. AI's integration into business models facilitates a more efficient, data-driven approach to business operations. (Mishra & Tripathi, 2021, pp. 1–3)

Businesses leverage AI technology to revolutionise their business models in several key ways. Firstly, AI-enabled personalisation leads to Enhanced Customer Experience, where companies use data to anticipate and meet customer preferences, thereby boosting satisfaction and loyalty. Secondly, Data-Driven Decision Making empowers organisations to make strategic choices based on AI-generated insights, instilling confidence in the decision-making process and leading to better outcomes and competitive advantages. Additionally, AI facilitates business model experimentation by enabling cost-effective simulation of different scenarios, allowing companies to test innovative models before full-scale implementation and further enhancing their confidence in the potential success of these models.(Lee et al., 2019, p. 1)

AI propels business model innovation by enabling product innovation and risk management excellence. By analysing market trends, consumer behaviour, and competitor activities, AI proactively empowers companies to develop products and services tailored to evolving customer needs, instilling a sense of security in their ability to meet market demands. Moreover, AI aids in risk identification and mitigation by analysing patterns, enhancing business resilience, and minimising disruptions, thereby preparing companies for potential risks. Utilising AI for product innovation and risk management helps companies stay ahead of the curve, drive growth, and maintain competitiveness in today's dynamic business landscape.(Lee et al., 2019, pp. 9–10)

AI and ML are critical in driving business innovation by enabling enhanced customer experience, data-driven decision-making, business model experimentation, product innovation, and risk management excellence.

3.6 AI-driven Cyber Security

The role of AI in safeguarding internet-connected systems from cyber threats, attacks, and unauthorised access unveils several critical challenges. AI technologies offer promising solutions to combat evolving cyber threats and enhance the resilience of internet-connected systems. The escalating complexity of cyber-attacks necessitating advanced detection and prevention methods; the overwhelming volume of data generated by systems, posing a challenge for traditional security systems to process and analyse effectively; and the reliance of conventional security systems on ad-hoc data processing,

lacking the intelligence and dynamism required to meet the demands of modern cybersecurity. (Sarker, 2021, pp. 1–2)

Some examples of cyber types of disturbance are malware attacks, ransomware, and phishing. These incidents can cause significant disruptions and financial losses, emphasising the need for advanced detection and prevention, which AI can provide. AI, ML, and DL technologies offer practical solutions and tools to work against malware attacks. These can detect cyber anomalies or attacks. In cybersecurity, deep learning methods can be used for various purposes, such as detecting network intrusions, detecting and classifying malware traffic, and backdoor attacks. (Sarker, 2021, pp. 8–9)

Cybercriminals increasingly use NLP and security defence tools that understand and process unstructured data, which are useful for prevention. NLP can also be applied to analyse unstructured data like user behaviour or system logs for phishing detection and vulnerability analysis. (Sarker, 2021, pp. 9–10)

In conclusion, the use of AI in cybersecurity has become increasingly important as the complexity and frequency of cyber-attacks continue to rise. AI offers promising solutions to detect and prevent various types of cyber threats. These technologies can help enhance the resilience of internet-connected systems and provide advanced detection and prevention methods to combat evolving cyber threats. However, as with any rapidly developing technology, the potential risks associated with AI must be considered and addressed to ensure that it is used safely and ethically.

4 Result based on literature review

This study explores the integration of AI technology for DigiFinland Oy within the RedHat Integration platform and its impact on business models. The primary objective of this research is to identify use cases within a specific field where AI technology can be applied and propose relevant enhancements to the business plan based on those findings.

The client's expectations and requirements are the basis for the AI-driven use cases in this research and are applied to areas connected to the integration platform's development process. The research scope includes Red Hat OpenShift, Apache CAMEL, and Quarkus frameworks, which are development tools used daily by programmers.

The use cases presented in this chapter have evolved into two clear branches during the research process. One branch involves using AI as a ChatGPT solution to directly support programmers in their work in the Quarkus Framework. The other branch utilises AI and machine learning as background processes or agents for monitoring and error handling in the Apache CAMEL framework. If the scope had been defined differently, other use cases like GRAFANA could have been researched and discussed in more detail. AI solutions in the GRAFANA framework will be addressed only as a future possible research area in this thesis.

4.1 AI behind ChatGPT and agents

This chapter explores the AI behind ChatGPT and agents, particularly within the context of NLP technology and Apache Camel. It delves into the foundational concepts of Connectionist AI and its crucial role in developing NLP models.

4.1.1 ChatGPT and 'prompting'

As described earlier, *Connectionist AI* played an essential role in developing the earlier NLP models. It effectively incorporated prior knowledge into neural networks, which is necessary for complex NLP tasks requiring understanding and context.

The ability to extract symbolic rules from trained neural networks can aid in the interpretability and explainability of NLP models. Symbolic rules also make NLP models more transparent and trustworthy. The integration of symbolic and neural learning represents a hybrid approach that leverages both methods' strengths, leading to a more robust and effective NLP, as it is known today.

NLP uses models trained on large datasets to generate texts to predict and produce sequences of words that form coherent sentences, also called the auto-regressive method. One essential technique employed in NLP is auto-regressive language generation, where models trained on large datasets predict and produce coherent sequences of words based on the previous context. Techniques like auto-regressive language generation are used, where the model predicts the next word based on the previous words in a sequence. Some well-known examples of NLP models are, for instance, GPT-4 from OpenAI and BERT developed by Google. (Yao, 2022)

Generating code examples with NLP follows a similar process to text generation. Both tasks involve training a model on a dataset specific to the desired output. For instance, when NLP is prompted to generate poetry, it is trained on a dataset of poems. Similarly, when generating code snippets, NLP is trained on a dataset of code examples, such as Java or Phyton programming code. The model learns the syntax and structure of the programming language, allowing it to produce new code snippets based on *given contexts* or prompts. Similarly to any prompting provided through a ChatGPT interface, an NLP trained in programming code generates programming code examples (snippets) in response. A request in a programming context, mentioning keywords related to programming languages, functions, variables, or specific programming problems, gives the user a program snippet – not a spring poem.

The literature review has identified several processes and tasks that ChatGPT can facilitate in programming situations. The technology has shown potential for supporting and coaching purposes, quality management, performance monitoring, and testing processes. These areas warrant further discussion and analysis in the upcoming chapters.

ChatGPT has exhibited high efficacy in providing developers with timely and relevant support. The technology has proven helpful in guiding developers through complex coding tasks, identifying potential quality issues, and monitoring the performance of software products. ChatGPT has also been instrumental in streamlining the testing process, enabling developers to deliver high-quality software products within tight deadlines.

As the transformative potential of NLP tools like ChatGPT in simplifying and accelerating programming tasks has been explored, it becomes clear that these technologies are poised to revolutionise the software development landscape within Red Hat's ecosystem. However, researching and utilising these technologies are ongoing processes with much yet to be uncovered. In the chapters ahead, a deeper exploration into specific applications of NLP within programming will be undertaken, examining how these tools can streamline code completion, enhance error detection, automate technical documentation generation, optimise code refactoring, and facilitate problem-solving and troubleshooting processes.

Overall, the findings of this literature review suggest that ChatGPT is an asset in the programming landscape. The technology has the potential to enhance developers' productivity and efficiency while facilitating the delivery of high-quality software products.

Conclusion:

NLP tools like ChatGPT have the potential to simplify and speed up programming tasks, improving developer productivity and efficiency. ChatGPT has proven effective in providing developers with timely and relevant support, guiding them through complex coding tasks, and identifying potential quality issues. It can enhance code completion, optimise code refactoring, and facilitate problem-solving and troubleshooting processes.

The technology has the potential to revolutionise the software development landscape within Red Hat's ecosystem by improving developers' productivity and facilitating the delivery of high-quality software products. In summary, NLP systems achieve greater interpretability, trustworthiness, and effectiveness by extracting symbolic rules and integrating them with neural network capabilities. Techniques like auto-regressive language generation and dataset training enable NLP to generate coherent text and code snippets.

4.1.2 Agents utilising AI

By utilising AI techniques, Apache Camel can effectively model agents' behaviour. These techniques involve analysing historical data and patterns to predict how agents behave in different scenarios. Using machine learning algorithms such as decision trees or reinforcement learning, agents can analyse past interactions and make informed decisions.

One of the main benefits of AI in Apache Camel is that it enables agents to make intelligent routing decisions based on real-time data and context. By analysing incoming messages or events, machine learning algorithms can identify the optimal route or processing logic for each message, thereby improving the efficiency and effectiveness of message routing.

Additionally, AI techniques allow agents to dynamically adapt their behaviour based on changing conditions or requirements. This means agents can continuously optimise their behaviour based on new patterns or environmental changes without manual intervention.

Another key advantage of AI and ML models in Apache Camel is their ability to perform predictive analytics. By using predictive models, agents can forecast resource usage, identify potential bottlenecks, or predict system failures before they occur. This allows for

proactive management and optimisation, resulting in a more streamlined and efficient system overall.

The distinction between AI in the form of NLP technology discussed earlier as ChatGPT or 'prompting' and AI agents is rooted in their level of visibility to the user, e.g. the programmer. ChatGPT is noticeable to the programmer and requires their input and guidance. At the same time, AI agents are designed to function autonomously, communicating with each other and collaborating to achieve their objectives. Unlike ChatGPT, which requires user programming and customisation, AI agents are self-sufficient, utilising advanced algorithms and machine learning techniques to adapt better and evolve to meet their objectives. This level of autonomy and collaboration makes AI agents particularly effective in complex and dynamic environments, where they can work together to solve problems and accomplish tasks more efficiently than traditional systems.

Apache Camel enables developers to create custom routing strategies based on specific conditions. These routing strategies are the decision-makers, similar to agents with AI capability. By studying the behaviour of these routing agents, developers can design effective routing rules, thereby playing a crucial role in the process. Endpoints can be dynamically selected based on runtime conditions using Apache Camel. Agents modify their actions based on contextual information. Developers ensure the selection of the right endpoints by studying the conduct of these agents and dynamically choosing them. (Li et al., 2023, pp. 2–4)

Error handling is crucial in integration scenarios. Agents responsible for handling errors play a critical role in the development process. Developers define error-handling routes (agents) in Apache Camel. These agents handle exceptions, retries, and dead-letter queues. By studying their behaviour, developers ensure robust error handling.

Integration flows involve multiple components working together, each of which can be seen as an agent. Developers study how these agents collaborate within the flow. Monitoring agents provide insights into system behaviour, which helps developers optimise routes and identify bottlenecks. Thinking of components within Apache Camel as agents with specific behaviours can lead to better-designed integration solutions. Studying their conduct helps developers create robust, efficient, and reliable systems.

Conclusion:

Apache Camel enables developers to create custom routing strategies and dynamically choose endpoints based on runtime conditions. It also offers robust error handling by incorporating AI and ML capabilities to define error-handling routes. Developers can enhance integration solutions by conceptualising components within Apache Camel as agents with specific behaviours and examining their collaboration within the flow.

Apache Camel utilises AI techniques to empower agents to make intelligent routing decisions based on real-time data and adjust their behaviour dynamically. These AI agents can work together to solve problems more efficiently than traditional systems, particularly in complex environments. One key advantage of AI in Apache Camel is predictive analytics, which facilitates proactive system management and optimisation. Additionally, the AI techniques used in Apache Camel are often abstracted from developers or programmers, allowing them to concentrate on building their application logic without concerning themselves with the underlying AI infrastructure.

4.2 AI-Driven Use Cases

As discussed in the previous chapter, tools like ChatGPT and processes such as ‘prompting’ allow developers to discuss a problem or task with AI by ‘chatting’ about it. The developer is integrating with AI through ChatGPT. ChatGPT uses ML algorithms to simplify and accelerate programming tasks.

Another way to utilise AI, in the scope of this thesis, is to use AI agents, which perform repeatable tasks in the background for the developer to save the developer time. Apache Camel leverages AI techniques to empower agents to make intelligent routing decisions based on live data and adjust their behaviour dynamically. These AI agents prove invaluable in lightening the developer's workload, allowing them to concentrate on programming tasks while the agents discreetly address issues in the background.

Use Case 1 describes how IBM utilised ChatGPT integration in Qurakus. The primary purpose of this use case is for developers to describe programming tasks in natural language, and ChatGPT (Quarkus-ChatGPT) assists them in generating the corresponding code snippets or solutions.

Use Case 2 is similar to Use Case 1; it also involves integrating ChatGPT into Quarkus (Quarkus-ChatGPT). For example, IBM and Red Hat utilise Quarkus-ChatGPT to comment on code. In this scenario, AI assists developers in programming by generating supportive comments based on their code. AI capabilities are also utilised to expedite creating and updating code documentation.

Use case 3 describes the CAMEL-AI framework, a research initiative that explores the potential of building scalable techniques to facilitate autonomous cooperation among communicative AI agents.

4.2.1 Use Case 1: Access the OpenAI ChatGPT API in Red Hat and Quarkus

The development tool Quarkus, created by Red Hat, provides a promising application of NLP. Quarkus-ChatGPT provides intelligent code suggestions, autocompletion, and error detection to assist developers in writing code more efficiently. Using a REST interface, Quarkus allows ChatGPT to analyse natural language descriptions of programming tasks and generate corresponding code snippets or solutions. This reduces the cognitive load on developers and enhances productivity.

Integrating Quarkus with ChatGPT (e.g. Quarkus-ChatGPT) has attracted the attention of prominent organisations such as IBM, Google, and Red Hat. IBM utilises the Quarkus-ChatGPT integration to develop cloud-native applications more efficiently, while Red Hat enhances the developer experience on its platform. Academic institutions such as MIT and Stanford also use the integration to teach programming languages to their students.

IBM is utilising the Quarkus-ChatGPT integration to develop cloud-native applications more efficiently. By leveraging the REST interface available for Quarkus, IBM's developers can describe programming tasks in natural language, and ChatGPT assists them in generating the corresponding code snippets or solutions. This has resulted in time savings and improved productivity for IBM's development teams.

Red Hat, on the other hand, is leveraging the Quarkus-ChatGPT integration to enhance the developer experience on its platform. ChatGPT's intelligent code suggestions, autocompletion, and error detection features help Red Hat's developers write code more efficiently and with fewer errors, leading to a more streamlined and productive development process for its teams.

Similarly, Google is leveraging ChatGPT to improve the developer experience on its platform. Using ChatGPT's natural language processing capabilities, Google's developers can more easily describe programming tasks and receive relevant code suggestions in real time.

IBM, Google and Red Hat are examples of organisations benefiting from the Quarkus-ChatGPT integration. By reducing developers' cognitive load and enhancing productivity, this integration has proven to be an effective tool for improving software development efficiency across various industries.

Below is an example of how the discussion with the ChatGPT through the integration might look; in this case, the integration over the REST API is prepared in GitHub (SW development platform). This example is from *A Software Deweloper's Guide to Generative AI*; Jason Beres owns the following examples. (Beres, 2023)

```
function countNumbersPoorPerforming(arr, target) {
  var count = 0;
  for (var i = 0; i < arr.length; i++) {
    if (arr[i] === target) {
      count++;
    }
  }
  return count;
}
```

Figure 3 Example: The developer's first coding attempt, the issue with the numbers in the array is known by the developer

```
If I have a very large array, is this the most optimized code to count numbers in the array?


function countNumbersPoorPerforming(arr, target) {
  var count = 0;
  for (var i = 0; i < arr.length; i++) {
    if (arr[i] === target) {
      count++;
    }
  }
  return count;
}
```

Figure 4 Example: The developer is asking ChatGPT for assistance with the array by prompting

```
function countNumbers(arr, target) {
  const countMap = {};

  for (let i = 0; i < arr.length; i++) {
    const num = arr[i];
    countMap[num] = (countMap[num] || 0) + 1;
  }

  return countMap[target] || 0;
}
```

Figure 5 Example: ChatGPTs solution on the array problem

However, it is essential to consider the potential shortcomings and risks of generating incorrect or inefficient code solutions when discussing the success of ChatGPT technology. Developers must ensure the accuracy of the generated code, and there is a risk that ChatGPT might not understand the full context of a programming task or requirement, leading to incomplete or inaccurate code suggestions. Despite these risks, the Quarkus-ChatGPT integration has proven to be an effective tool for improving software development efficiency across various industries.

Conclusions:

Quarkus can integrate with OpenAI ChatGPT, which allows it to generate code snippets from natural language descriptions of programming tasks. This integration reduces

cognitive demand and enhances productivity. IBM, Red Hat, Google, and academic institutions have benefited from this integration. However, it's important to note that there is a risk of generating incorrect or inefficient code solutions.

4.2.2 Use Case 2: Enhancing Code Documentation with Quarkus-ChatGPT Integration

The technical implementation and integration of Quarkus-ChatGPT are the same in Use Case 1 and Use Case 2.

Many programmers face challenges related to incomplete or outdated code comments and documentation. Developers spend much time searching for relevant documentation or understanding complex code segments.

Additionally, there might be struggles to ensure consistent and accurate documentation across the entire codebase. One solution for this problem is to utilise the Quarkus-ChatGPT integration, as mentioned in use case 1, instead for support in the coding process for code commenting and documentation purposes.

The same type of integration, the Quarkus-ChatGPT, as in use case 1, can result in numerous benefits, such as enhanced code documentation, reduced cognitive load for developers, improved productivity, and facilitation of community collaboration through open-source projects like the Quarkus-ChatGPT extension.

Christine Pinto owns the following two examples, which were released on the webpage *Writing Easier Code Comments with ChatGPT* in March 2024. (Pinto, 2024)

```
import { BrowserContext, Page, test as baseTest, chromium } from '@playwright/test';
import path from 'path';
import Landing from '../pages/landing';

export const test = baseTest.extend<{
  context: BrowserContext;
  extensionId: string;
  page: Page;
  landing: Landing;
}>({
  // parts of the setup for the persistent context from https://playwright.dev/docs/chrome-extensions#testing
  context: async ({}, use) => {
    const extPath = path.join(__dirname, '../../build');
```

Figure 6 Example: The original code snippet

After prompting ChatGPT: *Could you add good comments to explain the code to someone who hasn't built it?* The code commenting is generated as follows in Figure 7**.**

```
// Import necessary modules from Playwright for browser automation and testing,
// as well as the 'path' module for handling file paths, and the 'Landing' class
// from a local module, assumed to be a page object model representation of a la
import { BrowserContext, Page, test as baseTest, chromium } from '@playwright/te
import path from 'path';
import Landing from '../pages/landing';

// Extend the base test context provided by Playwright with additional propertie
// for browser context, extension ID, page, and a specific page object (landing
export const test = baseTest.extend<{
  context: BrowserContext;
  extensionId: string;
  page: Page;
  landing: Landing;
}>({
  // Async function to create and configure a browser context for testing a Chrc
  // This setup is part of testing Chrome extensions as per Playwright's documer
  context: async ({}, use) => {
    // Resolve the path to the extension's build directory.
```

Figure 7 Code after ChatGPT has generated appropriate comments for the code

Integrating ChatGPT with Quarkus presents challenges like natural language understanding, safety, and model bias. To enhance code documentation successfully, it is imperative to ensure responsible usage and continuous improvement.

Conclusion:

The Quarkus-ChatGPT integration offers a promising solution by enhancing code documentation and commenting, reducing cognitive load, boosting productivity, and fostering community collaboration. However, integrating ChatGPT with Quarkus presents challenges, including natural language understanding and model bias. Responsible usage and continuous improvement are crucial to successfully harnessing AI's benefits in code documentation while mitigating risks.

4.2.3 Use Case 3: AI and ML Capabilities for Apache CAMEL Through CAMEL-AI

As discussed in the Theoretical Framework section, Apache Camel is an open-source integration framework that facilitates seamless data exchange between different systems, applications, and protocols. CAMEL-AI is a ‘sub-category’ or research area with potential that can be useful in the integration development process.

CAMEL-AI framework is a *research initiative* exploring the potential of building scalable techniques to facilitate autonomous cooperation among communicative agents. The framework provides insights into these agents' "cognitive" processes, which can help researchers develop more effective communication models.

One of the critical features of the CAMEL-AI framework is the introduction of a novel communicative agent framework called "role-playing." This framework guides chat agents towards task completion while ensuring consistency with human intentions. The role-playing framework also enables the creation of more complex conversational structures, which can be used to generate more accurate and nuanced chatbot responses.

In addition to facilitating the development of more effective communication models, the CAMEL-AI framework also generates conversational data that can be used to study the behaviours and capabilities of chat agents. This data can be used to investigate various aspects of conversational language models, such as their ability to understand natural language input, generate appropriate responses, and maintain context over time.

Developers who work with the CAMEL-AI framework define agent behaviours, communication protocols, and decision-making rules. They can integrate existing AI models (for example, NLP, reasoning, and planning) into the framework. CAMEL-AI can be used for debugging, monitoring agent interactions, ensuring proper coordination, and addressing conflicts. The framework provides tools for visualising agent interactions, analysing logs, and improving system performance.

Error handling is crucial in any integration scenario. Agents responsible for handling errors could play a critical role. Some concrete examples where developers could utilise CAMEL-AI are error-handling routes (or agents) within the Apache Camel framework. Another exciting area is integration flows, which often involve multiple components working together. Each element in the integration flow (e.g., data transformation, validation,

enrichment) can be seen as an agent. Developers study how these agents collaborate within the flow. Finally, monitoring agents can be built to collect data about system performance, message throughput, and resource utilisation. Developers configure monitoring agents (e.g., JMX, Prometheus) in Apache Camel. These agents provide insights into system behaviour.

Overall, the CAMEL-AI framework offers a powerful tool for programmers interested in developing more advanced, autonomous communicative agents. Its combination of advanced communication frameworks and sophisticated data generation capabilities make it a valuable resource for anyone interested in exploring the potential of AI-powered agents.

Conclusions:

CAMEL-AI is designed to enable autonomous cooperation among communicative agents in the Apache Camel integration framework. The framework provides effective communication models by offering insights into agents' cognitive processes and generating conversational data. Developers can integrate existing AI models into the framework and use it for error handling, monitoring, and coordination. As the demand for AI-powered agents continues to rise, CAMEL-AI emerges as a valuable tool, driving innovation and efficiency in integration tasks across diverse domains. The CAMEL-AI framework offers a powerful tool for developing advanced, autonomous communicative agents, which can support the development process of integration tasks.

4.3 Best practices based on the use cases

The client is requesting a thorough analysis of the current best practices regarding AI and ML as of spring 2024 as part of the final assignment for this research project. This chapter will gather best practice recommendations based on the research.

4.3.1 ChatGPT and ‘prompting’

Use natural language descriptions: When using ChatGPT, it's best to provide natural language descriptions of programming tasks. This will help the system generate more accurate code suggestions.

Clearly define the programming challenge or problem. For example, before interacting with ChatGPT, ensure a clear understanding of the coding problem. The best prompting results will be received if the problem statement is clearly defined.

Providing context is vital for ChatGPT to generate accurate code suggestions. By providing context, the system can understand what the developer is trying to achieve and provide relevant recommendations.

Continuously improve the system: ChatGPT is an AI-powered system that learns over time. By constantly providing feedback and improving the system, developers can get more accurate and relevant code suggestions.

ChatGPT is a collaborative tool that offers creative suggestions and insights to help solve problems and generate new ideas. Always remain open to alternative solutions and view ChatGPT as a co-pilot or assistant. While ChatGPT is intelligent, it is still subject to errors and limitations.

If an inaccurate response is received, reformulate the prompt instead of directly contradicting ChatGPT. This approach preserves the context of the conversation. By implementing this strategy, the user can ensure that they receive an appropriate and accurate response from ChatGPT without disrupting the flow of the conversation.

Conclusion:

The programming challenge or problem should be clearly defined to optimise the results obtained from ChatGPT. Contextual information is crucial for accurate code suggestions. The system's effectiveness improves with continuous feedback and enhancement. ChatGPT serves as a collaborative tool, offering creative insights. Reformulating prompts rather than directly contradicting ChatGPT preserves conversation context, ensuring accurate responses without disruption. Use ChatGPT as a tool, not a replacement: ChatGPT helps developers write code more efficiently. It's important to remember that it's not a replacement for human expertise and judgment.

4.3.2 AI-agents

Utilise machine learning algorithms to empower agents to make intelligent routing decisions based on real-time data and context, thereby enhancing the efficiency and effectiveness of message routing.

Implement advanced AI techniques to empower agents to autonomously adjust their behaviour in response to evolving conditions or requirements without manual intervention. This capability ensures that the system continuously optimises its performance based on real-time changes, enhancing efficiency and effectiveness.

Predictive models can forecast resource usage, identify bottlenecks, or proactively predict system failures. This facilitates proactive system management and optimisation.

Design AI agents to operate autonomously, communicating and collaborating to achieve objectives without requiring constant guidance or input from programmers.

Conclusion:

In conclusion, integrating ML algorithms and advanced AI techniques empowers agents within Apache Camel to operate intelligently and autonomously. By utilising these technologies, agents can make informed routing decisions based on real-time data and context, leading to enhanced efficiency and effectiveness in message routing. Furthermore, the ability of AI agents to autonomously adjust their behaviour in response to evolving conditions ensures continuous optimisation of system performance without manual intervention. Predictive models enhance system management by proactively identifying resource usage, bottlenecks, and potential failures. Designing AI agents to operate autonomously fosters seamless communication and collaboration, driving towards achieving system objectives without requiring constant guidance from programmers.

4.4 Impact on business models

Industry 4.0, as briefly outlined in the introduction, signifies a revolutionary era in business characterised by the influence and adaptations catalysed by advanced AI and ML technologies. Numerous companies have long been anticipating and adapting to this monumental shift. As highlighted within the theoretical framework, innovative businesses

such as Airbnb, Uber, and Amazon have successfully utilised AI to enhance their operations through dynamic pricing, personalised services, route optimisation, and supply chain efficiency. These trailblazing companies have dramatically transformed the hospitality, transportation, and retail sectors. Nevertheless, while these examples are impressive, they may not fully meet the requirements of IT systems and technology providers.

4.4.1 Industry 4.0 and business models

Studies have also delved into the impact of AI on companies' business models, with an overwhelming consensus that data-driven decision-making lies at the heart of AI-based models. In contrast, traditional models rely on human judgment and historical data. By leveraging sophisticated algorithms to rapidly and precisely analyse vast data, AI models enable businesses to tailor their services to individual customers' unique preferences and behaviours. The level of personalisation these models offer is often beyond the capabilities of traditional models.

AI allows businesses to provide personalised products, services, and experiences based on customer preferences and behaviour. This level of customisation is often unattainable with traditional business models. AI-powered business models automate repetitive tasks and processes, increasing efficiency and productivity. Traditional models may require more manual intervention and time-consuming processes. With AI, businesses can predict future trends, customer behaviour, and market changes with a high degree of accuracy. AI enables enterprises to innovate and gain a competitive edge quickly through cutting-edge technologies. Traditional models may struggle to keep up with technological advancements and innovation. (Mishra & Tripathi, 2021, pp. 5–11)

In their January 2022 publication, Åström, Reim, and Parida discussed aligning value creation and capture, focusing on how value is generated for the customer. AI facilitates the creation of additional value for customers by improving operational efficiency and accelerating innovation. The potential of AI is to improve cost efficiency and, in the end, revenue for end customers. (Åström et al., 2022, pp. 2111–2112)

The integrations themselves create unique value for customers as they reduce manual work, including, for example, time-consuming error handling. As the development can be performed with support from AI, the end customers capture value through lower pricing

or better service agreements. Innovative business models can differentiate offerings. Scalability can more easily be considered, e.g. how well prepared the development processes are for handling increased data volumes or additional customers.

Reliable sources such as Forbes discuss AI's advantages, especially in areas that enhance decision-making processes and strategic planning. These technologies can analyse vast amounts of data in real-time, extracting insights that human analysis might overlook. Strategic planning becomes more accurate, adaptive, and efficient, allowing businesses to pivot and seize opportunities quickly. (Grant, 2023)

4.4.2 Examples of business models

Several examples of business model approaches where AI is involved as a part of the business have been investigated for instance, these examples:

Integrated Digital IT Operating Model:

- Impact: This model streamlines processes, enhances collaboration, and ensures a unified view of technology capabilities.
- Business Model Change: Companies shift from separate digital and IT units to an integrated approach, organising around technology capabilities rather than specific assets or functions.

Networked Business Models for Sustainability Transitions:

- Impact: Co-developing networked business models integrates systemic change toward sustainability.
- Business Model Change: Companies adapt their models to address sustainability challenges, emphasising collaboration and shared value creation.

Amazon’s Integration of Digital Business Models:

- Impact: Amazon seamlessly integrates IoT technologies, sensors, and AI for customer profiles and contactless payments.
- Business Model Change: Amazon’s business model leverages digital tools to enhance customer experiences, drive growth, and transform retail.

4.4.3 Handling return of investments

The integration of ChatGPT functionality and AI agents into development tools has the potential to save money and increase effectiveness. However, calculating return on investment (ROI) when AI assists developers daily is still challenging. As the understanding of AI's impact on development processes evolves and tools for measuring ROI improve, more precise insights into the tangible returns are anticipated to be gained. The focus remains on using AI to streamline development workflows, enhance productivity, and drive innovation within our organisation.

Insights from reputable sources such as Gartner are drawn upon, recognising Gartner as a widely respected authority in the field. Gartner investigated assessing AI ROI, including measuring value and cost, quick wins, use cases and transformative initiatives. Even if Gartner does not provide examples from a broader perspective, insights are provided in area strategic guidance for effectively using AI for business transformation and competitive edge. (Wiles, 2023)

Evaluating technical and business metrics when assessing the ROI of AI initiatives is essential. Gartner emphasises that AI-mature organisations tend to focus on combining these metrics. Also, customer success-related business metrics are usually incorporated. Technical metrics are often used to measure the value of AI use cases for less AI-mature organisations. (Stamford, 2023)

In conclusion, based on Gartner's research, AI has the potential to improve productivity and transform businesses significantly. To evaluate investments in AI, it is recommended to simulate potential costs and value realisation across various AI activities. Prioritise AI initiatives based on their impact, cost, and complexity and fund them with cost savings and productivity gains. Executives should consider strategic, competitive, and market-level impacts for transformational AI initiatives.

As AI technology has progressed, organisations have been reaping the benefits of integrating it into their operations. However, decision-makers are understandably cautious about investing in AI due to the potential for substantial costs and disruptions, with no guaranteed return on investment. The main objective of incorporating AI is to improve operational efficiency, with predictive, interpretive, and AI all offering unique advantages. Forbes also points out that while the average enterprise sees a return on their AI

investment within 16 months, it is worth noting that approximately 30% of organisations do not see any returns. In these cases, Forbes advises reassessing the intended use cases and goals. (Mahurkar, 2024)

Businesses can reap benefits from predictive AI, including increased revenue growth through improved forecasting, optimised pricing, and reduced customer attrition. In addition, predictive AI can aid in reducing operational costs and enhancing overall efficiency by providing valuable insights from data. Interpretive AI, a subset of predictive AI, mitigates potential risks from documents and images. Integrating predictive AI within business operations can improve efficiency and effectiveness in processes. Overall, these applications of predictive AI present a valuable opportunity for businesses to achieve a positive return on investment while considering the implementation of AI technologies. (Mahurkar, 2024)

According to Gartner and Forbes, the general payback time for investing in AI is estimated to be 1-2 years. However, since AI is a broad concept, the payback time must be calculated and analysed in each use case. It is essential to evaluate the specific applications of AI in each use case, using examples such as the use cases in this thesis that show how AI has been successfully implemented.

4.5 Potential future research area: Code analysis tool (GRAFANA)

AI-powered tools for code analysis have become increasingly popular in recent years. These tools use advanced AI techniques like natural language processing, machine learning and static code analysis to detect issues such as bugs, security vulnerabilities and code quality concerns in seconds. They can also identify performance bottlenecks by analysing code patterns and providing customised recommendations to enhance code quality and maintainability. These tools have many functions, from detecting and resolving bugs to enforcing coding standards and optimising code performance. In addition, these tools can improve software's dependability, quality and maintainability. They provide developers with valuable insights and practical recommendations for improving their code, streamlining workflows and enhancing productivity.

For instance, AI-powered code analysis tools can provide developers with a detailed report highlighting the most critical issues in their codebase and suggesting how to fix them. By

automating the code review process, these tools can help developers save time, reduce errors, and improve the overall quality of their code.

AI-powered code analysis tools are an essential part of modern software development workflows. They can help developers produce high-quality code, reduce technical debt, and improve time to market, making them invaluable assets for any software development team. Although the possibilities of AI-powered tools within GRAFANA are only briefly listed below and shall be considered out-of-scope for this thesis, they could be potential areas for future research.

Monitoring of performance

ML models can predict system failures and performance issues in OpenShift environments by analysing logs, metrics, and historical data. This can help organisations schedule maintenance tasks, allocate resources efficiently, and minimise downtime.

Automated Testing

AI can automate testing processes by generating cases, predicting potential failure points, and prioritising test scenarios. NLP techniques can analyse testing requirements and create test plans or documentation.

Resource optimisation

AI algorithms can optimise resource allocation in Red Hat OpenShift clusters based on workload demands, performance metrics, and cost considerations, improving system efficiency while reducing infrastructure costs.

Monitoring of security

AI-powered security analysis tools can detect security threats and anomalies in Red Hat OpenShift environments by analysing system logs, network traffic, and application behaviour. ML models can learn standard behaviour patterns and identify deviations that may indicate malicious activities, helping organisations mitigate security risks effectively.

In conclusion, Apache Camel AI agents and AI-powered tools in Grafana utilise artificial intelligence techniques. However, they operate within different domains and serve various purposes. Apache Camel AI agents enhance integration capabilities, while AI-powered tools

in Grafana focus on improving the monitoring and observability of systems and applications.

5 Discussion

The transformative potential of AI tools, such as ChatGPT, has been recognised in this research. At first, their capabilities were underestimated due to limited familiarity and expertise in AI. However, after a comprehensive exploration of background literature, theoretical frameworks, and various real-world and hypothetical use cases, the profound impact of ChatGPT-type solutions has been elucidated. The examples presented in the theoretical framework and literature review demonstrate the considerable benefits that ChatGPT-type tools and AI agents can bring to support developers.

Did this research identify the appropriate use cases when the scope was Red Hat Openshift development tools, such as Quarkus and Apache Camel? The research questions, as stated at the beginning of the research, are as follows:

- ***What AI-driven techniques*** are best suited for integration platforms like Red Hat OpenShift, and how can they support the company's operations and objectives?
- ***To what extent*** does the use of AI affect the amount of work in development processes compared to traditional methods?
- ***How will the adoption*** of AI in programming tasks impact the company's existing business models, and what benefits and challenges are expected in specific areas of developing the integration platform?

These aspects will be discussed in the following sections of this research.

5.1 The potential of AI in integration platforms

An effective incorporation of AI into development processes necessitates meticulous planning, requisite expertise, and comprehensive understanding. Pilot testing and proof-of-concept methodologies emerge as reliable and prudent strategies when introducing novel technologies, such as AI. The requirement for Pilots, Proofs-of-concept and an overall need for an increase in the level of knowledge when it comes to AI can be observed, both

based on the knowledge gained from the literature review as well as from the internal discussions held regarding AI and its potential and possibilities, within the client organisation over the past six months.

During the early stages of discussions with the client, it was evident that identifying a use case that would allow technically proficient individuals with limited programming backgrounds to execute complex programming tasks on the integration platform was of interest.

By utilising low-code components available on Quarkus and Apache Camel, along with ChatGPT as a programming co-pilot within the Quarkus environment, individuals with less programming experience could potentially participate in programming tasks more effectively. This can be achieved by a blend of intuitive visual interfaces provided by low-code tools in Quarkus, along with the assistance of ChatGPT in generating code snippets or giving guidance. This could lower the barrier to entry for less experienced developers, enabling them to contribute meaningfully to development projects.

Although AI agents are not overtly apparent tools for developers, such as ChatGPT's copilot or assistant, they can effectively manage mundane and repetitive tasks in the developer's background, leveraging their AI capabilities. AI utilised in these agents can also make decisions autonomously, which can be particularly useful for developers working on clearly defined tasks and functions. This reduces the cognitive load on the developer and minimises the risk of human error, which can be costly in terms of time and resources.

Additionally, AI agents can learn from past decisions and improve their performance over time, making them an asset for developers looking to optimise their workflows and increase productivity. By relying on AI to handle more routine tasks, developers can focus on more complex and creative work, ultimately leading to better outcomes and more innovative solutions.

Conclusion:

In conclusion, integrating AI into development processes requires careful planning, expertise, and understanding. Pilot testing and proof-of-concept approaches are essential strategies for introducing AI technologies effectively. By identifying appropriate use cases, such as those within Red Hat Openshift development tools like Quarkus and Apache Camel,

and leveraging low-code components alongside ChatGPT as a programming co-pilot, resources with less programming experience can contribute meaningfully to development projects.

AI agents can effectively manage repetitive tasks, reducing the cognitive burden on developers and minimising the risk of human error. Overall, AI agents enhance productivity and facilitate innovation by allowing developers to concentrate on challenging tasks while optimising workflows and improving performance over time.

5.2 Impact on the business model

Based on the extensive exploration of AI's impact on business models, Industry 4.0 represents a significant shift characterised by the pervasive influence of advanced AI and ML technologies. Businesses across various sectors have embraced this transformation, leveraging AI to enhance operations, drive innovation, and deliver personalised experiences to customers. Adopting AI-powered business models enables organisations to streamline processes, increase efficiency, and gain valuable insights from data.

Several studies have underscored the importance of data-driven decision-making in AI-based business models, highlighting the role of AI in tailoring services to individual customer preferences and behaviours. The level of personalisation offered by AI models surpasses traditional approaches, leading to enhanced customer satisfaction and loyalty.

Furthermore, AI-powered business models automate repetitive tasks, freeing up resources and enabling organisations to focus on strategic initiatives. Predictive analytics allows businesses to anticipate trends, optimise resource allocation, and identify potential risks or opportunities. Examples from industry leaders such as Amazon demonstrate how the integration of AI technologies can revolutionise business operations and enhance customer experiences.

Despite the potential benefits, evaluating AI initiatives' return on investment (ROI) remains challenging for many organisations. However, insights from reputable sources like Gartner and Forbes provide valuable guidance on assessing AI ROI, prioritising initiatives, and measuring value realisation. Organisations can make informed decisions and strategically allocate resources for AI implementation by simulating potential costs and benefits.

Overall, while AI presents opportunities for significant productivity gains and business transformation, organisations must carefully evaluate AI investments' specific applications and potential returns. By embracing AI technologies strategically and leveraging insights from industry experts, businesses can unlock new opportunities for growth and innovation in the rapidly evolving digital landscape.

5.3 AI and governance

Managing Risks:

Through machine learning, AI helps businesses understand customer and employee behaviour, perform tasks, support specialists in many fields, and strengthen security by detecting bugs and cyberattack risks. However, every technology, AI included, has downsides and challenges. Regarding risk, the sources for this research are Gartnes and Forbes.

When AI is involved, risks can include bias and fairness in AI algorithms, data privacy and security concerns, the potential for job displacement due to automation, and the ethical implications of AI decision-making. (Weizman, 2023)

Bias and fairness are concerns in AI, as algorithms can perpetuate or even exacerbate existing biases in the data they are trained on. This can lead to unfair treatment or discrimination against certain groups. Ensuring fairness and mitigating bias in AI systems is crucial for building trust and promoting equitable outcomes. One specific example of bias in AI, which is not connected to programming tasks or integration platforms, involves facial recognition technology. Facial recognition algorithms have been found to exhibit racial bias, often misidentifying individuals with darker skin tones more frequently than those with lighter skin tones. This bias can lead to discriminatory outcomes, such as increased surveillance and targeting of minority groups by law enforcement agencies. Additionally, biased facial recognition systems can affect access to services like banking or transportation, as well as employment opportunities, further perpetuating systemic inequalities. This highlights the importance of addressing bias in AI algorithms to ensure fair and equitable outcomes for all individuals.

Using AI for coding can lead to expensive mistakes, harming a company's infrastructure and reputation. AI systems are vulnerable to cyberattacks, posing risks of financial losses and security breaches. When AI is used in automated decision-making, and its speed is rapid, there is a risk of multiple erroneous chained decisions being made. AI may also struggle to comprehend human emotions, potentially resulting in insensitive interactions and customer loss.

To help businesses handle the risks and challenges that AI brings, IBM, along with Gartner and Forbes, has developed best practices and conclusions for managers in different types of businesses. (IBM, 2024a)

The common factors for handling and mitigation risks that these prominent influences suggest are:

- Controlling by governance (for example, GDPR)
- Models of transparency and documentation
- Robust AI Model validation
- Bias mitigation by regular audition

Conclusion:

In conclusion, while AI has the potential to benefit businesses in many ways, there are also risks and challenges to be considered. These include issues related to bias and fairness, data privacy and security, job displacement, and ethical implications. To mitigate these risks, it is crucial to focus on governance, transparency, validation, and bias mitigation through regular auditions. By addressing these concerns, businesses can harness the power of AI while promoting equitable outcomes and building trust.

Ethical considerations:

The ethical approach to AI development involves several considerations. One crucial aspect is incorporating diverse data sets, which can help reduce bias and ensure fairness in AI models. Ethical considerations are important even in AI integration within integration platforms and programming. One key aspect involves the careful selection and utilisation

of diverse datasets. By ensuring that the AI algorithms within the platform are trained on inclusive and representative datasets, potential biases can be mitigated, promoting fairness and equitable outcomes. A proactive approach aligns with ethical principles and enhances the integrity and reliability of AI-powered features within the platform. Organisations can prevent biased outcomes and ensure fair treatment for all individuals by utilising a wide array of data sets.

Another important consideration is the inclusion of ethicists in AI project teams. Ethicists can provide unique insights and perspectives on ethical considerations during AI development, helping to identify potential biases, assess risks, and promote responsible AI practices.

Regular audits should be conducted to ensure the ethical integrity of AI systems. These audits help identify unintended consequences or biases that may have emerged, and organisations can take corrective action accordingly.

Transparency and explainability are also critical components of ethical AI development. AI systems must be transparent and comprehensible, and companies should strive to explain how decisions are made, particularly in vital areas such as lending, hiring, or healthcare.

The organisation must also clearly define accountability and responsibility for AI development, deployment, and monitoring. This ensures that ethical considerations are taken seriously and the organisation is held accountable for any ethical lapses.

Privacy protection is another crucial consideration in ethical AI development. Companies must prioritise user privacy, and AI systems should handle personal data securely while complying with privacy regulations such as GDPR.

While AI has the potential to create new jobs and improve productivity, there is also a concern that it may take over tasks traditionally performed by humans. Organisations must consider the impact of AI on the workforce and work to ensure that any job displacement is minimised while also ensuring that workers are trained in new skills to adapt to the changing job market.

Conclusion:

Ethical AI development demands a comprehensive and responsible approach. Key considerations include incorporating diverse data sets, including ethicists in project teams, ensuring transparency and accountability, protecting user privacy, and considering the impact on the workforce. By implementing responsible AI practices and ongoing education, organisations can promote equitable treatment for all individuals and minimise any negative impact.

AI and legal domains:

The pervasive influence of technology on daily activities and societal norms has ushered in a new era of challenges and opportunities for legal professionals. From data privacy concerns and e-commerce regulations to the complexities of AI and cybersecurity threats, legal practitioners must navigate a rapidly evolving landscape to address emerging legal issues effectively. Adapting to these technological shifts is essential for legal professionals to uphold legal standards, protect individual rights, and foster innovation in a digital age. The intersection of law and technology has given rise to various complex issues that require careful consideration. Emerging themes and concerns in this field include artificial intelligence, cryptocurrency, and data protection. (Brożek et al., 2023)

Ethical considerations are among the critical issues in the development of artificial intelligence. These include addressing bias, accountability, and transparency in AI development. Additionally, it is crucial to develop legal frameworks to govern AI applications in areas like healthcare, finance, and autonomous vehicles.

Organisations must adhere to regulations like the GDPR to safeguard personal data in data protection. In addition, establishing protocols for responding to data breaches and mitigating risks to individuals' privacy and security is critical.

Finally, cybersecurity is another critical issue that must be addressed. This includes avoiding evolving cyber threats, such as ransomware attacks and phishing schemes, to protect critical infrastructure and sensitive information. Determining legal liability in cybersecurity breaches and data theft cases is also a key concern, including the responsibilities of organisations and individuals.

Digital rights are also a growing concern. This includes balancing freedom of expression with regulating harmful content and misinformation on digital platforms. Addressing

concerns about government surveillance, data collection practices, and protecting civil liberties in the digital age is also critical.

Conclusion:

Integrating technology and law presents legal professionals with various challenges and opportunities. From navigating AI ethics and data protection regulations to combating cybersecurity threats and safeguarding digital rights, practitioners must adapt to an ever-evolving landscape. Key concerns include addressing AI bias, upholding data privacy standards, and ensuring legal liability in cyber breaches. Balancing regulatory oversight with individual freedoms remains crucial in fostering innovation while protecting civil liberties in the digital age.

6 Conclusions

The present research investigates the real-world implications of integrating advanced AI-powered tools, including ChatGPT and AI agents, into the development processes of integration platforms designed for Red Hat Openshift.

Initially, the capabilities of NLP or ChatGPT type of AI were underestimated due to the researcher's limited familiarity and expertise. However, a comprehensive exploration of the background literature and real-world use cases helped overcome this underestimation. The use cases presented in this research demonstrate the potential benefits of integrating AI technology into the development processes of the Red Hat platform, including reduced costs, modified business models, increased efficiency, and new revenue streams.

The research suggests that the effective incorporation of AI into development processes is a joint effort, requiring meticulous planning, requisite expertise, and a comprehensive understanding of the technology from developers. Pilot testing and proof-of-concept methodologies are reliable and prudent strategies for introducing novel technologies like AI. By identifying appropriate use cases and leveraging low-code components alongside AI-powered tools, resources with less programming experience can contribute meaningfully to development projects. Furthermore, the study highlights how AI agents, such as ChatGPT's copilot or assistant, can effectively manage mundane and repetitive tasks, making the work of programmers more efficient and effective.

Integrating AI-powered tools into development processes can transform how developers work, leading to better outcomes and more innovative solutions. For instance, AI-powered tools can automate repetitive and time-consuming tasks, freeing developers to focus on more complex and creative tasks. AI can also analyse large volumes of data and provide developers with insights that they might not have been able to obtain otherwise. Additionally, AI-powered tools can help developers identify coding errors and security vulnerabilities, improving the quality and security of their code.

Research supports the idea that integrating AI-powered tools into development processes can lead to benefits. Studies have found that AI can help improve operational efficiency, reduce costs, enhance decision-making processes, and identify new opportunities. By automating repetitive and time-consuming tasks, AI can free developers to focus on more complex and creative tasks. Additionally, AI-powered tools can help businesses improve the quality and security of their code.

However, certain limitations and weaknesses exist when integrating AI into development processes. Firstly, AI-powered tools may not be suitable for all programming tasks, especially those requiring high human expertise and creativity. AI may not replicate the human-driven creative process often required for complex and creative programming tasks. Secondly, AI-powered tools may have data privacy and security limitations, requiring developers to be aware of the potential risks associated with using AI-powered tools and take appropriate measures to ensure the security of their data and systems.

Thirdly, using AI-powered tools may lead to over-reliance on technology, limiting the development of human expertise and creativity. Hence, developers must balance using AI-powered tools and developing their skills and expertise. Management and developers must also be aware of the limitations of AI-powered tools and use them effectively and responsibly.

Finally, developing and implementing AI-powered tools can be time-consuming and costly. The client's management must be willing to invest the necessary time and resources to integrate AI into the development processes effectively. Training developers to use and integrate AI-powered tools effectively can also be a remarkable investment.

In conclusion, while there are certain limitations and weaknesses to consider when utilising AI for programming purposes, the benefits of integrating AI into the development

processes of the Red Hat OpenShift integration platform cannot be ignored. A balanced integration of AI-powered tools into development processes can transform the way developers work, leading to better outcomes and more innovative solutions. Developers must be aware of the limitations of AI-powered tools and take appropriate measures to use them effectively and responsibly. Integrating AI into development processes is a continuous process that requires investment, expertise, and a comprehensive understanding of the technology.

7 Future pathways based on the research

The thesis explores the integration of AI into development processes, focusing on its potential impact on Red Hat's operations and objectives. It discusses how AI integration can reduce development costs, its economic impact on the organisation, and the expected benefits and challenges in developing integration platforms. The text also highlights the transformative potential of AI tools like ChatGPT and AI agents in supporting developers and reducing the cognitive burden on them. Additionally, it emphasises the potential of AI in business models, referencing a paper on AI's impact on business approaches.

The research paper, conducted in the spring of 2024, aimed to understand better AI and ML techniques in the chosen development frameworks. The extensive literature review focused on the technology's theoretical framework. The study intended to navigate the evolving waters of AI and ML and extract relevant and reliable information to meet the research objectives.

The analysis examined the concepts, theories, and frameworks underpinning AI and ML and the advantages and limitations of different techniques and methods—the research aimed to deepen the understanding of AI and ML and its potential use in integration platforms. The literature review was crucial in setting the foundation for subsequent study and enabling a better comprehension of the research problem and context.

As discussed, integrating AI-powered tools into development workflows can potentially transform software development profoundly. To ensure effective integration and mitigate risks, a strategic approach is critical. This involves pilot testing, providing training and support, monitoring performance, addressing privacy and security concerns, promoting responsible use of AI, and staying informed. In conclusion, a few possible approaches to be

considered are listed below. The first three specifically target the development and programming process. The last ones are more general and target the organisation more broadly.

Conduct POC and Pilot Testing:

Initiate pilot or proof-of-concept (POC) methodologies to assess the feasibility and effectiveness of integrating AI-powered tools into development workflows. Gather feedback from developers and stakeholders to evaluate the impact on programming tasks and overall project outcomes. Discuss with other companies in the field to understand how they have built up their development with AI in programming processes.

Provide Training and Support:

Offer training programs and support resources to developers to familiarise them with AI-powered tools and help them leverage these tools effectively in their day-to-day work. Ensure developers can access guidance and assistance as they integrate AI into their development processes.

Monitor Performance and Impact:

Continuously monitor the performance and impact of AI-powered tools on development processes, including metrics such as cost reduction, efficiency gains, and quality improvement. Use feedback and data analytics to refine strategies and optimise the integration of AI into development workflows.

Address Privacy and Security Concerns:

Implement measures to address data privacy and security concerns using AI-powered tools. Develop protocols and policies to safeguard sensitive data and mitigate risks related to data breaches and security vulnerabilities.

Promote Responsible Use of AI:

Promote a culture of responsible AI usage among developers and management, emphasising the importance of ethical considerations, transparency, and accountability.

Encourage collaboration and knowledge sharing to ensure the organisation uses AI effectively and responsibly.

Stay Informed and Adapt:

Stay informed about advancements in AI technology and industry best practices, and be prepared to adapt strategies and approaches as the technology landscape evolves. Continuously seek opportunities to innovate and optimise development processes by strategically integrating AI-powered tools.

8 REFERENCES

2004-2024 The Apache Software Foundation. (2024). *What Is Camel?*

Antila Mirva. (2023). *DigiFinland Yleisesittely*.

Åström, J., Reim, W., & Parida, V. (2022). Value creation and value capture for AI business model innovation: a three-phase process framework. *Review of Managerial Science*, *16*(7), 2111–2133. https://doi.org/10.1007/s11846-022-00521-z

Beattie, T., Hepburn, M., O'Connor, N., Spring, D., & Doria, I. (2021). DevOps Culture and Practice with OpenShift. In *https://www.packtpub.com/product/devops-culture-and-practice-with-openshift/9781800202368*.

Beres, J. (2023, June 13). *A Software Developer's Guide to Generative AI*. https://builtin.com/software-engineering-perspectives/generative-ai-tips-for-software-development

Biswas, S. (2023). Role of ChatGPT in Computer Programming. *Mesopotamian Journal of Computer Science*, 8–16. https://doi.org/10.58496/mjcsc/2023/002

Black, A. P. (2013). Object-oriented programming: Some history, and challenges for the next fifty years. *Information and Computation*, *231*, 3–20. https://doi.org/10.1016/j.ic.2013.08.002

Boering, I. (2021, January 11). *A beginners guide to Open Source Innovation*. Applying Open Source to the Enterprise.

Bogdan M. Wilamowski, & J. David Irwin. (2018). Intelligent Systems. In *https://books.google.fi/books?id=fn50DwAAQBAJ*.

Brożek, B., Kanevskaia, O., & Pałka, P. (2023). *Research handbook on law and technology*.

Bueno, A. S. (2023, October 24). *Access the OpenAI ChatGPT API in Quarkus*. Access the OpenAI ChatGPT API in Quarkus. https://developers.redhat.com/articles/2023/10/24/access-openai-chatgpt-api-quarkus#

Cisco. (2019). *Cisco UCS Infrastructure for AI and Machine Learning with Red Hat OpenShift Container Platform 3.11*.

Dhruv, M. (2023, December 23). *Navigating the Integration Landscape: iPaaS Use Cases, Trends, and the Transformative Power of AI and ML*. Navigating the Integration Landscape: IPaaS Use Cases, Trends, and the Transformative Power of AI and ML.

Ebert, N., Weber, K., & Koruna, S. (2017). Integration Platform as a Service. *Business and Information Systems Engineering*, *59*(5), 375–379. https://doi.org/10.1007/s12599-017-0486-0

Goasduff, L. (2016, March 1). *Getting Ready for Industrie 4.0*. https://www.gartner.com/smarterwithgartner/getting-ready-for-industrie-4-0

Grant, D. (2023, September 12). *Harnessing AI And ChatGPT Technology: The Next Industrial Revolution*. Harnessing AI And ChatGPT Technology: The Next Industrial Revolution.

Hauck, M., Machhamer, R., Czenkusch, L., Gollmer, K. U., & Dartmann, G. (2019). Node and Block-Based Development Tools for Distributed Systems with AI Applications. *IEEE Access*, *7*, 143109–143119. https://doi.org/10.1109/ACCESS.2019.2940113

Hinton, G., LeCun, Y., & Bengio, Y. (2015). Review Deep learning. *Deep Learning*, 435–444.

IBM. (2024a). *IMB - Manage AI risk and reputation*.

IBM. (2024b). *What is Industry 4.0?* https://www.ibm.com/topics/industry-4-0

Kulkarni, R. H., & Padmanabham, P. (2017). Integration of artificial intelligence activities in software development processes and measuring effectiveness of integration. *IET Software*, *11*(1), 18–26. https://doi.org/10.1049/iet-sen.2016.0095

Lee, J., Suh, T., Roy, D., & Baucus, M. (2019). Emerging technology and business model innovation: The case of artificial intelligence. *Journal of Open Innovation: Technology, Market, and Complexity*, *5*(3). https://doi.org/10.3390/joitmc5030044

Li, G., Hammoud, H. A. A. K., Itani, H., Khizbullin, D., & Ghanem, B. (2023). *CAMEL: Communicative Agents for "Mind" Exploration of Large Language Model Society*. http://arxiv.org/abs/2303.17760

Linthicum David S. (2003). Enterprise Application Integration David S. Linthicum 2003. *Enterprise Application Integration*.

Macdonald, C., Adeloye, D., Sheikh, A., & Rudan, I. (2023). Can ChatGPT draft a research article? An example of population-level vaccine effectiveness analysis. *Journal of Global Health*, *13*. https://doi.org/10.7189/JOGH.13.01003

Mahurkar, A. (2024, April 10). *The Elusive Case Of ROI In AI Projects*. Https://Www.Forbes.Com/Sites/Forbestechcouncil/2024/04/10/the-Elusive-Case-of-Roi-in-Ai-Projects/?Sh=79f20396113f.

Mishra, S., & Tripathi, A. R. (2021). AI business model: an integrative business approach. *Journal of Innovation and Entrepreneurship*, *10*(1). https://doi.org/10.1186/s13731-021-00157-5

Oh, D. (2021, August 12). *A Java developer's guide to Quarkus*. Https://Opensource.Com/Article/21/8/Java-Quarkus-Ebook.

Pinto, C. (2024, March 10). *Writing Easier Code Comments with ChatGPT*. https://dev.to/christinepinto/writing-easier-code-comments-with-chatgpt-3g89

Quarkus. (2023). *What Is Quarkus*. Https://Quarkus.Io/About/.

Red Hat. (2023, April 20). *Red Hat OpenShift enterprises with Red Hat Services*.

Russel, S., & Norvig, P. (2022). *Artificial Intelligence - A Modern Approach 4th Edition* (4.). https://link.springer.com/content/pdf/10.1007/BF00993982.pdf

Sarker, I. H. (2021). *AI-Driven Cybersecurity: An Overview, Security Intelligence Modeling and Research Directions*. https://doi.org/10.20944/preprints202101.0457.v1

Shaik, T., Tao, X., Li, Y., Dann, C., McDonald, J., Redmond, P., & Galligan, L. (2022). A Review of the Trends and Challenges in Adopting Natural Language Processing Methods for Education Feedback Analysis. *IEEE Access*, *10*, 56720–56739. https://doi.org/10.1109/ACCESS.2022.3177752

Shavlik, J. W. (1994). *Combining Symbolic and Neural Learning* (Vol. 14). https://doi.org/10.1007/bf00993982

Silkin, O. (2023, March 21). *ChatGPT: 3 productivity tips for IT professionals*.

Stamford, Conn. (2023, July 27). *Gartner Survey Finds 55% of Organizations That Have Deployed AI Take an AI-First Strategy with New Use Cases*. Https://Www.Gartner.Com/En/Newsroom/Press-Releases/2023-07-27-Gartner-Survey-Finds-55-of-Organizations-That-Have-Deployed-Ai-Take-an-Ai-First-Strategy-with-New-Use-Cases.

Symeonidis, G., Nerantzis, E., Kazakis, A., & Papakostas, G. A. (2022). *MLOps -- Definitions, Tools and Challenges*. http://arxiv.org/abs/2201.00162

Synopsys. (2024). *Open Source Software*. Open Source Software.

Towell G. Geoffrey, & Shavlik W. Jude. (1994). Knowledge-based artificial neural networks. *Artificial Intelligence*, 119–165. https://doi.org/https://doi.org/10.1016/0004-3702(94)90105-8

Weizman, T. (2023). Understanding The Benefits And Risks Of Using AI In Business. *Https://Www.Forbes.Com/Sites/Forbesbusinesscouncil/2023/03/01/Understanding-the-Benefits-and-Risks-of-Using-Ai-in-Business/?Sh=4ade03036bba*.

White, S. R., Amarante, L. M., Kravitz, A. V., & Laubach, M. (2019). The future is open: Open-source tools for behavioral neuroscience research. *ENeuro*, *6*(4). https://doi.org/10.1523/ENEURO.0223-19.2019

Wiles, J. (2023, August 15). *Big Picture Rethink how to measure ROI of your GenAI investments*.

Yao, M. (2022, June 17). *10 Leading Language Models For NLP In 2022*. 10 Leading Language Models For NLP In 2022.