Baština Akademije nauka i umjetnosti Bosne i Hercegovine

Basic Technologies and Models for Implementation of Industry 4.0

Karabegović, Isak

2023-10-04

https://bastina.anubih.ba/handle/123456789/779

Preuzeto s Baštine Akademije nauka i umjetnosti Bosne i Hercegovine

https://bastina.anubih.ba/

DOI: 10.5644/PI2023.209.05

The Role of Software Engineering in Industry 4.0

Samir Lemeš[*1]

Abstract: *In Industry 4.0, software engineering plays a critical role in enabling and driving the digital transformation of industries. Key roles of software engineering in Industry 4.0 include the development of IoT and connectivity, automation and robotics, data analytics and artificial intelligence (AI), cybersecurity, human-machine interfaces, software integration and system interoperability. Software engineering provides the foundation for Industry 4.0 by developing the software systems and infrastructure required for automation, connectivity, data analysis, and cybersecurity. It enables the digital transformation of industries, driving efficiency, innovation, and new business models.*

Keywords: *Industry 4.0,Software Engineering, Artificial Intelligence, Cybersecurity*

1. Introduction

Data plays a central role in Industry 4.0, as it is the foundation for driving insights, optimization, and decision-making in various industrial processes. In a data-based Industry 4.0, data collection, analysis, and utilization are key components for driving efficiency, productivity, and innovation. Industry 4.0 relies heavily on software engineering to enable and drive its digital transformation. Software engineering focuses on the systematic and structured approach to designing, developing, testing, deploying, and maintaining software systems. It involves applying engineering principles, methodologies, and practices to create high-quality software that meets specific requirements while considering scalability, reliability, maintainability, and usability.

Kipper et al. in [1] tried to identify what competencies are necessary for Industry 4.0. Apart from general and soft skills, such as leadership, self-organization, creativity, problem-solving, innovation, and adaptability, they also emphasized a set of required skills from the contemporary fields, including algorithms, software development and security, and data analysis. The latter skills are the essence of software engineering.

Peres et al. provided a definition and holistic view of Artificial Intelligence (AI) in Industry 4.0 by analyzing its basic building blocks and future trends. They defined Industrial Artificial Intelligence as an interdisciplinary area of

[*1]University of Zenica, Polytechnic Faculty, 72000 Zenica, Bosnia and Herzegovina
ORCID: 0000-0002-3596-645X, E-mail: samir.lemes@unze.ba

research,encompassing machine learning, natural language processing and robotics, or as a ‘’systematic discipline focusing on the development, validation, deployment and maintenance of AI solutions for industrial applications with sustainable performance’’ [2]. Industrial AI faces challenges, such as data availability, data quality, cybersecurity, privacy, and governance. All these challenges are components of software engineering.

Dalzochio et al. investigated machine learning and reasoning for predictive maintenance in Industry 4.0. They identified issues like scalability, latency, and data security as fields deserving further investigation [3].

Aheleroff et al. explored relationships between Digital Twin and mass individualization. They identified suitable Industry 4.0 technologies and a holistic reference architecture model to accomplish the most challenging Digital Twin-enabled applications [4]. They proposed Digital Twin as a Service (DTaaS) under Industry 4.0.

Aceto, Persico and Pescape analyzed ten technological enablers and their interdependencies for Industry 4.0 [5]. Interestingly, they marked the Blockchain enabler as still in exploration and adoption, as opposed to other enabler technologies (IoT, Cloud Computing, Big Data, Artificial Intelligence). They also identified the transition from proprietary software formats to open standards with multiple open-source implementations as one of the challenges for implementation. Although open-source software boosts the application of IoT, the difficulties in estimating the transitioning costs are marked as a challenge. They correctly identified that challenges for open-source software are not in technological issues but in interaction with legacy conditions, corporate culture, market, and regulations [5].

Alladi et al. reviewed the existing blockchain applications in Industry 4.0 and industrial IoT by presenting the current research trends in various industrial sectors and successful commercial blockchain implementations [6]. Liu et al. [7] proposed standardization for blockchain implementation to provide interoperability between different manufacturers. Liu et al. proposed the decentralization of Industry 4.0 by applying blockchain in Product lifecycle management (PLM).

Klingenberg, Borges and Antunestried identified current technologies related to Industry 4.0 and to developed rationale to enhance the understanding of their functions within a data-driven paradigm [8].Their literature survey showed that Industry 4.0 publications focus on enabling technologies that transmit and process data rather than value-creating technologies, which apply data to develop new solutions.Based on their systematic research results, they suggest intensifyingthe research into Data Application technologies.

Ferreira, Armellini and De Santa-Eulaliasystematicallyanalyzed the development of simulation-based research in Industry 4.0 [9]. Their resultsrevealed that

hybrid simulation, digital twin and discrete-event simulation are the primary simulation-based approaches in the context of Industry 4.0.
Margariaand Schieweck discussed the difference between the Digital Twin and Digital Thread [10]. The Digital Thread connects real things and their twin models, communication networks, decision algorithms, visualizations, construction, and operation within a mature Industry 4.0 environment. They claim that software and software models are underestimated in their relevance, complexity, challenges, and cost.
Key aspects of software engineering include:

- Analysis and design where user needs and requirements are used to define the functionality and features of a software system;
- Development and programming of computer code using programming languages, frameworks, and development tools;
- Quality assurance and testing to identify and fix defects, validate the software against requirements, and ensure its reliability and quality;
- Deployment and maintenance of the software system in the target environment, monitoring the system's performance, addressing bugs and errors, and implementing updates and enhancements to meet changing user needs;
- Project management in teams, collaborating with clients, managers, and other engineers; and
- Documentation and communication, including technical specifications, user manuals, and system documentation.

To ensure efficient and effective software development, software engineering encompasses variouspractices and methodologies, including Agile, Waterfall, DevOps, and Continuous Integration/Continuous Deployment (CI/CD). It emphasizes use of tools, techniques, and standards to manage complexity, improve productivity, and deliver reliable software solutions. By leveraging data and applying software engineering principles, Industry 4.0 organizations can gain a competitive edge through improved decision-making, optimized processes, and enhanced operational efficiency. Data-based approaches enable agility, innovation, and the ability to respond quickly to changing market demands.

2. Development of IoT and Connectivity

Software engineers design and developsoftware systems that enable the Internet of Things (IoT) and connectivity between various devices, machines, and systems. Software development includes creating protocols, communication frameworks, and interfaces to facilitate seamless data exchange and integration.
IoT infrastructure is developed using hardware and software solutions [11]. IoT software engineering processes data collected via sensors with visual

representation and an intuitive user interface. For IoT software development, three components are required: programming language, development platform, and operating system. In exceptional cases, IoT devices can be built without an operating system, using only PLCs (Programmable Logic Controllers). PLCs are programmed for a single application and do not require an operating system to run the application (Figure 1). Pocket-size computers, such as Raspberry Pi, can be used as a basis for IoT devices but require an operating system. They can be programmed to run multiple software applications simultaneously. However, the software is needed to operate both devices.

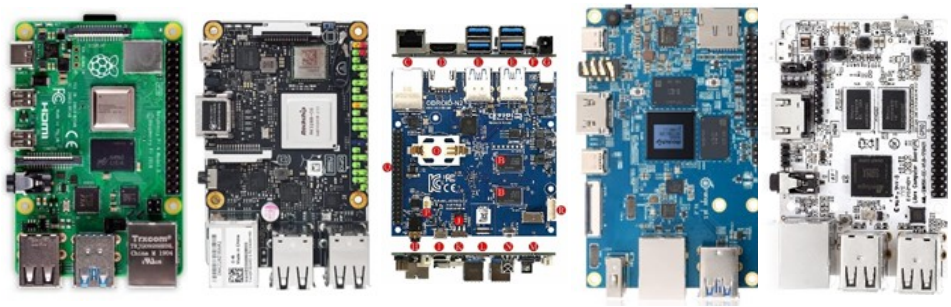
Raspberry Pi Arduino PLC



Figure 1. Raspberry Pi computer needs an operating system, unlike Arduino PLC [12]

Both types of IoT devices can be used for industrial applications. PLC generally offers more automation-specific processing than computers. PLCs include hardware and software watchdogs. Software watchdogs make sure the program is running as desired. For example, poor programming could lead to code execution in an endless loop, causing a harmful and potentially dangerous out-of-control situation [13]. Hardware watchdogs monitor the peripheral devices connected to the PLC, such as switches, sensors, and actuators. These watchdogs are not built-in in Raspberry Pi, and the software routines must be developed or found to deliver this functionality.

Commercially available industrialized products cost more than systems created from consumer electronics controllers but include significant mission-specific hardware and software benefits. Do-it-yourself (DIY) options can be cost-effective but require considerable attention to integrate hardware and software [13] properly.



Raspberry Pi *Asus Tinker Board* *Odroid N2+* *Orange Pi* *Libre Computer Board*

Figure 2. Examples of pocket-size computers used for the development of IoTdevices

IoT devices based on Raspberry Pi or similar computers (Figure 2) consist of low-powered processing units, small RAMs, restricted storage and communication I/O ports used to connect peripherals. They often have built-in network interfaces (Ethernet, Wi-Fi, Bluetooth, Zigby). They use lightweight operating systems due to limited storage capacity. These are some of the most frequently used operating systems in Industrial IoT devices:

- **Contiki-NG**: open-source OS for resource-constrained IoT devices (http://www.contiki-os.org/)
- **FreeRTOS**: a real-time operating system for microcontrollers and small microprocessors, with long-term support (LTS)from Amazon(https://www.freertos.org/)
- **Mbed OS**: open-source OS for IoT devices based on an Arm Cortex-M microcontroller with a broad range of connectivity options (https://os.mbed.com/mbed-os/)
- **Nucleus**: real-time operating system produced by the Embedded Software Division of Mentor Graphics, a Siemens Business, supporting 32- and 64-bit embedded system platforms (https://www.plm.automation.siemens.com/global/en/products/embedded/nucleus-rtos.html)
- **OpenWrt**:open-source project for embedded operating systems based on Linux, primarily used on embedded devices to route network traffic (https://openwrt.org/)
- **Raspbian**: a free operating system based on Debian Linux and optimized for the Raspberry Pi hardware (http://www.raspbian.org/)
- **Tizen**: Linux-based free, open-source IoT OSdeveloped by Samsung Electronics (https://www.tizen.org/)
- **VxWorks**: real-time operating system supporting application deployment through containers, developed as proprietary software by Wind River Systems (https://www.windriver.com/products/vxworks)

- **Windows IoT** (formerly Windows Embedded): a component of the Microsoft Windows 10 OS for IoT devices that run on Arm and x86/x64 devices (https://developer.microsoft.com/en-us/windows/iot/)
- **Zephyr**: a small real-time operating system for connected, resource-constrained and embedded devices supporting multiple architectures (https://zephyrproject.org/)

Software for IoT is developed using different programming languages. **C** and **C++**are programming languages written with a hardware perspective in mind. Since most introductory courses for computer programming are based on these languages, they are the most widely spread languages. Their direct competitor is **Java**. Initially known as the programming language for mobile devices and web-based applicationsmainly because of its high portability, Java is compatible with various peripheral devices and is well-suited for IoT devices. **Python**is a good choice for IoT applications,as it can easily handle data-heavy applications. **JavaScript** is often used for connectivity and interoperability. It can be applied for both back-end and front-end processes. A low entry level and many specialized coders on the market are the main benefits of JavaScript. Although the **Go**language is a relatively new programming language, it has been widely implemented in various IoT projects due to its multipleuses, such as optimized coding.

The software development process for IoT devices differs considerably from the traditional desktop computer process. Developing software for IoT involves unique considerations due to the distributed nature of IoT systems and the diversity of devices and protocols involved. A typical software development process for IoT consists of the following phases:

- Requirements gathering: Understand the business objectives and functional requirements of the IoT system. Identify the devices, sensors, actuators, and data sources involved. Determine the desired interactions, data flows, and integration points.
- Architecture and design: Design the overall system architecture, including the interactions between IoT devices, edge devices, gateways, cloud platforms, and user interfaces. Consider factors like scalability, reliability, security, and data management. Define protocols, APIs, and data formats for communication between devices and other components.
- Device selection and integration: Select appropriate IoT devices based on the requirements and architecture. Develop or configure drivers and protocols to enable communication and integration of devices within the system. Ensure compatibility and interoperability among different devices.
- Connectivity setup: Set up the necessary connectivity infrastructure, such as Wi-Fi, Bluetooth, or cellular networks, to establish

communication between devices and gateways and configure network settings, security protocols, and authentication mechanisms.

- Data management: Define the data collection, storage, and processing mechanisms. Determine the data flow, including local data processing at the edge or cloud-based processing, and design databases or data repositories to store and manage IoT data efficiently.
- Software development: Develop the software components required for the IoT system, including firmware for devices, software for gateways and edge devices, cloud-based applications, and user interfaces. Implement functionalities such as data acquisition, device control, data processing, analytics, and visualization.
- Security implementation: Incorporate security measures to protect the IoT system from threats. Implement secure communication protocols, encryption, access control, and authentication mechanisms. Consider data privacy and compliance with regulations.
- Testing and validation: Conduct thorough testing of the IoT system components, including devices, connectivity, data transmission, and software functionalities. Perform integration testing to ensure seamless communication and data flow. Validate the system against the defined requirements and use cases.
- Deployment and monitoring: Deploy the IoT system in the target environment. Set up monitoring and management tools to track device performance, connectivity, and data flow. Implement mechanisms for remote device management, software updates, and troubleshooting.
- Continuous improvement: IoT systems often require constant iteration and improvement. Gather feedback, monitor system performance, and analyze user behaviour to identify areas for enhancement. Regularly update software components, address security vulnerabilities, and add new features based on user needs and evolving technologies.

It is crucial to collaborate with cross-functional teams throughout the software development process, including hardware engineers, data scientists, network specialists, and domain experts, to ensure a holistic approach to IoT system development.

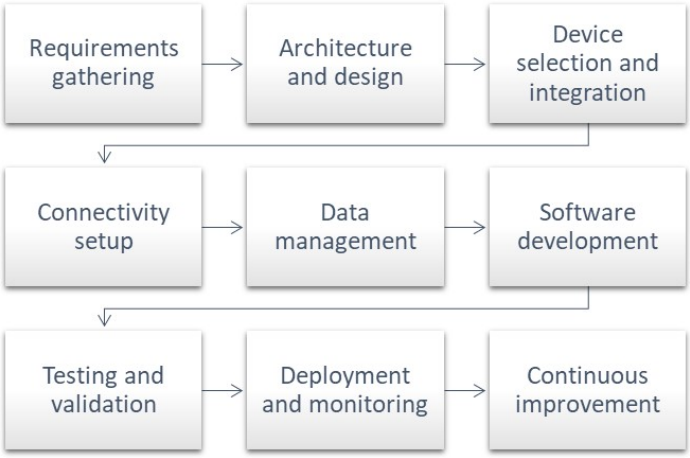


Figure 3. The software development process for IoT

The connectivity aspect of IoT refers to the ability of devices, sensors, and systems to connect and communicate with each other, enabling data exchange and enabling the functioning of an IoT ecosystem. Connectivity is a fundamental requirement for IoT systems to operate effectively.

IoT devices use various communication protocols to exchange data. Common protocols include Wi-Fi, Bluetooth, Zigbee, Z-Wave, LoRaWAN, cellular networks (2G/3G/4G/5G), and Ethernet. The selection of a communication protocol depends on factors like range, data rate, power consumption, and specific use case requirements. Wireless connectivity enables devices to communicate without physical wired connections. Wi-Fi and Bluetooth are widely used for short-range communications within a local area, while cellular networks provide broader coverage and enable devices to connect over long distances. Low-power protocols like Zigbee and Z-Wave are suitable for home automation and smart devices.

Cloud platforms are frequently utilized in IoT systems to store, process, and analyze data from connected devices. IoT devices establish connections to the cloud to transmit data, receive commands, and access cloud-based services. Cloud connectivity enables centralized management and data storage and enables advanced analytics and machine learning algorithms to process data at scale.

In IoT systems, edge computing often comes into play to process and analyze data closer to the source, reducing latency and bandwidth requirements. Gateways act as intermediaries between devices and the cloud, aggregating data from multiple devices and enabling communication with the cloud infrastructure.

Wireless Standard	Power	Transmission Range (typical)	Data Rates
Bluetooth	Medium	1 to 100 m	1 to 3 Mbps
Bluetooth LE	Lower	>100 m	125 kbps to 2 Mbps
LoRaWAN	Low	10 km	0.3 to 50 kbps
NB-IoT	Low	<35 km	20 kbps to 5 Mbps
NFC	Low	<10 cm	106 to 424 kbps
Sigfox	Low	3 to 50 km	100 to 600 bps
6LoWPAN	Low	100 m	0 to 250 kbps
802.11/Wi-Fi	Medium	100 m to several km (with boosters)	10 to 100+ Mbps
802.15.4/Zigbee	Low	10 to 100 m	20 to 250 kbps
Z-Wave	Low	15 to 150 m	9.6 to 40 kbps

Figure 4. Features of major wireless connectivity technologies used in IoT [14]

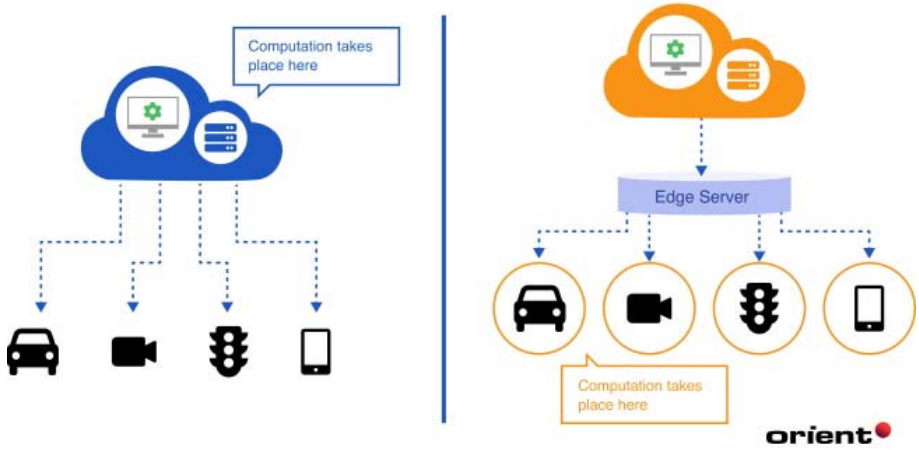


Figure 5. Difference between Edge computing and Cloud computing [15]

IoT platforms provide tools and services that facilitate connectivity and data management in IoT systems. They offer APIs (Application Programming Interfaces) that enable developers to integrate devices, collect data, and control device behaviour. These APIs allow seamless communication between devices, gateways, and cloud services, simplifying the development and integration process.

IoT systems often involve many devices, which require scalable network management. Network management tools and techniques ensure

smoothoperation, efficient resource utilization, and device connectivity and performance monitoring.
Finally, security is a critical concern in IoT connectivity. As IoT systems involve numerous interconnected devices and data exchanges, security measures are necessary to protect against unauthorized access, data breaches, and cyber threats. Encryption, authentication mechanisms, secure communication protocols, and regular software updates are essential to maintain the security and privacy of IoT systems.

3. Automation and Robotics

Industry 4.0 emphasizes use of automation and robotics to improve efficiency and productivity. Software engineers are responsible for developing the software that controls and manages these automated systems. They design algorithms, develop control systems, and program robots to perform complex tasks in logistics, manufacturing, and other industries.
Robot Operating System (ROS) is a popular open-source framework for developing and controlling robots. It provides a collection of software libraries and tools that facilitate communication, hardware abstraction, sensor integration, and robot control. ROS enables the development of modular and interoperable robot software components.ROS is released as distributions. Some releases come with long-term support (LTS), which means better stability is obtained through extensive testing. Other distributions have shorter lifetimes butsupport more recent platforms and versions of their constituent ROS packages [16].
ROS distributions require an operating system and are tailored for installing different OSs. ROS Foxy Fitzroy runs on Ubuntu (Debian) Linux and Windows 10. ROS Noetic Ninjemysis made for Ubuntu Linux. ROS Humble Hawksbill, ROS Iron Irwini, and ROS Rolling Ridley run on Ubuntu (Debian), Windows 10 and RedHat Linux. ROS Melodic Moreniais primarily targeted at Ubuntu, though other Linux systems, as well as Mac OS X, Android, and Windows, are supported to varying degrees [16].

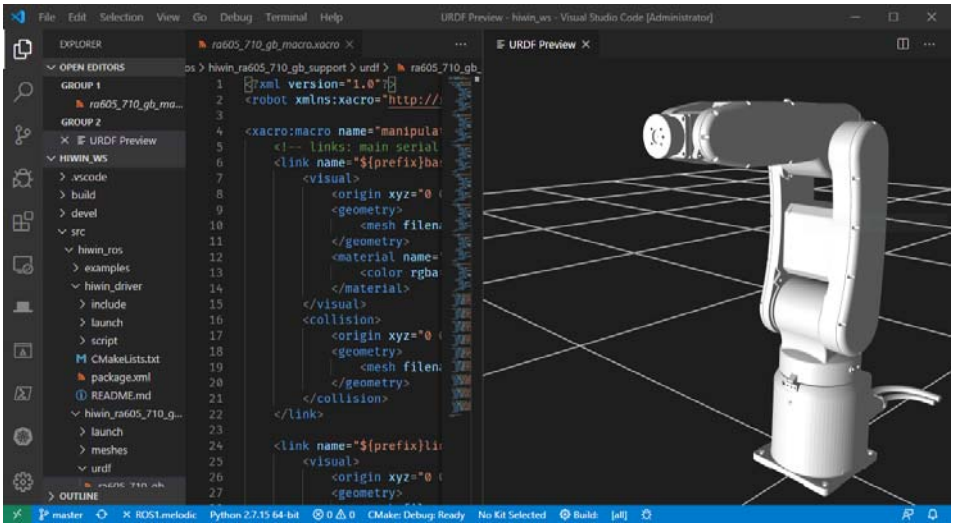


Figure 6. Robot Operating System (ROS)integrated with Microsoft Azure services [17]

Industry 4.0 envisions close collaboration between humans and machines. Software engineers are responsible for designing intuitive and user-friendly human-machine interfaces (HMIs) that enable effective communication and interaction between operators and automated systems. They focus on usability, visualizations, and integrating different technologies to create seamless interfaces. Human-Machine Interfaces are software applications that provide a graphical user interface (GUI) for operators to interact with automation systems and robots. HMIs allow users to monitor system status, control processes, and receive feedback. They often include features like data visualization, alarms, and data logging.

Smart cyber-physical systems require new types of interfaces for smooth human-machine interaction, primarily when operating in dark or dustyenvironments. New types of HMIs are being implemented in Industry 4.0, such astouch-screen displays, voice interfaces, gesture interfaces, and ARtools [18].Touch-screen displays have been in use for more than 20 years already. Modern touch interfaces allow managing machines even in gloves. In Industry 4.0, voice interfaces have limited use due to noisy conditions. Where available, voice recognition controls appliances on the shop floor and asks questions regarding machines' outputs. Gesture control enables touchless manipulation of industrial devices using wired or wireless gloves, cameras, or hand-tracking controllers. Augmentedreality combines simulated environments with real-world video to allow control of machinery or computers. It can assist the assembly process by projecting leading lines or alphanumerical instructions on an assembly floor or using AR glasses, thus reducing the risk of mistakes.



Figure 7. Evolution of human-machine interfaces

Computer-Aided Design (CAD) and Computer-Aided Manufacturing (CAM)software are used in robotics and automation systems' design and manufacturing stages. CAD tools enable engineers to design and model robotic systems, while CAM software translates these designs into machine instructions for manufacturing processes like milling, cutting, or 3D printing. Typical CAD/CAM user needs only functional skills for manipulating graphic data and designing 3D models. In some cases, software engineering and computer programming skills are required to automate the design process and perform advanced engineering tasks such as shape optimization, statistical analysis, or photorealistic visualization. Simulation software allows engineers to virtually model and test automation and robotic systems before physical implementation. Digital Twin is a technology that creates a virtual replica of physical systems, enabling real-time monitoring, analysis, and optimization. Simulation and Digital Twin software help optimize system performance, identify bottlenecks, and predict maintenance needs.

Artificial Intelligence (AI) and Machine Learning (ML) technologies are increasingly integrated into automation and robotics systems. Software applications leverage AI and ML algorithms to enable advanced capabilities like machine vision, object recognition, autonomous decision-making, and predictive maintenance. These technologies enhance the flexibility, adaptability, and intelligence of automation and robotics systems.

Workflow and process automation software enable the automation of repetitive tasks, coordination of processes, and workflow optimization. These tools provide visual programming interfaces or rule-based engines to define workflows, trigger actions, and integrate different systems and components. They improve operational efficiency, reduce errors, and enhance collaboration.

With increased connectivity and digitalization, robust cybersecurity solutions are crucial for protecting automation and robotics systems from cyber threats. Software applications focused on cybersecurity provide features like intrusion detection, encryption, access control, and vulnerability management to safeguard industrial systems.
All these software applications form a technology stack that enables automation, control, optimization, and intelligent decision-making in Industry 4.0 robotics and automation systems. They increase productivity, efficiency, and flexibility in manufacturing and other industries.

4. Data Analytics and Artificial Intelligence

In Industry 4.0, there is a significant focus on leveraging data and applying artificial intelligence (AI) techniques for improved decision-making and predictive analytics. Software engineers develop applications that collect, process, and analyze large volumes of data generated by smart sensors, machines, and systems. They also build AI models and algorithms to extract insights, optimize processes, and enable predictive maintenance.Artificial Intelligence (AI) contributes to Industry 4.0 in different areas: machine learning, computer vision, natural language processing for HMIs, intelligent decision support and cognitive automation.
In recent years, machine learning has led to remarkable progress in many application areas, all thanks to the availability of more significant amounts of data and advances in computer technology that have enabled data processing in increasingly complex machine learning models. Machine learning (ML) is a component of artificial intelligence where, by applying information technologies to existing data, which are seemingly unrelated, and appropriate algorithms, one tries to discover new knowledge that could then be used to solve new problems. Machine learning algorithms can be classified into several categories depending on the learning method:

- supervised learning,
- unsupervised learning,
- assisted learning, and
- deep learning.

Supervised learning implies that for each instance from the training set, the class to which it belongs is also known, i.e., it is necessary to manually classify the training data so that the algorithm can learn to do it on new data [19, 20]. It is most often applied in cases of classification and regression. Unsupervised learning does not require human intervention, and the algorithm recognizes patterns in the training data. Reinforced learning means learning where the model's performance is supported by appropriate reward and punishment depending on what results the model gives. In deep learning, we have models

with multiple layers of artificial neural networks that can make intelligent decisions based on large amounts of data.
Computer vision techniques use AI to enable machines to interpret and understand images and videos. It finds applications in quality inspection, object recognition, defect detection, and even autonomous robotics.
Natural Language Processing (NLP) enables machines to understand and process human language. It facilitates human-machine interaction, voice-based control systems, chatbots, and natural language interfaces for data querying and analysis.
AI technologies provide decision support systems that assist operators, managers, and engineers in making informed decisions. They analyze data, provide insights, and recommend actions based on historical and real-time information. Cognitive automation combines AI technologies with robotic process automation (RPA) to automate complex tasks that require cognitive abilities like reasoning, learning, and understanding. It helps streamline administrative processes, reduce errors, and enhance efficiency.
By leveraging data analytics and AI in Industry 4.0, organizations can gain valuable insights, optimize processes, improve productivity, enhance quality control, enable predictive maintenance, and drive innovation. These technologies empower businesses to make data-driven decisions and unlock new opportunities for growth and competitive advantage.
ML and AI require a significant amount of computing power, which is mainly used in cloud computing. However, thousands of IoT devices represent a significant potential which is not utilized at full capacity. Most of the time, IoT devices are idle, creating a possibility to supplement the computing power in the cloud. Cloud computing is usually charged per usage, and operating costs can be very high. Edge computing is a technology trying to utilize distributed computing resources from the network of connected IoT devices.

4.1.Edge computing

Internet of Things devices are increasingly used in industry, thus, the amount of data they generate increases. In the case of data processing in the cloud, the data must first be transferred via an Internet connection, undergo processing, and only then be returned to an IoT device or another terminal device [21]. This, in addition to increasing the application's response time, also represents a growing challenge for the throughput of Internet connections due to the increased volume of data. This is especially distinct in cases where machine learning is applied in computer vision applications. For this reason, previous research suggests using edge computing to create solutions that fit the application with IoT devices so that their potential can be used to the best advantage.
In contrast to the cloud approach, where computing resources are centralized at one remote location, with edge computing, data processing approaches the

datasource itself [22, 23]. Here, one must consider that edge devices usually have weaker computing resources in terms of accommodation and data processing capabilities. For this reason, it is most often resorted to combining cloud and edge computing, where more demanding tasks, such as the machine learning process, take place in the cloud, which has much larger resources adapted for that purpose. In contrast, trained models can be adapted to limited edge resources and thus ensure the functioning of the process inferences close to the data source. Edge computing is a relatively new technology that is not yet standardized. Many platforms still work through incompatible protocols, which is a limitation during implementation [24].

Although machine learning in edge computing has advantages over cloud computing, certain trade-offs are necessary, including cost, reliability, network infrastructure, data privacy, and storage space. Material costs are rising to support machine learning in edge computing, but with large volumes of cloud transactions, cloud communication costs for the device are also increasing. Reliability significantly depends on the speed of Internet access. Edge computing eliminates delays associated with network resources. As some applications require data to be stored locally, keeping specific data on the device can improve data privacy. The IoT devices on which edge computing is based do not have enough storage space, which introduces a new challenge to storage resources.

Web 3.0 is a new concept that uses blockchain technologies to decentralize the Internet [25]. It is based on several principles: decentralization, no permissions, payment is exclusively in cryptocurrencies, and it does not use intermediaries to establish user trust but relies on a distributed blockchain system. Blockchain creates a faster, decentralized, and more secure computing environment. In contrast, Edge computing provides the supporting infrastructure that enables fast and reliable transactions, especially with advanced implementations of blockchain proof of stake (PoS) [26].

5. Cybersecurity

With the increased connectivity and digitalization in Industry 4.0, the importance of cybersecurity cannot be understated. Software engineers developsecure software systems, implementing robust authentication and encryption mechanisms and addressing vulnerabilities and threats. They work to ensure the integrity, confidentiality, and availability of data and systems in the industrial environment.

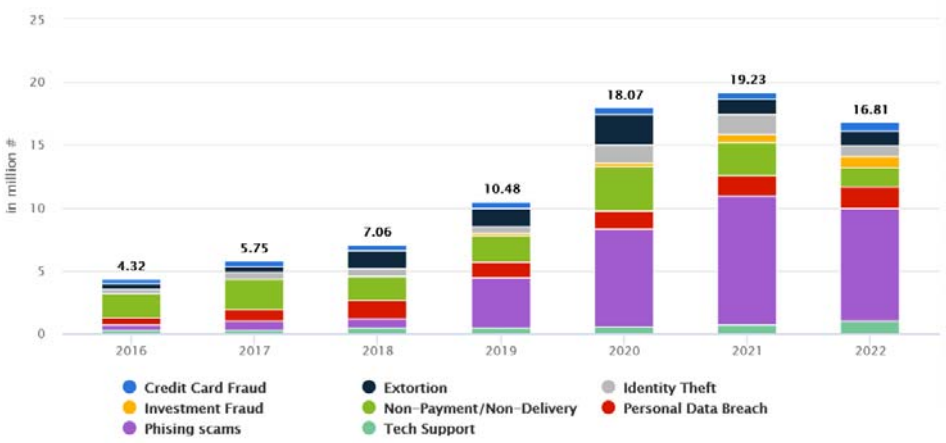


Figure 8. Recorded cyber-attacks [27]

Figure 8 illustrates the increase in cyber-attacks in the past seven years, indicating that phishing scams are the most frequently recorded cyber-attacks. Figure 9 reveals that in 2022 manufacturing had the highest share of cyber-attacks among the leading industries worldwide. During the examined year, cyber-attacks in manufacturing companies accounted for nearly 25% of the total cyber-attacks, even more than finance and insurance [28].

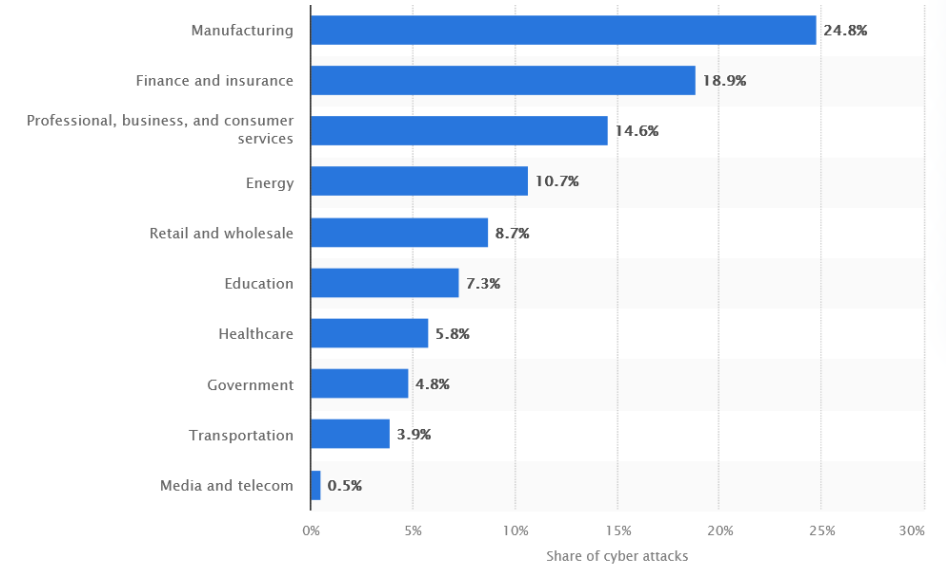


Figure 9. Distribution of cyber-attacks across worldwide industries in 2022 [28]

Cybersecurity is paramount in software engineering due to the increasing reliance on digital systems, interconnected networks, and the ever-evolving

threat landscape. Cybersecurity is crucial in software engineering. Cybersecurity ensures that sensitive data, such as personal information, financial records, and intellectual property, remains confidential and protected from unauthorized access. This is particularly important in finance, healthcare, and government sectors, where data privacy regulations mandate strict protection measures. But statistical data shows that the increasingly automated and computerized manufacturing sector is exposed to many cyber-attacks. Figure 10 shows how the number of IoTcyber-attacks worldwide amounted to over 112 million in 2022. Over the recent years, this figure has increased significantly from around 32 million detected cases in 2018. In the latest measured year, the year-over-year increase in the number of IoTmalware incidents was 87% [29].

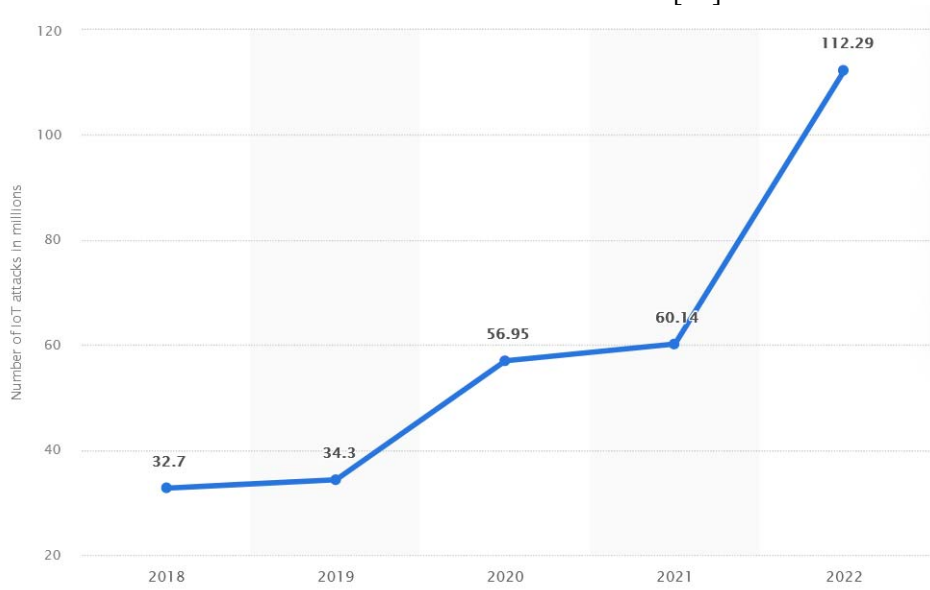


Figure 10. The annual number of IoT malware attacks worldwide from 2018 to 2022 [29]

Cybersecurity measures protect software systems from unauthorized modifications, tampering, or unauthorized access. By ensuring system integrity, cybersecurity prevents unauthorized individuals from altering critical functionalities, manipulating data, or introducing malicious code that could compromise the system's reliability and stability. Cybersecurity helps maintain the availability and continuous operation of software systems. It safeguards against distributed denial-of-service (DDoS) attacks, which overload systems with excessive traffic, causing services to become inaccessible. By implementing robust security measures, software engineers can prevent or mitigate such attacks, ensuring system availability for users.

Security breaches and cyber-attacks can lead to substantial financial losses, lawsuits, and reputational damage for organizations. The recovery cost from a security incident, investigating the breach, notifying affected parties, and restoring systems can be significant. Organizations can reduce the risk of breaches and the associated financial and reputational repercussions by prioritizing cybersecurity in software engineering.

Cybersecurity measures, such as secure coding practices and vulnerability management, help defend against malware and ransomware attacks. These attacks can compromise systems, encrypt data, and demand ransom for its release. Robust security practices, including regular patching, intrusion detection systems, and secure software development, can significantly mitigate the risk of malware and ransomware infections.

Many industries have regulatory requirements that mandate strong cybersecurity measures. Organizations must adhere to regulations such as the European General Data Protection Regulation (GDPR) or the Payment Card Industry Data Security Standard (PCI DSS). By incorporating cybersecurity best practices into software engineering processes, organizations can demonstrate compliance with these regulations.

The cybersecurity landscape constantly evolves, with new threats and attack vectors emerging regularly. Software engineers need to stay abreast of the latest vulnerabilities, exploits, and security best practices to develop secure software. Software engineers can stay one step ahead of potential attackers by continuously updating security measures and addressing emerging threats.In an increasingly digital world, customers value trust in their software systems. Organizations can demonstrate their commitment to protecting customer data and providing secure services by prioritizing cybersecurity. This fosters trust and confidence among users, leading to stronger customer relationships and brand loyalty.

Cybersecurity is essential in software engineering to protect data, ensure system integrity and availability, mitigate financial losses and reputational damage, defend against malware and ransomware, comply with regulations, address emerging threats, and foster customer trust. Incorporating robust cybersecurity practices throughout the software development lifecycle is crucial for building secure and resilient software systems in today's interconnected and threat-prone environment.

Cybersecurity is critical to Industry 4.0 as integrating digital technologies, interconnected systems, and the Internet of Things (IoT) introduce new security risks. Designing a secure architecture is crucial for Industry 4.0 systems, involving implementing defense-in-depth principles, segmenting networks, and ensuring secure communication protocols between devices, sensors, and systems. Security should be considered at every architecture layer, from edge devices to cloud infrastructure.

IoT devices and sensors in Industry 4.0 systems should have built-in security features, such as strong authentication mechanisms, encryption for data in transit and at rest, secure boot processes, and over-the-air firmware updates to address vulnerabilities and patch security flaws.
Industry 4.0 networks should have robust security measures in place. Secure configuration of network devices, firewalls, intrusion detection and prevention systems, and regular network traffic monitoring for anomaliesis necessary. Network segmentation can help limit the impact of potential breaches and contain security incidents.
Strong authentication mechanisms, such as multi-factor authentication, should be implemented to ensure that only authorized individuals can access critical systems and data. Access control policies should be enforced to restrict privileges based on roles and responsibilities, limiting access to sensitive information and functions.
Industry 4.0 systems involve the collection and processing of vast amounts of data. Encryption should be employed to protect data both in transit and at rest. Privacy considerations, including compliance with data protection regulations, should be addressed to safeguard personal and sensitive information.
Implementing security monitoring tools and techniques allows for continuous monitoringof Industry 4.0 systems. Detecting abnormal behaviour, intrusion attempts, and potential security incidents is necessary. Security Information and Event Management (SIEM) proprietary solutions, log analysis, and anomaly detection can help promptly identify and respond to threats.
Establishing an incident response plan is essential to respond to cybersecurity incidents effectively. Roles and responsibilities, incident detection and reporting processes, and procedures for containment, eradication, and recovery should be defined. Regular testing and simulations of incident response plans help ensure their effectiveness.
Cybersecurity awareness and training programs should be provided to employees to educate them about best practices, potential threats, and their roles and responsibilities in maintaining security. These programs helpfoster a cybersecurity culture and reduce the risk of human error leading to security incidents.
Industry 4.0 systems often involve multiple vendors and suppliers. It is essential to ensure they have robust cybersecurity practices, such as performing due diligence when selecting vendors, establishing contract security requirements, and regularly auditing and monitoring their security measures.
The ISO 27000 family of international standards is published jointly by the ISO (International Organization for Standardization) and IEC (International Electrotechnical Commission). These standards enable organizations to manage the security of digital assets, and there is an internationally recognized

certification scheme for organizations, similar to the ISO 9000 certification system.

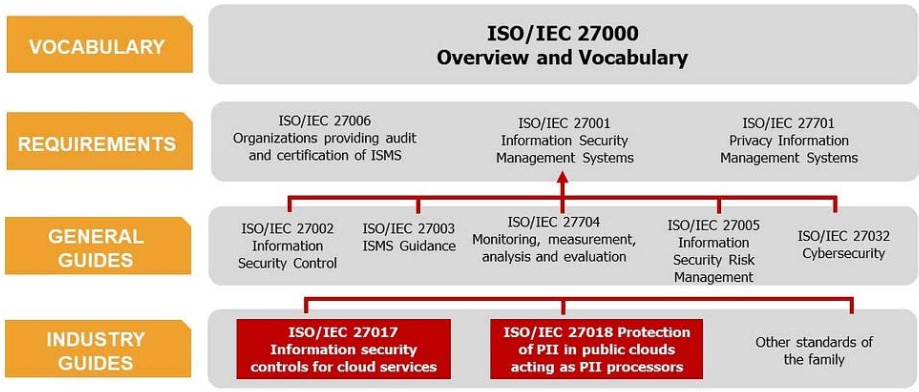


Figure 11. The ISO 27000 series of standards[30]

The ISO 27001 standard provides a framework for establishing, implementing, maintaining, and continually improving an information security management system (ISMS). Implementing ISO 27001 can significantly enhance cybersecurity in Industry 4.0 by addressing essential security requirements and best practices. ISO 27001 requires organizations to conduct a systematic risk assessment to identify and assess information security risks. The risk assessment includes identifying assets, assessing vulnerabilities and threats, and determining the impact and likelihood of risks. Organizations can proactively identify potential cybersecurity risks in Industry 4.0 systems by conducting comprehensive risk assessments and implementing appropriate controls to mitigate them.

ISO 27001 provides a set of controls outlined in Annex A, which covers various domains of information security. These controls include technical measures, policies, procedures, and organizational measures organizations can implement to protect their information assets. By implementing these controls, organizations can enhance the security of their Industry 4.0 systems and address specific cybersecurity challenges.

ISO 27001 emphasizes the importance of secure software development practices. It encourages organizations to follow safe coding guidelines, conduct security testing, and ensure secure configuration and management of software and systems. Adhering to these practices in Industry 4.0 software development helps identify and mitigate vulnerabilities and reduce the risk of cyber threats.

Industry 4.0 systems often involve multiple vendors and suppliers. ISO 27001 promotes effective supplier management practices, including assessing the security capabilities of suppliers, defining security requirements in contracts, and establishing processes for monitoring and auditing suppliers' security measures.

Organizations can mitigate the risk of vulnerabilities introduced through the supply chain by ensuring that suppliers meet appropriate security standards.
ISO 27001 requires organizations to establish incident response and business continuity management processes. These processes include identifying and responding to security incidents, conducting incident investigations, and ensuring timely recovery and restoration of critical systems. Organizations can effectively mitigate the impact of cybersecurity incidents in Industry 4.0 systems by having robust incident response and business continuity plans.
ISO 27001 emphasizes the importance of security awareness and training for employees. It encourages organizations to provide regular security awareness programs to educate employees about information security risks, policies, and procedures. By raising employee awareness and knowledge about cybersecurity, organizations can reduce the risk of human error and improve the overall security posture of Industry 4.0 systems.
ISO 27001 promotes a culture of continuous improvement in information security management. It requires organizations to conduct regular internal audits, management reviews, and risk assessments to identify areas for improvement. By continuously monitoring and assessing the effectiveness of security controls, organizations can adapt to emerging cybersecurity threats and maintain compliance with evolving standards and regulations.
Implementing ISO 27001 standards in Industry 4.0 systems provides a systematic and comprehensive approach to cybersecurity. It helps organizations identify and mitigate security risks, establish robust controls, ensure secure development practices, manage suppliers, respond to incidents, and foster a culture of security awareness. By adhering to ISO 27001 standards, organizations can strengthen their cybersecurity posture, protect critical assets, and ensure the secure operation of Industry 4.0 systems.
By considering these cybersecurity measures in the design, implementation, and operation of Industry 4.0 systems, organizations can mitigate identified risks, protect the most critical assets, and ensure the reliability and integrity of their operations in the face of evolving cybersecurity threats.

6. Software Integration and System Interoperability

In Industry 4.0, there is a need to integrate various software systems and ensure interoperability between heterogeneous devices and platforms. Software engineers work on integrating software components, designing APIs (Application Programming Interfaces), and creating middleware solutions that enable seamless data flow and communication across different systems and technologies.
Software integration is crucial in Industry 4.0, enabling seamless connectivity and interoperability of diverse systems, devices, and software applications.

Figure 12 shows this development in a factory of Industry 4.0 and introduces the principles of vertical and horizontal integration [31]. Vertical integration means merging planning and development with production. Horizontal integrationis performed with the networked production, increased interconnections and information interexchangeamong departments and companies. Different software solutions support all these services and resources, which should also be integrated vertically and horizontally to enable digital factories within Industry 4.0.



Figure 12. Vertical and horizontal integration under Industry 4.0 [31]

Industry 4.0 integrates various systems, such as enterprise resource planning (ERP) and manufacturing execution systems (MES), supply chain management systems, and IoT platforms. Software integration enables these systems to communicate and share data in realtime, facilitating end-to-end visibility and control over the entire value chain. The proliferation of IoT devices and sensors in Industry 4.0 requires integrating their data and functionality into existing software systems, IoT platforms, data ingestion mechanisms, and sensor data processing capabilities to enable real-time data collection, analysis, and decision-making.

Software integration enables the integration and consolidation of data from various sources, including sensors, machines, enterprise systems, and external data feeds. This integrated data can then be analyzed using advanced analytics techniques to derive insights, support decision-making, and drive optimization in predictive maintenance, quality control, and resource utilization.

Many organizations have existing legacy systems that need to be integrated with new Industry 4.0 technologies and platforms. Software integration allows for the seamless connection between legacy systems and modern applications, enabling data exchange and leveraging the functionalities of both systems. Legacy systems can be retrofitted by introducing low-cost IoT sensors and controllers, thus allowing the automation of old analog systems.

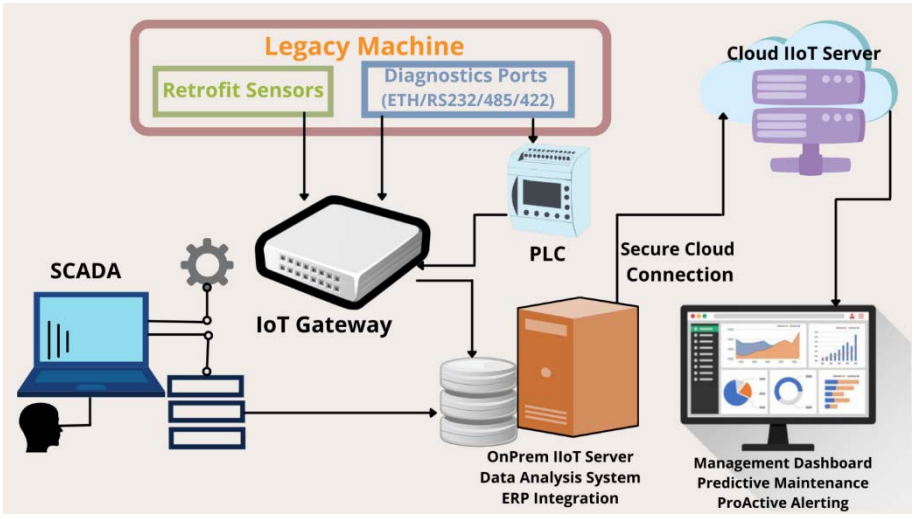


Figure 13. Retrofitting legacy machineswith IoT [32]



Figure 14. Retrofitting boosts the machine's productivity, uptime, reliability, and efficiency [33]

Software integration in Industry 4.0 often involves addressing the challenge of diverse systems and technologies from different vendors. Standardization and adherence to interoperability standards, such as OPC UA (Open Platform Communications Unified Architecture), MQTT (Message Queuing Telemetry Transport), and RESTful APIs (Representational State Transfer), facilitate smooth integration and communication between systems. Software integration enables the integration of cloud platforms and services with on-premises systems

and edge devices. This allows for scalable computing power, storage, analytics capabilities, and real-time decision-making at the edge of the network.
Application programming interfaces (APIs)are a key enabler for software integration in Industry 4.0. APIs enable interaction between the different software applications and systems and share data in a standardized and controlled manner. They provide the interfaces and protocols for seamless integration, allowing the developers to build integrations easily.
Effective software integration in Industry 4.0 ensures that disparate systems, devices, and applications work harmoniously, enabling data-driven decision-making, process optimization, and improved operational efficiency. It allows the seamless flow of information across the entire value chain, enhances collaboration, and supports the realization of the full potential of Industry 4.0 technologies.

6.1. Software development process

The software development process is divided into smaller, parallel, or sequential steps or subprocesses to improve the design or management of the software product. It is also known as the Software Development Life Cycle (SDLC). Traditional methods are based on a series of development steps, such as requirements definition, solution development, quality testing, and implementation. These methods include waterfall, spiral development, prototyping, incremental and iterative development, rapid application development, and extreme programming (XP).
The waterfall method consists of stages with a precisely defined sequence, which must be completed before moving on to the next stage. In the first phase, system and software requirements are defined. The second phase establishes the software design. In the third phase, programmers write the code, the fourth phase is the testing phase (and integration if it is a complex system), the fifth phase includes implementation, training and documentation, and the sixth phase is software maintenance. Its main advantage is simplicity and ease of use, and its disadvantage is that it does not allow much thought or revision. When an application reaches the testing phase, it is complicated to go back and change something that was not well documented or thought out in the concept phase.

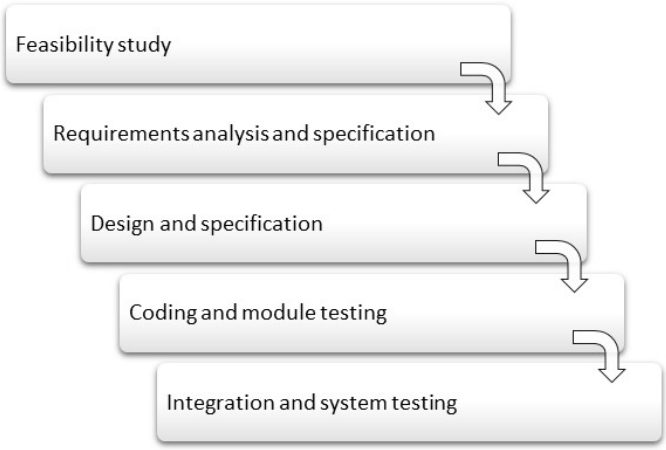


Figure 15. The waterfall method of software development

The spiral method consists of four phases: goal setting, risk assessment, development and quality control, and planning. The spiral method is used for large, complex projects. The progressive nature allows developers to break a large project into smaller pieces and tackle a single feature, ensuring nothing is missed. High costs, dependence on risk analysis, complexity, and complex time management could limit this method.

The Unified Process Method is a software development process that uses the UML (Unified Modeling Language) language to represent the model of the software system being developed. It is iterative, architecture-oriented, use-case-driven and risk-facing. It is based on expanding and refining the design through multiple iterations, with cyclical feedback and adaptation. The system is developed gradually over time, iteration by iteration, so this approach is also known as iterative and incremental software development. The iterations are divided into four phases,each consisting of one or more iterations: preparation, elaboration, construction, and transition.

Changing needs and market conditions have caused the emergence of agile software development methods. Agile methods have mostly the same characteristics:

- Specification, design, and implementation processes are intertwined. There is no detailed specification, and design documentation is automated or minimized.
- The system is developed through a series of versions, where users play a role in changing the next version. That's why agile methods are incremental, with small and fast increments.

Agile methods strive to achieve speed and resolve processes that are considered unnecessary. These methods include Extreme Programming (XP),

Scrum,Crystal, Adaptive Software Development (ASD), Dynamic System Development Method (DSDM), and Feature Driven Development (FDD).
XP (eXtremeProgramming) has short development cycles, incremental planning, constant feedback, reliance on communication and evolutionary design.
Scrum is an agile software development method that focuses on managing an iterative process rather than on individual technical approaches. Scrum is designed to work with small and agile teams of three to nine members led by a Scrum Master, who is responsible for ensuring that the team adheres to the rules and values of Scrum and interacts with people outside the team. He does not tell the team how to do their work; they decide it themselves. He only manages the process and gives general instructions.
The Scrum process is simple, and the so-called sprint is the central part that determines it. A sprint is a relatively short time frame of two to four weeks in which one increment of an iterative process is made. A sprint includes planning, daily Scrums, engineering work, reviews, and revisions. Daily Scrums are 15-minute daily meetings led by the Scrum Master, where they plan what should be done in the next 24 hours. The sprint review takes place at the very end of the sprint when what has been done is compared to what was planned, sometimes with the participation of potential product customers. A review (retrospection) allows the development team to express its opinion. This meeting happens after the review and before planning the next sprint.



Figure 16. Sprint is the scrum term for iteration[34]

The Crystal method focuses primarily on people and their interactions. Software development is treated as a game where everyone is encouraged to interact, be creative and generate ideas. The size of the development team defines the methods, which can be Clear (up to 6 people), Yellow (7-20 people), Orange (21-40 people), Red (41-80 people) and Maroon (over 80 people).
Adaptive Software Development (ASD) aims to enable teams to adapt efficiently to changing market demands or needs by developing products with light planning and continuous learning. Adaptability enables teams to align with organizational goals. The ASD approach encourages teams to develop through a three-phase process: speculate, collaborate, and learn. The benefits of ASD include end-user focus, which can lead to improved and more intuitive products,

enabling early delivery and encouraging greater transparency between developers and clients. Disadvantages of ASD are that it requires more user involvement, testing is integrated into each phase which increases project costs, and rapid iteration and continuous feedback that can slow down the process.
Dynamic System Development Method (DSDM) focuses on the entire product life cycle and consists of the following phases: feasibility study, business study, functional model iteration, design and development iteration, and implementation. Development participant roles vary from project to project and team to team and may include business architect, quality manager, system integrator, and many others.

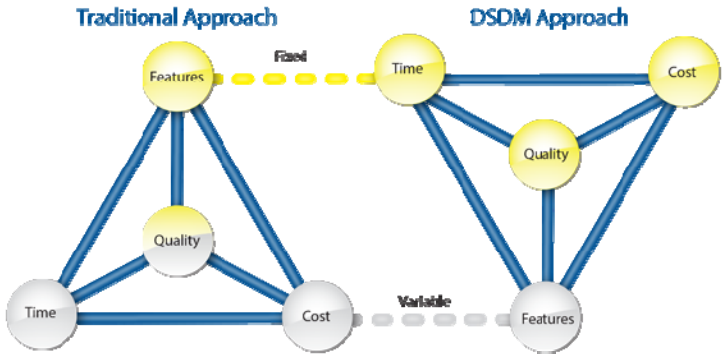


Figure 17. Dynamic System Development Method[35]

Feature Driven Development (FDD) is an iterative and incremental software development process. FDD brings together several industry-recognized best practices into a cohesive whole. These practices are driven from the perspective of features valued by clients. It consists of five main units: development of the overall model, construction of the list of features, planning by features, design of features, and construction of features. The characteristics (Features) mean different functionalities of the software. FDD gives the team an excellent understanding of the scope and context of the project. FDD also requires fewer meetings, unlike the Scrum method.
The Kanban Method, originating from just-in-time manufacturing, is a means to design, manage, and improve flow systems for knowledge work in software development [37]. The method allows organizations to start with their existing workflow and drive evolutionary changeby visualizing their workflow and limiting work in progress (WIP). Kanban systems use mechanisms such as a kanban board to visualize work and its process. Various Kanban boards are used: basic, advanced, and Heijunka boards [37].

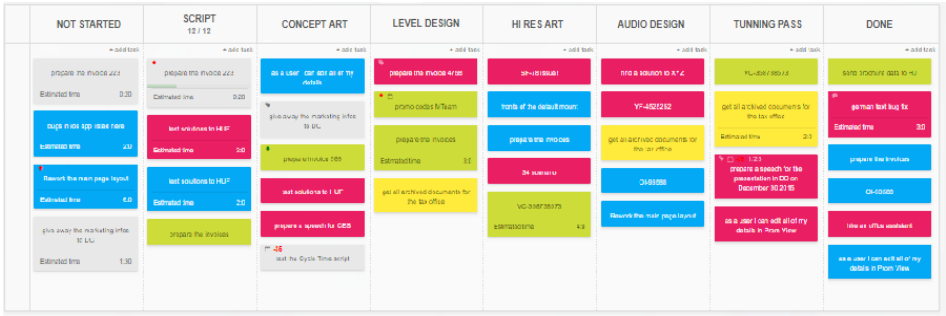


Figure 18. An example of a Heijunka boardfor the sequential software development process [37]

Agile software development techniques like Scrum or Kanban are not inherently different for desktop and IoT applications. Agile's fundamental principles and values remain the same regardless of the type of application being developed. However, there may be certain considerations and nuances specific to IoT applications that need to be considered when applying Agile practices, such as hardware dependencies, connectivity and network considerations, data processing and analytics, user experience and interface (UX/UI) design, and information security.

IoT applications often involve integration with hardware components and devices, introducing additional considerations regarding hardware compatibility, device communication protocols, and firmware updates. Agile teams developing IoT applications may need to coordinate closely with hardware teams and incorporate hardware testing and validation as part of their development process.

IoT applications rely on connectivity to communicate with devices and systems. Agile teams working on IoT applications must consider network reliability, latency, and security aspects related to data transmission and device communication. Testing connectivity scenarios and handling intermittent or unstable network conditions may be specific concerns for IoT development.

IoT applications generate vast data that must be processed, analyzed, and potentially integrated with other systems. Agile teams working on IoT applications may need to incorporate data analytics and processing capabilities in their development process, including Agile practices specific to data engineering, such as data pipeline development, data validation, and data quality assurance.

User interfaces for IoT applications often extend beyond traditional desktop interfaces. They may include mobile apps, web portals, or voice and gesture interfaces. Agile teams must consider the various user interface paradigms and design considerations specific to IoT applications, which may involve

multidisciplinary collaboration with UX/UI designers and specialists in human-computer interaction.
IoT applications often handle sensitive data and are potential targets for security breaches. Agile teams developing IoT applications should prioritize security considerations from the outset, including secure authentication, data encryption, access control, and vulnerability management. Incorporating security testing and addressing security requirements as part of the Agile development process is crucial for IoT applications.
While the core Agile principles and practices remain consistent, the specific context and requirements of IoT applications may necessitate some adaptations and considerations in Agile implementation. Agile teams working on IoT projects should know these unique aspects and collaborate closely with stakeholders, hardware teams, data engineers, and security experts to ensure a successful Agile development process for IoT applications.

7. Conclusion

Software engineering provides the foundation for Industry 4.0 by developing the software systems and infrastructure required for automation, connectivity, data analysis, and cybersecurity. It enables the digital transformation of industries, driving efficiency, innovation, and new business models.
Software engineering provides the essential principles, methodologies, and practices that form the foundation of Industry 4.0. It enables the development of reliable, scalable, and secure software systems that drive automation, connectivity, and data-driven decision-making in various industries. By leveraging software engineering expertise, organizations can successfully embark on their Industry 4.0 journey and unlock the benefits of digital transformation.
The interdisciplinary nature of software engineering reflects the complexity and diversity of challenges faced in developing software systems. Collaboration across disciplines brings together different perspectives, expertise, and approaches, leading to the development of robust, innovative, and effective software solutions. Software engineers must be open to learning from various fields and working collaboratively to address the multifaceted aspects of software development.

9. References

[1] Kipper, L. M., Iepsen, S., Dal Forno, A. J., Frozza, R., Furstenau, L., Agnes, J., &Cossul, D. (2021). Scientific mapping to identify competencies required by industry 4.0. Technology in Society, 64, 101454. https://doi.org/10.1016/j.techsoc.2020.101454

[2] Peres, R. S., Jia, X., Lee, J., Sun, K., Colombo, A. W., &Barata, J. (2020). Industrial artificial intelligence in industry 4.0-systematic review, challenges and outlook. IEEE Access, 8, 220121-220139. https://doi.org/10.1109/ACCESS.2020.3042874

[3] Dalzochio, J., Kunst, R., Pignaton, E., Binotto, A., Sanyal, S., Favilla, J., & Barbosa, J. (2020). Machine learning and reasoning for predictive maintenance in Industry 4.0: Current status and challenges. Computers in Industry, 123, 103298. https://doi.org/10.1016/j.compind.2020.103298

[4] Aheleroff, S., Xu, X., Zhong, R. Y., & Lu, Y. (2021). Digital twin as a service (DTaaS) in industry 4.0: an architecture reference model. Advanced Engineering Informatics, 47, 101225. https://doi.org/10.1016/j.aei.2020.101225

[5] Aceto, G., Persico, V., &Pescapé, A. (2019). A survey on information and communication technologies for industry 4.0: State-of-the-art, taxonomies, perspectives, and challenges. IEEE Communications Surveys & Tutorials, 21(4), 3467-3501. https://doi.org/10.1109/COMST.2019.2938259

[6] Alladi, T., Chamola, V., Parizi, R. M., & Choo, K. K. R. (2019). Blockchain applications for industry 4.0 and industrial IoT: A review. Ieee Access, 7, 176935-176951. https://doi.org/10.1109/ACCESS.2019.2956748

[7] Liu, X. L., Wang, W. M., Guo, H., Barenji, A. V., Li, Z., & Huang, G. Q. (2020). Industrial blockchain based framework for product lifecycle management in industry 4.0. Robotics and computer-integrated manufacturing, 63, 101897. https://doi.org/10.1016/j.rcim.2019.101897

[8] Klingenberg, C. O., Borges, M. A. V., & Antunes Jr, J. A. V. (2021). Industry 4.0 as a data-driven paradigm: a systematic literature review on technologies. Journal of manufacturing technology management, 32(3), 570-592. https://doi.org/10.1108/JMTM-09-2018-0325

[9] de Paula Ferreira, W., Armellini, F., & De Santa-Eulalia, L. A. (2020). Simulation in industry 4.0: A state-of-the-art review. Computers & Industrial Engineering, 149, 106868. https://doi.org/10.1016/j.cie.2020.106868

[10] Margaria, T., &Schieweck, A. (2019). The digital thread in industry 4.0. In Integrated Formal Methods: 15th International Conference, IFM 2019, Bergen, Norway, December 2–6, 2019, Proceedings 15 (pp. 3-24). Springer International Publishing. https://doi.org/10.1007/978-3-030-34968-4_1

[11] Shah, H. (2021) IoT software engineering: The new wave of IoT development, https://www.techtarget.com/iotagenda/post/IoT-software-engineering-The-new-wave-of-IoT-development (available 28.5.2023)
[12] Arduino vs Raspberry Pi, https://peppe8o.com/arduino-vs-raspberry-pi/ (available 28.5.2023)
[13] Reneker, D. and Shaffer, W. (2020) Model-based control: Raspberry Pi vs programmable logic controllers, https://www.controlglobal.com/control/loop-control/article/11297516/model-based-control-raspberry-pi-vs-programmable-logic-controllers (available 28.5.2023)
[14] Scharfglass, K. (2018) Which Wireless Is Right Wireless?https://hackaday.com/2018/10/19/which-wireless-is-right-wireless/ (available 28.5.2023)
[15] Jackson-Barnes, S. (2022) Edge Computing VS Cloud Computing: Differences, Benefits, and Best Practices, https://www.orientsoftware.com/blog/edge-computing-vs-cloud-computing/(available 28.5.2023)
[16] ROS Installation, https://www.ros.org/blog/getting-started/ (available 28.5.2023)
[17] HIWIN robots use ROS on Windows, https://microsoft.github.io/Win-RoS-Landing-Page/hiwin_case_study# (available 28.5.2023)
[18] Human-Machine Interfaces, https://sceta.io/human-machine-interfaces/ (available 28.5.2023)
[19] Angelopoulos, A. et al. (2020). Tackling Faults in the Industry 4.0 Era—A Survey of Machine-Learning Solutions and Key Aspects. Sensors 2020, 20, 109.https://doi.org/10.3390/s20010109
[20] Schmitt, J., Bönig, J., Borggräfe, T., Beitinger, G., &Deuse, J. (2020). Predictive model-based quality inspection using Machine Learning and Edge Cloud Computing. Advanced engineering informatics, 45, 101101.https://doi.org/10.1016/j.aei.2020.101101
[21] Georgakopoulos, D. et al. (2016). Internet of Things and edge cloud computing roadmap for manufacturing. IEEE Cloud Computing, 3(4), 66-73.https://doi.org/10.1109/MCC.2016.91
[22] Cao, K. et al. (2021). A survey on edge and edge-cloud computing assisted cyber-physical systems. IEEE Transactions on Industrial Informatics, 17(11), 7806-7819.https://doi.org/10.1109/TII.2021.3073066
[23] ZhihanLv, Dongliang Chen, Ranran Lou, Qingjun Wang, (2021), Intelligent edge computing based on machine learning for smart city, Future Generation Computer Systems, Vol. 115, 2021, 90-99.https://doi.org/10.1016/j.future.2020.08.037

[24] Bajic, B.et al. (2019). Edge computing vs. cloud computing: challenges and opportunities in Industry 4.0. Annals of DAAAM & Proceedings, 30.https://doi.org/10.2507/30th.daaam.proceedings.120
[25] Wood G. (2023) Introduction to Web3, https://ethereum.org/en/web3/(available 28.5.2023)
[26] Zenlayer (2022) Why Web 3.0's Future Builds on Edge Computing & Blockchain Technology, https://www.zenlayer.com/blog/web-3-edge-computing-blockchain/(available 28.5.2023)
[27] Recorded cyber attacks.https://www.statista.com/outlook/tmo/cybersecurity/worldwide#cybercrime(available 28.5.2023)
[28] Distribution of cyberattacks across worldwide industries in 2022. https://www.statista.com/statistics/1315805/cyber-attacks-top-industries-worldwide/(available 28.5.2023)
[29] Annual number of IoT malware attacks worldwide from 2018 to 2022. https://www.statista.com/statistics/1377569/worldwide-annual-internet-of-things-attacks/(available 28.5.2023)
[30] The ISO/IEC 27000 Family of Standards. https://cfecertification.medium.com/the-iso-iec-27000-family-of-standards-2f04697f2e82(available 28.5.2023)
[31] Gehrke, L.et al. (2015). A Discussion of Qualifications and Skills in the Factory of the Future: A German and American Perspective. VDI White Paper, VDI Verein DeutscherIngenieuree.V. / ASME, https://www.vdi.de/ueber-uns/presse/publikationen/details/industry-40-a-discussion-of-qualifications-and-skills-in-the-factory-of-the-future-a-german-and-american-perspective(available 28.5.2023)
[32] Mooppan, A. (2021). Retrofit IoT. https://www.linkedin.com/pulse/retrofit-iot-vuelogix-technologies-pvt-ltd/?trk=organization-update-content_share-article(available 28.5.2023)
[33] FANUC Retrofit Solutions. https://www.fanuc.eu/ch/en/lifetime-management/fanuc-retrofit-solutions(available 28.5.2023)
[34] Bangamuwage, N. (2020). Sprint flow. https://nerukanadun.medium.com/sprint-flow-8281d9e1ecf5(available 28.5.2023)
[35] Alyakskin, I. (2017). Dynamic Systems Development Method. https://www.luxoft-training.com/news/dynamic-systems-development-method-/(available 28.5.2023)
[36] Kanban. https://www.agilealliance.org/glossary/kanban/(available 28.5.2023)