IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 4 December 2024, accepted 17 December 2024, date of publication 23 December 2024, date of current version 2 January 2025.

Digital Object Identifier 10.1109/ACCESS.2024.3521867

RESEARCH ARTICLE

Enhancing Software Distribution and Collaboration in Manufacturing: Developing the IMOS Platform

ANTÓNIO MONTE PEGADO, ANDRE DIONISIO ROCHA, (Member, IEEE), AND JOSE BARATA, (Member, IEEE)

NOVA School of Science and Technology, Center of Technology and Systems (UNINOVA-CTS) and Associated Lab of Intelligent Systems (LASI), NOVA University Lisbon, 2829-516 Lisbon, Portugal

Corresponding author: António Monte Pegado (al.pegado@campus.fct.unl.pt)

This work was funded in part by Fundação para a Ciência e Tecnologia through the program CTS/00066 and Center of Technology and Systems (CTS).

ABSTRACT In today's fast-evolving tech landscape, small and medium-sized businesses often struggle to keep pace with innovation due to financial constraints and the complexity of integrating new solutions. As the industrial world transitions from Industry 4.0 to Industry 5.0, both manufacturers and software providers are embracing a modular approach to software procurement and deployment. This strategy not only boosts flexibility and affordability but also prioritizes sustainability and the value of collaborative efforts. The Industry Modular Operating System (IMOS) is presented as a platform supporting software accessibility, execution, and integration, while encouraging partnerships between software developers and industrial site managers. IMOS is designed to address recent industry challenges, with a focus on building a solid framework that highlights modularity, facilitates distribution, and promotes cooperation. This system facilitates the distribution of industry-oriented tools by offering containerized software solutions through a unified marketplace. The platform enables industry professionals to efficiently manage diverse applications and services, both on-site and in the cloud, granting access to tailored data and resources. IMOS uniquely extends the concept of Industrial Symbiosis to digital assets, proposing the development of a collaborative forum that values co-creation, facilitates resource sharing, and fosters symbiotic relations. By creating this flexible ecosystem, IMOS closes the gap between development and production, fostering collaboration, innovation, and shared expertise within an active community of industry leaders.

INDEX TERMS Application, cloud, collaborative manufacturing, community, containerization, IMOS, integration, marketplace, symbiosis.

I. INTRODUCTION

The rise of cutting-edge technologies and modularization is reshaping production processes within the evolving industrial landscape. As Industry 4.0 (I4.0) transitions to Industry 5.0 (I5.0), there is an increasing focus on adaptable and sustainable applications, and more collaborative and human-centered manufacturing. But a key challenge remains: *seamlessly integrating and distributing these software applications within manufacturing workflows*.

The associate editor coordinating the review of this manuscript and approving it for publication was Jolanta Mizera-Pietraszko.

As digitalization progresses, the need for scalable and specialized solutions becomes critical, facilitating not only streamlined integration but also enhanced collaboration in response to changing market dynamics. Manufacturers are now seeking sustainable approaches that enable incremental integration. This shift also opens doors for smaller developers to create niche and modular solutions.

However, while software modularity offers greater flexibility and operational simplicity, it also presents challenges, particularly for small and medium-sized enterprises (SMEs). Rapid technological advancements and the short lifecycle of modern solutions can overwhelm small manufacturers,

who often struggle with the high costs and complexities of technology compatibilities. Due to market competitiveness, existing software solutions often lack seamless integration, forcing SMEs to invest heavily in custom-made alternatives.

Efforts to address modularization and sustainability in industry have led to the development of containerized applications for various functions, such as AI-driven quality control and safety prevention. However, a centralized platform for distributing and standardizing these solutions is still missing, limiting collaboration among stakeholders.

This prompts the following research question: *In what ways can a collaborative ecosystem improve the distribution and integration of software within industrial environments?*

To best answer this question and build towards a solution that can meet its requirements the following structured approach is presented:

1) Conceptualization - designing an industry-oriented marketplace framework that builds towards a collaborative ecosystem;
2) Prototype Development - building a proof-of-concept using a lightweight, modular tech stack;
3) Ecosystem Building - iteratively developing tools and features tailored to user needs and feedback, fostering long-term engagement and collaboration;
4) Deployment - scaling the platform for industrial use with enhanced security, standardization, and customization.

Establishing a software marketplace for SMEs in industry could significantly enhance access to emerging solutions and resources. Such a platform would promote integration through modularity and standardization, and at the same time, provide developers with greater visibility and access to valuable data. In turn, this solution can leverage its distinct user profiles and beneficiaries by fostering resource-sharing, co-innovation, and collaborative initiatives, bringing together development, research, and production stakeholders.

The Industry Modular Operating System, introduced in this paper, provides a comprehensive platform for modular software distribution, integration, and orchestration. More than just a technological solution, it embodies a paradigm shift by applying the principles of Industrial Symbiosis to digital ecosystems, promoting co-creation, innovation, and collaborative initiatives.

The work being presented lays the foundation for this vision, with the main contributions summarized as follows:

1) Overview of industrial software marketplaces, collaborative ecosystems, and modular applications;
2) Conceptual framework for modular software distribution and industrial symbiosis;
3) Implementation results of the proposed platform, including application use cases;
4) Public release of platform advancements and use cases to promote further research.

The remainder of this paper is organized as follows: Section II reviews key concepts and related work, followed by Section III, presenting the proposed conceptual framework, and Section IV detailing the platform's architecture. Section V discusses the implementation and technology specifics. Afterwards, Section VI covers the results of use case examples, followed by conclusions, limitations, and future research directions in Section VII.

II. RELATED WORK

This section offers a concise review of relevant literature, focusing on the key concepts and methodologies that underpin the proposed framework. These insights establish a solid foundation for understanding the current state of software distribution in the context of smart manufacturing, paving the way for the project's development.

A. INDUSTRIAL MARKETPLACE CONCEPT

Smart manufacturing is driving significant changes in technology adoption and system sustainability. Traditional models of acquiring custom-made, rigid solutions are being replaced by more dynamic and adaptable paradigms.

Gröger et al. [1] introduced the concept of app-based manufacturing, envisioning a unified platform in which users can design task-specific tools for various production levels. They propose a marketplace, similar to mainstream app stores like Google's Play Store or Apple's App Store, where manufacturing apps can be developed, acquired, and submitted by users across different companies. However, the lack of a unified platform and standardized protocols remains a significant barrier to the widespread adoption of this approach.

The marketplace concept aligns with emerging principles, offering hyper-customization through software applications that address key areas like predictive maintenance, robot collaboration, and augmented reality integration. Unlike traditional outsourcing, an open software ecosystem allows multiple developers to contribute to a product, facilitating a collaborative approach to manufacturing systems [2].

One example is Siemens' Insights Hub [3], [4], an industrial IoT application suite that spans edge to cloud, enhancing adaptability and integration. By gathering data from products, systems, and facilities, the platform fuels innovation in business models, streamlines operations, and facilitates advanced analytics and AI-driven capabilities via distributed applications. However, its closed-platform nature limits broader academic exploration and open innovation.

Akula et al. [5] propose a comprehensive framework featuring an App Runtime Management, a crucial capability that allows developers to create, add, publish, and delete programs within a marketplace platform. This management system not only enables customers to easily acquire and execute applications but also allows these apps to serve as foundational blocks for the development of new applications.

Another example is vf-OS, a Virtual Factory Open Operating System supported by the H2020 Programme, which tackles industrial digitalization challenges through edge computing and modular, extensible components. This project decentralizes computing, storage, and networking

resources, bringing them closer to data sources. Within the vf-OS ecosystem, a modular and extensible stack of components, as presented by Anaya et al. [6], enhances the core operating system, fostering a dynamic marketplace of virtual applications (vApps) and vf-OS services. The ecosystem leverages a Service-Oriented Architecture (SOA), allowing components to implement and publish REST interfaces, facilitating seamless data exchange. Its assets, built around Docker images, are available in the vf-OS marketplace, streamlining application integration into the execution environment [7], [8].

Similarly, J. Giao et al. propose a SOA that integrates and standardizes enabler agents - open-source software modules designed for cross-domain functionality [7]. The Enabler Framework uses a single REST API to connect applications with enablers, facilitating seamless integration and dynamic communication with sensors, actuators, and data storage. This architecture enhances the interaction between applications and physical systems, streamlining the integration of new or updated software modules.

In marketplace ecosystems, addressing interoperability challenges is crucial. Adopting a standardized approach from the start simplifies the integration of various software applications and services. A key strategy involves utilizing a reference architecture aligned with industry standards and methodologies. This architecture acts as a blueprint, providing a standardized framework for integrating software, devices, and data [9], [10].

Two notable projects in this area are ZDMP and KITT4SME. ZDMP [11] (Zero Defects Manufacturing Platform) aims to enhance process and product quality by reducing errors through preventative, corrective, and predictive methods. Using data-driven technologies, it prevents defective items from leaving production sites, improving manufacturing sustainability. A key enabler is software interoperability, supporting seamless integration for defect detection, prevention, and repair [12], [13].

KITT4SME [14], funded by Horizon 2020, supports European SMEs and mid-caps with modular AI solutions tailored to production systems. Its platform integrates with factory infrastructure - MES, ERP systems, IoT devices, and robotics - to drive innovation. Built on the FIWARE framework, it ensures scalability, leveraging ecosystems like the RAMP [15] marketplace. Partnerships with digital innovation hubs and open calls further advance AI adoption, enhancing competitiveness in manufacturing [16], [17].

B. COLLABORATIVE ECOSYSTEMS AND SYMBIOSIS

On a different topic, the authors discuss sustainable manufacturing as a central piece to the proposed marketplace ecosystem. This concept differs from traditional approaches by incorporating the triple bottom line framework, which evaluates environmental, economic, and social dimensions simultaneously.

Abubakr et al. [8] highlight key challenges in implementing sustainable manufacturing within collaborative and intelligent networks, including the need to address safety, privacy, and ethical concerns, as well as bridging the gap between industrial practices and academic research. The complexity of these advanced systems requires strong economic justification, and integrating new smart, sustainable systems with existing infrastructures poses significant compatibility challenges.

Focusing on collaboration and sustainability, industry can bridge these gaps, fostering innovation and creating manufacturing processes that are both efficient and environmentally responsible. The authors of [8] suggest a holistic approach that has the potential to ensure long-term viability and integrity in the manufacturing landscape.

From the Circular Economy (CE) perspective, Industrial Symbiosis (IS) is conceptualized by [18] as a business model archetype that enhances resource efficiency and creates value from waste through shared use of infrastructures and by-products. The foundation of a circular business rests on three pillars: technical innovation, collaboration, and a sustainable business model. Technical innovation in IS involves exchanging waste, resources, and energy across multiple production processes, optimizing resource use, minimizing waste, and contributing to a CE. On the other hand, collaboration is crucial, requiring the identification and cooperation of various stakeholders to ensure the successful implementation and operation of an IS cluster.

Moreover, the authors of [9] argue that embracing system modularity and leveraging cutting-edge collaboration tools enables smaller developers to focus on specialized projects. This strategy provides SMEs with a broader array of solutions, enhancing their competitiveness against larger companies in the industry. Consequently, this method fosters a more sustainable, competitive, and innovative environment for all participants.

C. MODULAR SOFTWARE APPLICATIONS

The contemporary manufacturing sector is transitioning from mass production to mass customization, focusing on tailoring both products and processes to meet individual customer needs. However, it faces significant challenges due to poor interoperability among various original equipment and software manufacturers, which complicates seamless integration into manufacturing systems. This challenge underscores the need for modular applications that align with architectural standards, as current monolithic approaches hinder system modularity and integration capabilities.

Torayev et al. [10] investigate the modularization of industrial systems using software, with a particular emphasis on containerization and microservices technologies. The suggested approach entails creating a "*plug-and-produce*", modular and interoperable manufacturing app architecture as well as a manufacturing app development kit. The idea

of "*appification*" makes it easier to distinguish between physical and digital implementations, allowing for modular and reusable solutions. This system, managed via a NodeJS web interface, uses Docker for containerization, with potential for more complex orchestration using other technologies like Kubernetes.

A similar, yet cloud-centered approach is the architecture presented by [19] that involves deploying virtual images of devices constituting the distributed automation system as containers in the cloud. This implementation method of representing both physical and software assets offers a great understanding of technologies and tools to achieve a modular system. A cloud-based virtual commissioning website facilitates system configuration, offering off-the-shelf virtual assets and allowing users to customize configurations without extensive engineering work.

Goldschmidt et al. [20] provide an architectural example that underscores the importance of implementing container-based solutions for industrial control as a foundation for future automation system architectures. The architecture they propose for a versatile controller leverages lightweight container technologies such as LXC or Docker. These containers, characterized by operating system-level virtualization, allow the hosting of multiple segregated user-space instances within the operating system kernel. This approach ensures the isolation of applications, preventing them from accessing each other's resources, while the container host offers flexibility in managing resource usage, including CPU, memory, disk I/O, and network. The paper highlights the dynamic nature of containers, which can be added, removed, started, and stopped based on evolving system requirements, making them highly adaptable to changing operational demands.

D. GAP ANALYSIS

Despite these advancements, the industry faces significant challenges. A major obstacle is the lack of robust software distribution solutions that can keep pace with the rapid advancements in technology. Furthermore, the complexity and rigidity of existing systems make it difficult to seamlessly integrate new technologies into existing workflows. In a competitive market, the intricate nature of required solutions often discourages stakeholders from pursuing collaborative efforts.

A review of scholarly literature revealed limited research on the topic, despite the existence of a few prominent projects in this area. This gap suggests that the subject might represent potential for research or an underexplored field, particularly in leveraging a marketplace to foster collaborative manufacturing environments. Existing studies indicate that industrial marketplaces are effective and that generic modular software systems represent a growing trend. This insight drives our goal of merging these two domains to develop an integrated solution tailored to the needs of industry, particularly SMEs.

This research aims to develop a framework that addresses these challenges by simplifying software distribution and system integration, while promoting collaboration among industry stakeholders and fostering more sustainable industrial practices.

III. THE CONCEPTUAL FRAMEWORK

To address the needs and challenges identified in the literature, this work proposes the creation of the *Industry Modular Operating System (IMOS)*. The IMOS framework aligns with core industry principles, enabling users to adapt, integrate, reuse, and upgrade processes sustainably. This chapter emphasizes software modularity, integration, and the collaborative aspects of co-innovation and industrial software symbiosis, aiming to provide a robust, adaptable solution for modern industry demands.

IMOS is an operating system built on top of an Open Containerization Platform (OCP) designed to facilitate modular software integration and acquisition in industrial environments. The platform provides built-in applications and tools customized to meet the diverse needs of various industry stakeholders, acting as a unified solution that bridges different sectors (Fig. 1). IMOS enables users to construct their manufacturing systems step by step, adding applications as needed, promoting collaboration and resource sharing.

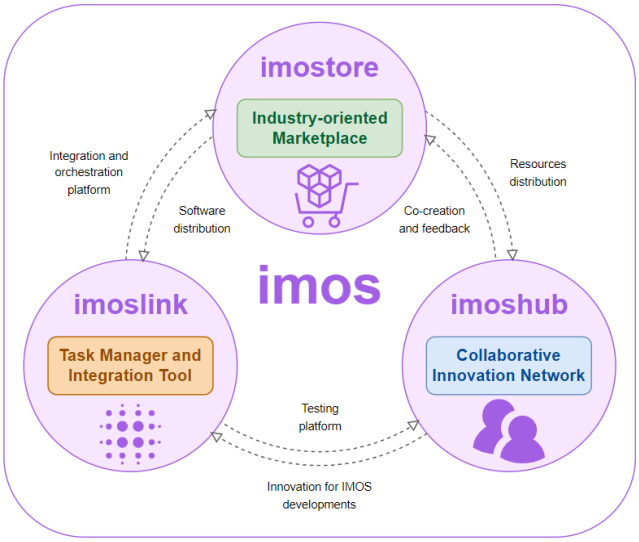


FIGURE 1. IMOS modules, including the three in-built applications supporting software integration, distribution, and co-creation.

The system comprises three interconnected modules:

1) *IMOStore*: An industry-specific marketplace for distributing containerized applications and assets addressing specific industry needs, supporting both manufacturing and software stakeholders, while ensuring compliance with IMOS integration standards. Users can access a wide range of resources, including software applications, libraries, AI models, and datasets. Developers can publish, monetize, and

collaborate on solutions, fostering the creation of innovative, modular software systems;
2) *IMOSlink*: This module automates application execution and software integration, centralizing all IMOS assets and allowing users to easily manage and orchestrate them. It serves as a specialized operating environment that simplifies decision-making and process orchestration, enhancing usability and efficiency within the IMOS framework;
3) *IMOShub*: A community-driven platform within IMOS that addresses industry concerns by fostering collaboration between software developers and industrial engineers. It supports innovation by facilitating feedback, data exchange, and joint efforts for integrated solutions and co-creation.

Additionally, *IMOScloud* offers a cloud-based service that manages verified applications published on IMOStore. It allows users to create, start, or stop applications on the cloud, mirroring the functionality of IMOSlink but extending it to cloud-based operations.

Together, these modules work seamlessly within the IMOS ecosystem. IMOSlink and IMOScloud provide platforms for integrating new modules and applications. IMOStore facilitates the distribution of software and resources, while IMOShub serves as a forum for co-creation and industry collaboration. Through this interconnected framework, IMOS enhances its effectiveness in meeting the evolving goals of industry.

A. COLLABORATIVE INNOVATION

Advances in networking and technology have greatly enhanced collaborative efforts, driven by the rise of computer networks and emerging platforms. Camarinha-Matos and Afsarmanesh [21] refer to this evolution as Collaborative Networks (CNs) 4.0, which share similar objectives with the IMOS ecosystem. CNs 4.0 are characterized by their ability to handle large volumes of data, generate revenue from collaboration, and co-create value through innovative business models. - the development of the IMOS platform has been aligned with these objectives.

Furthermore, the UN Agenda 2030, which delineates 17 Sustainable Development Goals (SDGs), underscores the significance of ''building resilient infrastructure, promoting inclusive and sustainable industrialization, and fostering innovation in manufacturing'' [22].

CNs face challenges in data-rich environments, necessitating a rethinking of design assumptions and the creation of new architectures. This need underscores the importance of higher integration levels, requiring collaboration across various knowledge domains and the development of new platforms [23]. However, data abundance also raises security, access, and quality concerns, making it essential to verify and monitor platform users.

The co-creation process in CNs enables diverse stakeholders to create value and resources, fostering innovation. Value co-creation is a business model centered on active interaction between consumers and producers, allowing consumers to take on roles akin to those of employees or designers [24]. This model fosters new synergies, enabling consumers as well as producers to improve their outputs.

Esposito De Falco et al. describe a collaborative digital platform as a ''stable, centralized core of a distributed innovation network'', designed to promote collective, synergistic, and decentralized advancements while capitalizing on future market opportunities and leveraging external creativity [25]. These systems build up knowledge and value over time by keeping ideas in the cloud for convenient access, allowing for continual long-term value generation. Smorodinskaya et al. go on to describe these creative ecosystems as ''dynamic collaborative networks of people and organizations formed around projects with an innovation objective'' [26].

An essential concept in collaborative ecosystems and sustainable manufacturing is industrial symbiosis, where the waste or by-products from one sector become valuable resources for another. This approach moves away from the traditional linear model, transforming waste into inputs for different processes and fostering a circular economy.

Industrial symbiosis promotes collaboration between typically independent entities through resource sharing, resulting in greater sustainability with environmental, economic, and social benefits [27]. This work [9] explores key initiatives that connect product life cycles to collaborative frameworks, creating standards that strengthen support for the circular economy and industrial symbiosis. These ideas have played a crucial role in shaping the IMOS framework, addressing the research challenges effectively.

Resource sharing promotes collaborative innovation within software development. Platforms like IMOS serve as a link between producers and consumers, facilitating feedback, knowledge sharing, and efficient problem-solving. *The authors also envision industrial symbiosis extending beyond physical resources to include the exchange of software modules and manufacturing data that might otherwise go unused.*

B. IMOS FRAMEWORK IN CNs 4.0

A key aspect to building the proposed ecosystem is offering collaborative tools and incentivizing innovation, while enhancing the marketplace and software distribution.

IMOShub, the community forum within the IMOS platform, can be categorized as a Collaborative Innovation Network (CIN), which falls under the broader categories of Virtual Organizations Breeding Environments (VBEs) and Professional Virtual Communities (PVCs). This classification is due to its inclusive membership, accommodating both organizations/enterprises and individuals/freelancers [28]. IMOShub facilitates the formation of collaborative groups, such as Virtual Organizations (VOs) or Virtual Teams (VTs), enabling them to work toward shared goals or joint projects through computer networks.

The success of a collaborative project is often shaped during the initial planning and construction of a CN. To support this critical phase, the authors designed IMOShub as a platform and tool to facilitate the early stages of CN life cycles. IMOShub aims to foster early connections, discussions, and experience exchanges, while also streamlining the process of partner search and selection. The framework presented in Fig. 2 is part of the authors' previous work [29] and intends to bring entities together in a shared space, allowing them to combine their competencies, share resources, and collaborate for shared objectives.

Most CNs backed by IMOShub are motivated by specific opportunities and focus on distinct projects. This frequently results in the creation of VOs or VTs, typically comprising manufacturing companies (consumers) seeking solutions, along with one or more software stakeholders (producers) developing applications to fulfill those needs. IMOShub serves as a bridge between these groups, facilitating communication and resource sharing.

Through the operational phase, IMOShub aims to support community-focused and knowledge-sharing spaces. It also enables the publication of open-source datasets and other resources that developers can leverage to create solutions. Collaboration within groups is enhanced by providing repositories and workspaces that streamline data exchange, programming tools, and communication.

During the dissolution phase of CNs, IMOShub emphasizes optimizing value and gathering user feedback to enhance the CIN within the IMOS ecosystem. This involves fine-tuning collaboration tools and enhancing the user interface by leveraging insights obtained from user interactions and CN outcomes.

Within the IMOS ecosystem, IMOShub operates in tandem with IMOSlink and IMOStore, forming a cohesive approach that streamlines collaboration efforts. IMOS brings together industry stakeholders from diverse backgrounds, roles, and interests, offering tools for software distribution, asset integration, and orchestration. This integrated approach enables users to effortlessly connect, share resources, and work together towards common goals, all within a platform that provides additional benefits.

IV. IMOS PLATFORM

IMOS was developed to align with previously mentioned industry goals, emphasizing the creation of a more human-centric, resilient, and sustainable industrial environment. By addressing the challenges identified in previous research, IMOS was conceptualized to deliver several key benefits that are critical for the modern manufacturing landscape:

- *Integration and Customization*: Manufacturers can select from a diverse range of containerized applications that integrate seamlessly with IMOSlink, allowing for customization and optimization of automation systems. This enhances efficiency, modularity, and ensures a cohesive system performance;
- *Distribution and Monetization*: IMOStore simplifies the access, deployment, and sharing of containerized applications across different environments, reducing the complexities and costs associated with traditional software distribution. Software providers can monetize their applications, while companies can purchase only what they need, minimizing expenses;
- *Standardization and Synergy*: Containerization ensures that software applications adhere to standardized formats, streamlining integration and compatibility across various manufacturing systems and fostering synergy through IMOStore;
- *Community and Co-Creation*: IMOS fosters a collaborative community where stakeholders contribute to the platform's growth by providing feedback, discussing industry challenges, and proposing solutions via IMOShub, encouraging innovation and co-creation through shared knowledge and resources;
- *Industrial Symbiosis and Feedback Loop*: IMOS promotes resource and knowledge sharing across different industrial sectors, leading to industry-specific solutions and greater efficiency in development. Feedback from application usage is fed back to developers, enabling continuous improvement and adaptation of software based on real-world needs;
- *Open-Source Flexibility*: The open-source, modular nature of IMOS allows users to modify, enhance, and create their own modules, fostering a collaborative environment where the platform evolves through community contributions - a platform *built by the industry, for the industry*.

The developed platform lowers the barrier to advanced software adoption by offering modular solutions that can be incrementally implemented. This reduces upfront costs and minimizes vendor lock-in. Additionally, IMOS promotes resource-sharing, proposing a collaborative module with collaborative tools, akin to shared manufacturing ecosystems.

IMOS tackles the challenges faced by SMEs by emphasizing accessibility, integration, and collaboration. With this foundation in place, the next chapter will explore the marketplace framework and platform architecture that drive IMOS, detailing how its core components come together to create a cohesive and reliable ecosystem.

A. MARKETPLACE AS AN ENABLER OF IMOS

The presented framework in Fig. 3 includes a software marketplace that was central to the development of IMOS. This marketplace provides a diverse selection of containerized software applications. and other assets to support the creation of new solutions. It also serves as a community space where engineers, manufacturers, developers, and academics can collaborate to solve emerging problems by sharing knowledge and resources.

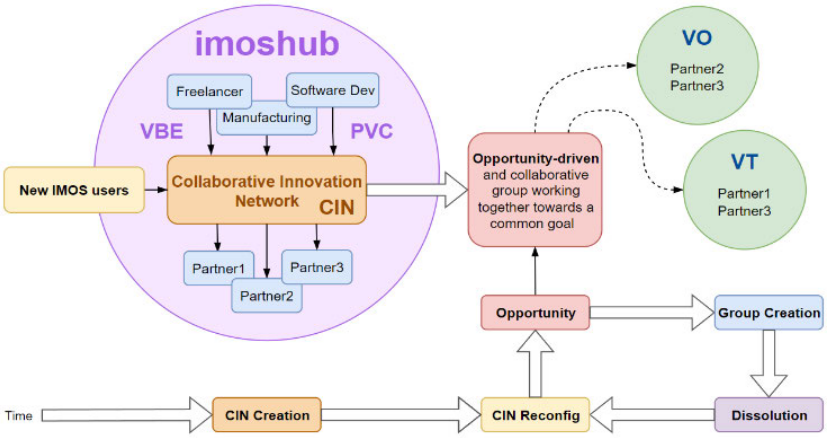


FIGURE 2. IMOShub described as a Collaborative Innovation Network. The platform is envisioned to enable the creation of VOs and VTs, from [29].

The marketplace primarily benefits two groups: manufacturers (SMEs), who gain access to IMOStore solutions, and software providers (DEVs), who distribute their applications while leveraging the collaboration and data resources provided by the IMOShub community.

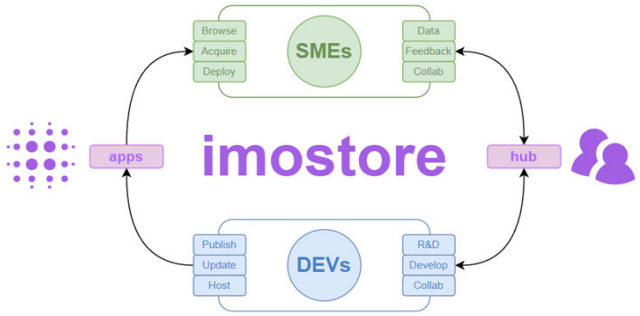


FIGURE 3. IMOStore marketplace concept representation, facilitating software distribution between users, and bridging the gap between production and development.

Manufacturing companies access IMOStore, a digital hub offering a wide range of applications and services that ensure seamless compatibility across diverse manufacturing environments. Clients can browse the marketplace to discover and choose software applications that best suit their unique operational requirements.

Software providers are integral to this ecosystem. Developers create and containerize their applications, ensuring compliance with marketplace standards for interoperability. These applications are then showcased in IMOStore, where they gain visibility and can be easily accessed by potential clients. This approach not only facilitates distribution but also supports ongoing development efforts by creating a cohesive, modular ecosystem.

IMOStore also functions as a community hub where SMEs and developers collaborate. Users provide feedback, share data, and exchange experiences to enhance development. Manufacturers identify industry needs, review app performance, and share data, which developers can use to refine existing solutions or develop new ones, fostering continuous innovation.

For example, a small manufacturing company seeking a safety system with Personal Protective Equipment (PPE) detection could *build its solution incrementally*, selecting modules from different providers at its own pace. *This modular approach results in a more affordable, scalable, and sustainable solution than an all-in-one system.*

IMOStore allows small software publishers to prosper. Developers that focus on narrow, niche solutions frequently find themselves cooperating, since their apps may complement one another. IMOStore raises the exposure of their products while leveraging IMOS' flexibility to streamline distribution and encourage interoperability. Furthermore, manufacturers may exchange real-world data, allowing developers to constantly improve their work.

B. IMOS ARCHITECTURE

To support the planned marketplace, a solution was required to integrate and orchestrate the acquired resources and apps depending on user preferences. The goal was to create a core platform that could integrate, execute, and orchestrate containerized assets while hosting built-in apps and tools.

IMOS was built as the central unit (parent) responsible for managing all other programs (childs), facilitating communication through Inter-Process Communication (IPC) and executing process management commands. The design enables containerized apps to integrate into manufacturing procedures via an *enabling agent*.

Fig. 4 depicts the IMOS architecture consisting of three separate modules (purple) that may be scaled to integrate a variety of additional software applications. IMOS operates in parallel with the local OCP (blue), integrating into the presented architecture and running additional modules as containers. The IMOS server (green) manages user information, authentication, marketplace operations, and application execution through API requests. The infrastructure is made up of two micro-services: one integrating two

separate clusters for user data administration and marketplace management, and the other dedicated to executing applications as containers on the cloud OCP (blue).

Although IMOShub was previously considered and envisioned, the authors chose not to incorporate it in the prior design since it has not yet been implemented. Further developments are needed, focusing on studying and enhancing user experience, and defining collaboration tools.

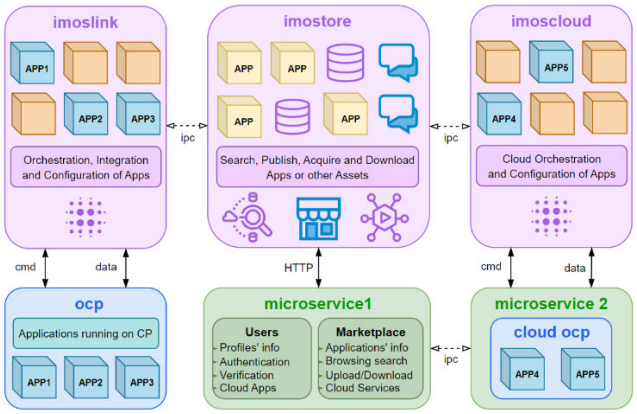


FIGURE 4. IMOS implemented architecture including IMOSlink, IMOStore and IMOScloud, the local OCP, and the server infrastructure.

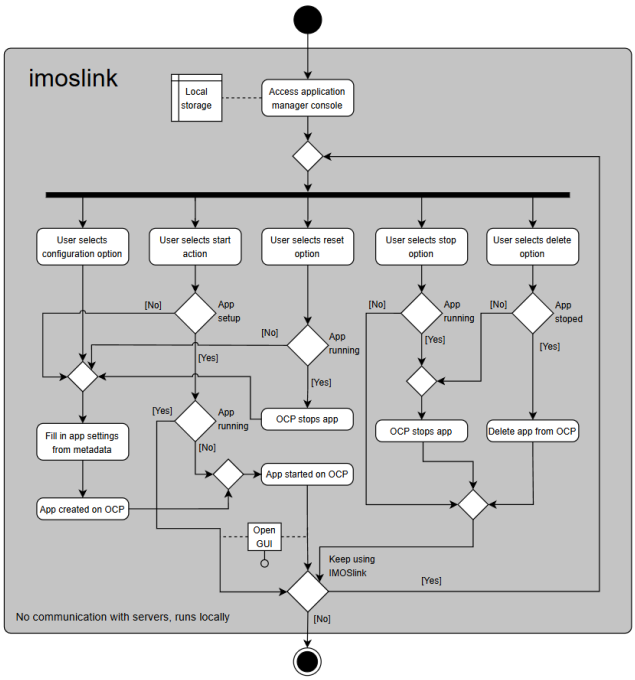


FIGURE 5. Activity diagram representing IMOSlink's flow of execution and OCP managing functionalities.

As depicted in Fig. 5, IMOSlink operates locally, functioning both as a container manager and an integration tool. When accessing IMOSlink, users can configure new applications by setting the correct inputs, settings, and endpoints as described by the app's metadata. They can also start previously configured modules, reset task executions, stop applications, or delete any module.

With IMOSlink, users can either integrate their own developed applications (adhering to integration standards) or acquire solutions from IMOStore, allowing for configuration and connection to various endpoints. IMOSlink acts as the intermediary between an Open Collaborative Platform (OCP) and the user, utilizing its features for smooth integration, distribution, and modularity. This creates a cohesive system of interconnected containers with the user as the central orchestrating unit.

IMOScloud, on the other hand, was introduced as a comprehensive cloud computing solution that prioritizes providing execution environments for certified and trusted applications available on IMOStore. It mirrors the orchestration capabilities of IMOSlink, utilizing endpoints provided by the IMOS server as commands for managing the execution of containers on the cloud OCP.

Users can acquire their desired applications and choose to run them on the cloud without needing local installation or execution. This approach ultimately enables users to run applications both on-premise and on-cloud, efficiently leveraging cloud-based resources while retaining the flexibility to customize the system's executing modules.

Although developing a platform like IMOS requires significant investment in infrastructure, implementation efforts, and training, the platform itself is intended to be sustainable and monetizable. Users, including SMEs, are not required to invest in building the solution itself or the underlying infrastructure. Instead, users can access the platform through a possible fee-for-use model or by monetizing their own assets.

The IMOS platform provides several built-in modules that, by themselves, offer value to different stakeholders, meaning that users would organically be drawn to the platform. Over time, as more users engage with the available tools, it is hypothesized that a collaborative ecosystem will emerge. Some beneficiaries may even contribute to the platform's development, aligning with the authors' goal of fostering an open-source and community-driven initiative.

V. IMPLEMENTATION AND RESULTS

This chapter details the development of IMOS, covering the technologies used, challenges faced, and final platform outcomes. The implementation process began with selecting a technology stack best suited to developing the IMOS ecosystem as a proof of concept. Table 1 outlines the chosen technologies and their roles, chosen for their ability to support IMOS's modular architecture, integration, and scalability. A key aspect of the proposed solution is its technological abstraction, offering flexibility for implementation with any technology stack.

The IMOS prototype leverages emerging technologies like NodeJS, Electron, Docker and MongoDB for their modern and scalable capabilities, ideal for rapid prototyping.

NodeJS was selected due to its asynchronous, event-driven architecture, and non-blocking I/O model, which

is particularly well-suited for microservices and modular systems. Comparatively, traditional frameworks like Django, Ruby on Rails, or Spring Boot limit real-time scalability because of their synchronous architectures. Electron was employed to build a modular cross-platform desktop application interface, integrating HTML, CSS, and JavaScript. While alternatives such as C# or Flutter provide powerful tools for native app development, they often come with steeper learning curves.

As for the database implementation, MongoDB was chosen for its flexibility and ease of use, allowing data lake storage without complex relationships. In the future, relational databases like PostgreSQL or MySQL can complement MongoDB, enabling complex queries while retaining efficient access to unstructured data.

Regarding the containerization platform, Docker was selected for its compatibility across diverse environments and wide adoption in the field. While alternatives like Kubernetes or LXC offer more advanced orchestration and control capabilities, they are more suitable for large-scale enterprise deployments, being identified as a transition in future work.

The depicted technology stack led to a successful implementation of the platform prototype, as the selected technologies and approach did not require any reformulation.

The platform was created in a Windows environment using NodeJS with Electron to structure IMOS, and Docker for containerization, streamlining both application management and scalability. The backend is powered by an Express API and a MongoDB database, offering a straightforward yet efficient solution for data management.

Fig. 6 presents the IMOS project's component diagram, offering a comprehensive view of the system and its core components. At the highest level of abstraction, IMOS acts as the parent system for both IMOStore and IMOSlink, along with all installed applications in the application repository.

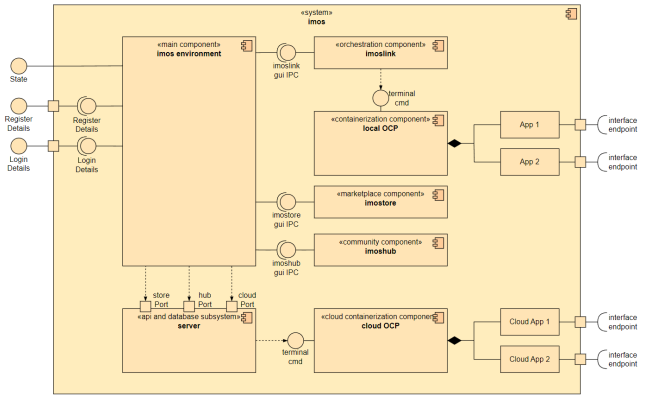


FIGURE 6. IMOS component diagram describing the implemented project's structure and its components' hierarchy.

The primary IMOS interface initially presents users with either the authentication console, prompting them to register or log in, or the desktop environment, which provides access to launch various child components and their associated elements. Once logged in successfully, users gain full interaction capabilities with any available module or application, as IMOS creates a direct IPC connection to the docker.js component, acting as middleware to facilitate access to the application repository.

The platform is designed emphasizing modularity and scalability to support IMOStore and enable the integration of additional modules. As shown in Fig. 7, the IMOS desktop environment, developed using NodeJS and Electron, includes several built-in applications such as IMOSlink, IMOStore, IMOShub, and Settings, alongside use-case apps like the Recycling App, Safety PPE App, Data Vision App, and Roboflow App. This unified interface consolidates all

TABLE 1. List and description of the selected technology stack.

Technology	Version	Specification
NodeJS	20.11.1	NodeJS is the environment base for the IMOS platform, offering an event-driven architecture that supports scalable and flexible backend a frontend tools. Its efficiency in handling multiple connections makes it ideal for a low-code solution prototype of IMOS [30].
Electron	28.1.0	Electron is used to build the cross-platform desktop application interface for IMOS (parent) and its in-built applications (childs). Electron ensures modularity to NodeJS, allowing it to create a window-like OS, serving as a prototype version [31].
Docker	26.0.0	Docker was chosen for its widespread use across many sectors. It provides application packaging to ensure consistency in any environment. It also facilitates application distribution and integration, while leveraging different technologies and coding bases. And supports IMOS's modular architecture through the execution of various applications and services [32].
Express	4.18.2	Express is the web application framework for IMOS's backend API. It simplifies the development of scalable web services, handling routing, middleware, and HTTP requests, and facilitating communication between the IMOS client applications and the MongoDB database [33].
MongoDB	6.3.0	MongoDB, a NoSQL database, is used for its simplicity and scalability through the integration with a data warehouse solution. Its document-oriented structure efficiently manages the data models of IMOS, handling user information, marketplace assets, application states, and other data [34].
NSIS	3.9.0	Nullsoft Scriptable Install System creates the installation package for the IMOS desktop application and the developed application use cases, simplifying the setup process. It supports advanced scripting for customized installations, ensuring a correct installation on various operating systems [35].

The depicted technology stack led to a successful implementation of the platform prototype, as the selected technologies and approach did not require any reformulation.

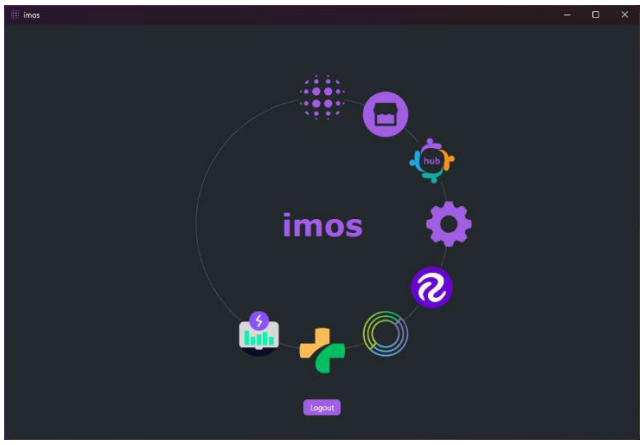


FIGURE 7. IMOS desktop environment, showcasing in-built application modules and various locally installed use-case apps.

modules to simplify user interaction, with IMOSlink managing IMOS-compatible applications via Docker, ensuring seamless integration and functionality across the platform.

Before detailing each module's implementation, Fig. 8 presents a sequence diagram outlining user interactions with IMOS. Divided into three lanes - IMOStore, IMOSlink, and IMOScloud - it illustrates how components interact with the user and each other.

All sequences begin with user authentication, where the server validates access and retrieves user-specific data.

In the *IMOStore sequence lane*, the user interacts with the marketplace, searching for new applications, adding assets to their library, and downloading and installing apps on IMOS and the local OCP.

The *IMOSlink sequence lane* details operations on the locally running module. Users can manage all sorts of applications, provided they meet integration standards. Actions include configuring, starting, stopping, or deleting apps, with commands sent to the local OCP, updating the app's status in IMOSlink.

Lastly, the *IMOScloud sequence lane* represents the cloud-based counterpart of IMOSlink. Integrated within the IMOStore interface, this module allows users to manage any application acquired from the marketplace. However, IMOScloud only runs certified IMOStore apps, with user commands sent to the server for execution on the cloud OCP, and the status updates reflected in the provided interface.

A. IMPLEMENTATION OF IMOS MODULES

IMOStore was the first module implemented in IMOS. Its responsive front-end provides a user-friendly interface inspired by mainstream marketplaces, as shown in Fig. 9. Users can access news, featured applications, and highlights from the home page, while a side panel offers links to the applications, and library pages. Depending on their role as a client or developer, users have a personalized experience tailored to their needs.

Registered developers see additional options for software submission and tracking review status. Each app card displays key details like version, requirements, and ratings. Developers can submit applications by providing detailed information, which is then reviewed by IMOS' validators for compatibility and quality. After review, developers receive feedback on any required changes. Once approved, applications are distributed via the platform. MongoDB's GridFSBucket is used for file storage, ensuring access and deployment of containerized apps directly from IMOStore.

IMOSlink, built as the second Electron child, acts as a bridge between IMOS and Docker, enabling the administration and orchestration of containerized applications. The *docker.js file serves as middleware, handling Docker metadata retrieval and executing commands* like starting, stopping, configuring, and deleting containers, through IPC and command requests. It ensures efficient container management in the IMOS environment, with the detailed processes available on the open-source project's GitHub repository [36].

This enabling agent serves as the central unit for managing and orchestrating user applications within IMOS, providing an interface for facilitating user interaction, as showcased in Fig. 10. IMOSlink launches a child setup console for configuring each application and properly integrate them.

IMOScloud is hosted on IMOStore's library page (see Fig. 11.) The main component is located on the server API, communicating with Docker via the serverDocker.js middleware. Similar to IMOSlink, IMOScloud handles application management through IPC, executing tasks on the Cloud OCP, which has full access to IMOStore's repository.

IMOScloud and the IMOS servers are deployed on an Ubuntu Server to optimize resources and improve computational efficiency. For security and trust, only published apps from IMOStore can be run on IMOScloud. When an app is published, its files are instantiated on the server. When executed, cloud apps are created with a unique name, distinguishing user-specific containers for better resource management.

Finally, the authors detail the *packaging and installation of applications*, handled using NSIS. The process encompasses several steps to ensure proper integration with IMOS:

1) *Creating Dockerfiles*: Each app requires a specific Dockerfile, whether it uses a single or multi-image. The app's source files should be saved in a folder named imos-≪appname≫, and Dockerfiles should be named Dockerfile. ≪imagename≫;
2) *Setting Dockerfile Labels*: Developers must define required labels, such as:
    a. com.available.configs=WEBCAM_IP, PORT (available configuration settings);
    b. com.required.configs=WEBCAM_IP (required user input for app configuration);
    c. com.user.display=False (defines if the app has an interface to display);
    d. com.main.multicontainer=imos-≪appname≫ (necessary multi-container apps).

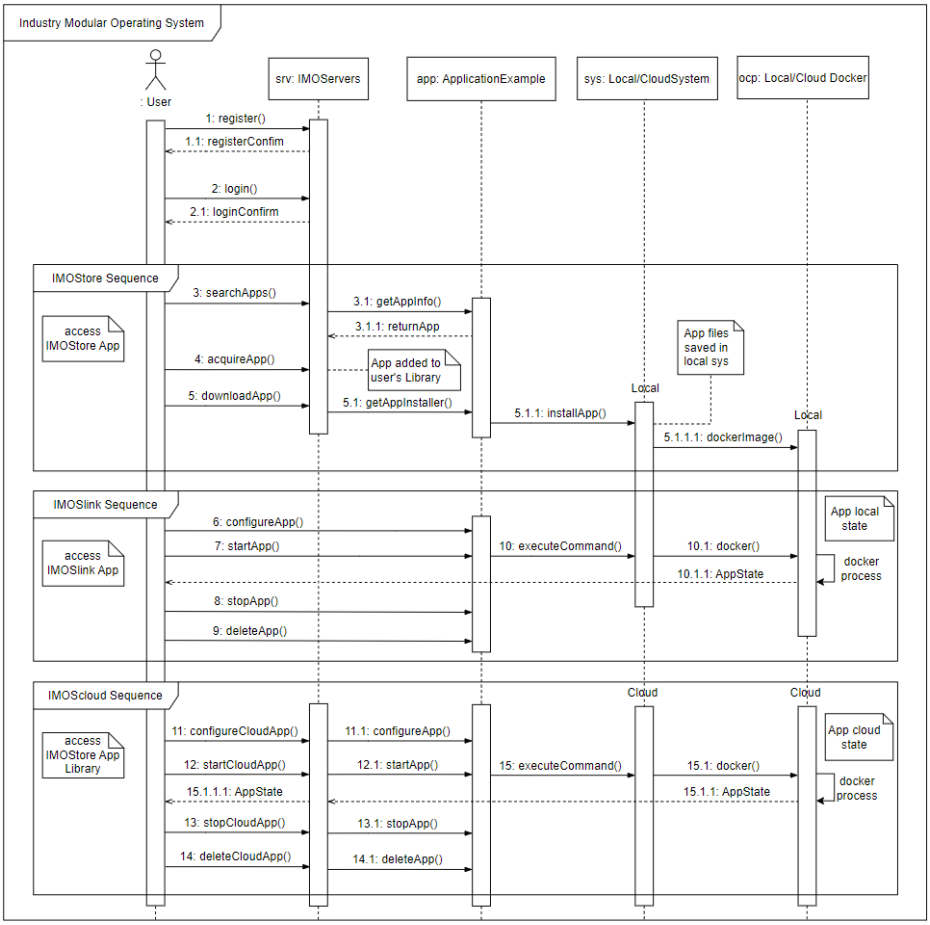


FIGURE 8. IMOS sequence diagram consisting of three lanes, each representing user interaction with the in-built applications.

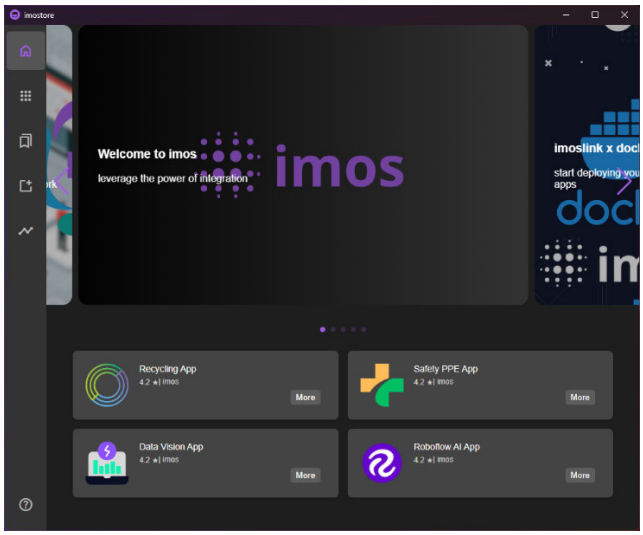


FIGURE 9. IMOStore marketplace home page overview, displaying currently available apps being distributed.

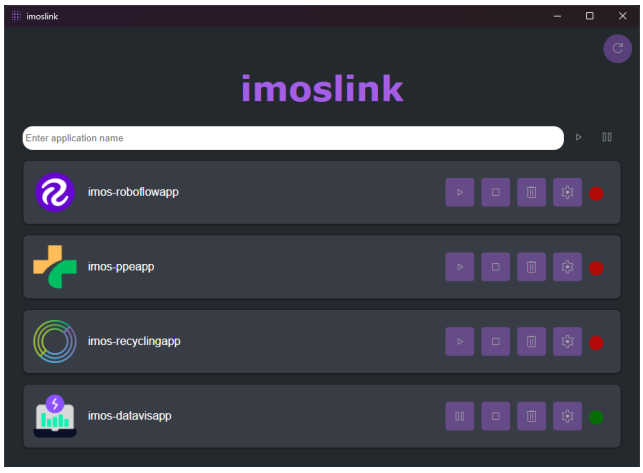


FIGURE 10. IMOSlink environment, with local use-case applications ready for execution and configuration.

3) *Updating docker-compose*: The app's docker-compose file should match the set labels, ensuring proper default values are applied, e.g., MODEL_PORT: $MODEL_PORT:-5001:5001;
4) *Saving app files*: The app's source files are saved to the app repository directory (C: \imos\apps\AppName) as defined in the NSIS build file;
5) *Launching installation*: Every app package requires the launcher to execute the installation after displaying a terms and agreement. Finally, Docker commands are executed, and the app-specific images are set up on the OCP.

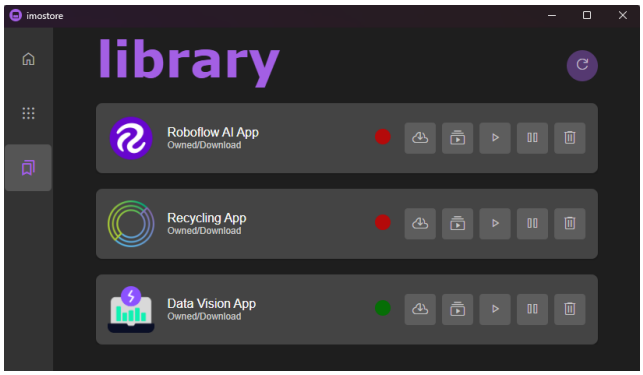


FIGURE 11. IMOScloud interface, hosted on IMOStore library page, offering cloud-app orchestration tools.

If the app packaging, labeling and setup was successful, upon downloading a new app, a NSIS executable becomes available for installation. Running the installer saves the app's source files to the correct directory, executes Docker commands, and integrates the module enabling it for usage.

The IMOS prototype proof of concept was successfully completed, creating a centralized platform with its key modules. All of the depicted implementation efforts are available at the project's repository [36] and demonstrated on the demo videos [37], [38].

VI. APPLICATION CASES AND RESULTS

To showcase IMOS's scalability and versatility, four application cases were developed and integrated, proving that any Docker-containerized app can run on the platform. These include two custom-built apps with AI features - the Recycling App and Safety PPE App - and two externally sourced apps: the Data Vision App, tested in a real industrial scenario, and the Roboflow App, based on the Roboflow API. While optimizing these applications was not the project's main goal, they effectively demonstrate the platform's capabilities in software distribution and integration as proof of concept.

A. RECYCLING APP

The Recycling App was an early use case for IMOS, designed to demonstrate its capabilities. This app uses a pre-trained AI model to classify residues into categories such as paper, plastic, glass, cardboard, metal, organic, and critical residues. It operates through two Docker containers: an inference server (recyclingapp-model-detector) and a graphical user interface client (recyclingapp-gui).

The recyclingapp-model-detector includes a Flask server with multiple endpoints, which handle HTTP requests and utilize the PyTorch Hub Inference API [39] and OpenCV for bounding box annotations. The AI model, based on YOLOv5 from Ultralytics [40], was trained on over 20,000 images collected through web scraping, real-data acquisition, with the labeling carried out on the Roboflow tool [41].

To use the app users must input the webcam's IP address and set the ports for its two containers on the configuration console. Accurate configuration is crucial to avoid conflicts such as port or IP address collisions, which could disrupt functionality. For testing, a script was created to stream a camera image to a random IP, simulating a webcam connection.

The model inference results are presented in Fig. 12. The interface is launched from the recyclingapp-gui container, featuring a live camera feed, real-time detections, and an inference history table and graph. These results are used in the application's interface visualization and can be included into smart recycling processes via a service-oriented endpoint.

FIGURE 12. Recycling App provided interface, showcasing inference results, detections history table and bar chart.

B. SAFETY PERSONAL PROTECTIVE EQUIPMENT APP

This application expands on the previous interface and model inference framework to deliver a Personal Protective Equipment (PPE) detection solution. The AI model, trained on a compact, web-sourced dataset, can recognize six classes of PPE: gloves, individuals, glasses, boots, vests, and helmets. Its purpose is to improve workplace safety by reducing accidents and ensuring adherence to safety protocols.

Just like the previous example, the Safety PPE App features a user interface that displays live image results and inference history and includes a detection endpoint for integration with on-site automation systems.

C. DATA VISION APP

A prominent use case is the Data Vision App, introduced in a previous work by UNINOVA-CTS and Introsys SA researchers [42] and integrated with IMOS following containerization, labeling, and packaging guidelines. This cloud-based machine learning application forecasts energy consumption and has been tested in a real-world industrial environment. It employs a technology stack that includes a Node-RED API for hosting models, Apache Kafka for messaging, PostgreSQL for data management, and Grafana for data visualization.

Fig. 13 compares real observed data to the application's estimates for energy usage and duration across several

recipes. The graph depicts energy consumption for the cell and the robot, emphasizing the accuracy of the projections.

Implemented in robotic cells meeting Volkswagen and Ford standards, the app accurately predicts energy consumption, helping managers identify issues and optimize scheduling. The solution was demonstrated at Introsys SA's facility in Quinta do Anjo, Portugal, with Volkswagen and Ford robotic stations as demonstrated by the cited authors.

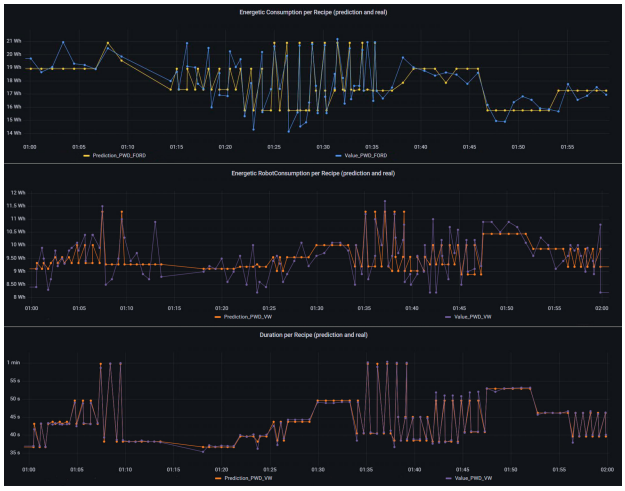


FIGURE 13. Real and predicted values of energy consumption and duration in the case-study cells, from [42].

D. ROBOFLOW APP

The Roboflow App utilizes the Roboflow Inference API to operate open-source AI models. Users can deploy pretrained models by entering the project ID and model version, authenticating with their API key, and selecting the desired model. Designed with the recommended inference SDK engine, the app processes user requests and delivers results in JSON format via a Flask server. It also saves annotated media on the user's local storage, usually located in the application directory (C: \imos\apps\AppName\Volume), and supports running multiple instances with different models.

To use the Roboflow App, users must configure settings such as the API URL and KEY, project ID, model version, and confidence level. After setup, the app provides an endpoint for annotation requests, enabling automation across various processes like quality control, fault detection, safety monitoring, and product sorting.

For demonstration, a 3D-printing fault detection model was used, as depicted in Fig. 14. Requests for annotations of a defective 3D-printed box were made via Postman, and the app provided are saved in the application's volume directory, validating the model's accuracy and the app's functionality.

E. TESTING RESULTS

Finally, the authors present the following Table 2, showcasing some performance results for IMOSlink actions across various application use cases. Each of the five actions was executed 50 times per application, with the average execution

FIGURE 14. Roboflow App results of 3D-printing fault detection model. It correctly locates and identifies the depicted defect.

TABLE 2. Performance testing of applications in IMOSlink.

Action	Application use case	Execution Time (ms)
Configure and create process	Roboflow App	456,76
	Data Vision App	7402,04
	Recycling App	1632,68
	PPE App	1750,19
Start process	Roboflow App	403,11
	Data Vision App	5219,88
	Recycling App	1433,25
	PPE App	1521,90
Stop process	Roboflow App	10554,34
	Data Vision App	29137,05
	Recycling App	21283,34
	PPE App	21599,13
Delete container	Roboflow App	70,53
	Data Vision App	1026,19
	Recycling App	526,67
	PPE App	554,05
Delete image	Roboflow App	76,42
	Data Vision App	1034,53
	Recycling App	560,06
	PPE App	701,60

The execution times in this table reflect the efficiency of deploying and orchestrating software applications on IMOS. When comparing results, it is essential to consider factors such as application size, complexity, and whether they use single or multiple containers.

times derived from metrics logged by a test script developed for this purpose.

Overall, the applications demonstrate satisfactory performance in the actions executed by IMOSlink, with relatively fast execution times. The only exception is the stop action, which typically requires additional time to ensure the safe halting of services and data.

The application use case creation stage provided valuable insights for implementing modular, industry-oriented apps, training AI models, and assessing IMOS's readiness to serve them or any other application.

VII. CONCLUSION, LIMITATIONS, AND FUTURE WORK

As this paper concludes, the work has shown promise in addressing I4.0 challenges - such as software integration, distribution, and modularity - and I5.0 goals - focusing on sustainability, collaboration, and software symbiosis. Results

are demonstrated in the IMOS short [37] and extended [38] demo videos, and on the IMOS GitHub repository [36].

The authors propose that developing a marketplace-driven and integration-ready platform could cultivate an ecosystem where stakeholders collaborate by sharing resources, forming symbiotic relationships, and driving innovation. Continued advancements in IMOS could lay the foundation for this ecosystem, promoting collaboration and innovation within a flexible and scalable framework.

Creating an ideal collaborative environment and managing an industry-oriented marketplace is complex, partly due to the market competitiveness, security risks, and quality assurance. Besides robust security measures, including data encryption (both in transit and at rest), secure authentication protocols, and access control mechanisms, a solution like this would require a robust back-office and solid standards to ensure the validation, suitability and proprietary control over the assets being distributed in the marketplace.

The roadmap for addressing IMOS' limitations includes transitioning to a Linux-based system to leverage its stability, security, and open-source advantages. This shift will improve performance and scalability, reducing reliance on high-code solutions like the NodeJS and Electron-based application.

Additionally, applying Helm Kubernetes [43] as the platform's Open Collaborative Platform (OCP) could enhance IMOS's scalability by managing more complex applications while mitigating challenges arising from Docker's move towards proprietary control.

Expanding research on IMOScloud and server infrastructure could enhance scalability, offering dedicated Virtual Machines or Shared Cloud Spaces for user applications, thereby addressing the presented simplified approach. Furthermore, standardizing asset descriptions using reference architectures like RAMI4.0 and the AAS concept can facilitate asset publication or distribution and ensure interoperability within the IMOS ecosystem.

Future work will focus on addressing challenges and refining the IMOS prototype to ensure readiness for testing in real industrial environments. This includes enhancing system robustness and implementing security measures to meet industry standards. Testing in real use cases will enable the iteration of the platform's requirements across different industry sectors, adapting the solution to ensure it meets their unique operational needs and scaling customization.

Finally, comprehensive documentation will enable users to maximize IMOS's potential, whether by developing compatible applications, contributing to the platform's evolution, or adapting it to their needs. By open sourcing IMOS, *the authors aim to create a collaborative ecosystem, shaped by the industry for the industry*.

REFERENCES

[1] C. Gröger, S. Silcher, E. Westkämper, and B. Mitschang, "Leveraging apps in manufacturing. A framework for app technology in the enterprise," *Proc. CIRP*, vol. 7, pp. 664–669, Jun. 2013, doi: 10.1016/j.procir.2013.06.050.

[2] K. Manikas and K. M. Hansen, "Software ecosystems—A systematic literature review," *J. Syst. Softw.*, vol. 86, no. 5, pp. 1294–1306, May 2013, doi: 10.1016/j.jss.2012.12.026.

[3] V. Kishorre Annanth, M. Abinash, and L. B. Rao, "Intelligent manufacturing in the context of Industry 4.0: A case study of Siemens industry," *J. Phys., Conf. Ser.*, vol. 1969, no. 1, Jul. 2021, Art. no. 012019, doi: 10.1088/1742-6596/1969/1/012019.

[4] *Siemens Insights Hub*. Accessed: May 3, 2024. [Online]. Available: https://plm.sw.siemens.com/en-U.S./insights-hub/

[5] A. R. Akula, P. Calyam, R. B. Antequera, and R. E. Leto, "Advanced manufacturing collaboration in a cloud-based app marketplace," in *Proc. Comput. Frontiers Conf.*, May 2017, pp. 146–155, doi: 10.1145/3075564.3077547.

[6] V. Anaya, F. Fraile, A. Aguayo, O. García, and Á. Ortiz, "Towards IoT analytics. A vf-OS approach," in *Proc. Int. Conf. Intell. Syst. (IS)*, Sep. 2018, pp. 570–575, doi: 10.1109/IS.2018.8710476.

[7] J. Giao, A. A. Nazarenko, F. Luis-Ferreira, D. Gonçalves, and J. Sarraipa, "A framework for service-oriented architecture (SOA)-based IoT application development," *Processes*, vol. 10, no. 9, p. 1782, Sep. 2022, doi: 10.3390/pr10091782.

[8] M. Abubakr, A. T. Abbas, I. Tomaz, M. S. Soliman, M. Luqman, and H. Hegab, "Sustainable and smart manufacturing: An integrated approach," *Sustainability*, vol. 12, no. 6, p. 2280, Mar. 2020.

[9] L. M. Camarinha-Matos, A. D. Rocha, and P. Graça, "Collaborative approaches in sustainable and resilient manufacturing," *J. Intell. Manuf.*, vol. 35, no. 2, pp. 499–519, Feb. 2024.

[10] A. Torayev, G. Martínez-Arellano, J. C. Chaplin, D. Sanderson, and S. Ratchev, "Towards modular and plug-and-produce manufacturing apps," *Proc. CIRP*, vol. 107, pp. 1257–1262, May 2022, doi: 10.1016/j.procir.2022.05.141.

[11] Eur. Commission. (2019). *Zero Defect Manufacturing Platform: ZDMP Project: Fact Sheet: H2020: Cordis: European Commission*. [Online]. Available: https://cordis.europa.eu/project/id/825631

[12] F. Psarommatis, F. Fraile, and F. Ameri, "Zero defect manufacturing ontology: A preliminary version based on standardized terms," *Comput. Ind.*, vol. 145, Feb. 2023, Art. no. 103832, doi: 10.1016/j.compind.2022.103832.

[13] K. Alexopoulos, T. Tsoukaladelis, C. Dimitrakopoulou, N. Nikolakis, and A. Eytan, "An approach towards zero defect manufacturing by combining IIoT data with industrial social networking," *Proc. Comput. Sci.*, vol. 217, pp. 403–412, May 2023, doi: 10.1016/j.procs.2022.12.236.

[14] *Kitt4SME*. Accessed: Jan. 17, 2024. [Online]. Available: https://kitt4sme.eu/

[15] *RAMP Marketplace*. Accessed: Jan. 17, 2024. [Online]. Available: https://ramp.eu/#/home

[16] B. Gladysz, D. Matteri, K. Ejsmont, D. Corti, A. Bettoni, and R. Haber Guerra, "Platform-based support for AI uptake by SMEs: Guidelines to design service bundles," *Central Eur. Manage. J.*, vol. 31, no. 4, pp. 463–478, Nov. 2023, doi: 10.1108/cemj-08-2022-0096.

[17] V. Harish, D. Krishnaveni, and A. Mansurali, "Artificial intelligence in manufacturing," in *Reinventing Manufacturing and Bus. Processes Through Artificial Intelligence*. Springer, 2021, pp. 63–77.

[18] B. Baldassarre, M. Schepers, N. Bocken, E. Cuppen, G. Korevaar, and G. Calabretta, "Industrial symbiosis: Towards a design process for eco-industrial clusters by integrating circular economy and industrial ecology perspectives," *J. Cleaner Prod.*, vol. 216, pp. 446–460, Apr. 2019, doi: 10.1016/j.jclepro.2019.01.091.

[19] T. Lyu, U. Dwi Atmojo, and V. Vyatkin, "Towards cloud-based virtual commissioning of distributed automation applications with IEC 61499 and containerization technology," in *Proc. IECON – 47th Annu. Conf. IEEE Ind. Electron. Soc.*, Oct. 2021, pp. 1–7, doi: 10.1109/IECON48115.2021.9589945.

[20] T. Goldschmidt, S. Hauck-Stattelmann, S. Malakuti, and S. Gruner, "Container-based architecture for flexible industrial control applications," *J. Syst. Archit.*, vol. 84, pp. 28–36, Mar. 2018, doi: 10.1016/j.sysarc.2018.03.002.

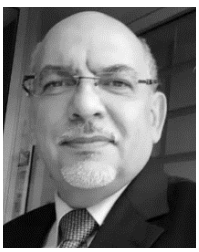
[21] L. M. Camarinha-Matos and H. Afsarmanesh, "The evolution path to collaborative networks 4.0," in *IFIP Advances in Information and Communication Technology*. Cham, Switzerland: Springer, 2021, pp. 170–193. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-030-81701-5_7

[22] (2015). *Transforming Our World: The 2030 Agenda for Sustainable Development*. [Online]. Available: https://sdgs.un.org/2030agenda

[23] L. M. Camarinha-Matos, R. Fornasiero, and H. Afsarmanesh, "Collaborative networks as a core enabler of Industry 4.0," in *Collaboration a Data-Rich World. PRO-VE*. Cham, Switzerland: Springer, Jan. 2017, pp. 3–17. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-319-65151-4_1
[24] I. Khan, M. Jurmu, M. Jurvansuu, S. Pirttikangas, J. Lilius, B. Iancu, O. Kauppila, M. Koho, E. Marjakangas, and J. Majava, "Collaborative innovation, collaborative capabilities and value co-creation in an Industry 4.0 context: An empirical evidence," 2021, *arXiv:2103.16279*.
[25] S. Esposito De Falco, A. Renzi, B. Orlando, and N. Cucari, "Open collaborative innovation and digital platforms," *Prod. Planning Control*, vol. 28, no. 16, pp. 1344–1353, Dec. 2017, doi: 10.1080/09537287.2017.1375143.
[26] N. Smorodinskaya, M. Russell, D. Katukov, and K. Still, "Innovation ecosystems vs. Innovation systems in terms of collaboration and co-creation of value," in *Proc. Annu. Hawaii Int. Conf. Syst. Sci.*, 2017, pp. 1–20, doi: 10.24251/hicss.2017.636.
[27] A. Neves, R. Godina, S. G. Azevedo, and J. C. O. Matias, "A comprehensive review of industrial symbiosis," *J. Cleaner Prod.*, vol. 247, Feb. 2020, Art. no. 119113, doi: 10.1016/j.jclepro.2019.119113.
[28] L. Camarinha-Matos and H. Afsarmanesh, "Classes of collaborative networks," in *Encyclopedia of Networked and Virtual Organizations*. Hershey, PA, USA: IGI Global, 2008, pp. 193–198, doi: 10.4018/9781605667706.ch021.
[29] A. M. Pegado, A. D. Rocha, and J. Barata, *Design and Development of a Marketplace-Based Collaborative Ecosystem for Software Integration and Distribution Within Manufacturing*. Cham, Switzerland: Springer, 2024, pp. 80–95. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-031-71743-7_6
[30] I. K. Chaniotis, K.-I.-D. Kyriakou, and N. D. Tselikas, "Is Node.Js a viable option for building modern Web applications? A performance evaluation study," *Computing*, vol. 97, no. 10, pp. 1023–1044, Oct. 2015, doi: 10.1007/s00607-014-0394-9.
[31] K. Kredpattanakul and Y. Limpiyakorn, *Transforming JavaScript-Based Web Application To Cross-Platform Desktop With Electron*. Singapore: Springer, 2019, pp. 571–579. [Online]. Available: https://link.springer.com/chapter/10.1007/978-981-13-1056-0_56
[32] D. Merkel, "Docker: Lightweight Linux containers for consistent development and deployment," *Linux J.*, vol. 2014, no. 239, p. 2, Mar. 2014.
[33] M. Wicha and B. Panczyk, "Performance analysis of REST API technologies using spring and Express.Js examples," *J. Comput. Sci. Inst.*, vol. 29, pp. 352–359, Dec. 2023, doi: 10.35784/jcsi.3796.
[34] S. H. Aboutorabiu, M. Rezapour, M. Moradi, and N. Ghadiri, "Performance evaluation of SQL and MongoDB databases for big e-commerce data," in *Proc. Int. Symp. Comput. Sci. Softw. Eng. (CSSE)*, Aug. 2015, pp. 1–7, doi: 10.1109/CSICSSE.2015.7369245.
[35] J. B. Tyndall, "Building an effective software deployment process," in *Proc. 40th Annual ACM SIGUCCS Conference User Services*, Sep. 2012, pp. 109–114, doi: 10.1145/2382456.2382482.
[36] *IMOS Project GitHub Repository*. Accessed: Jun. 11, 2024. [Online]. Available: https://github.com/altsmpegado/imos
[37] A. M. Pegado. (2024). *IMOS Short Demo Video*. Accessed: Jun. 11, 2024. [Online]. Available: https://youtu.be/gMBulW-aJ8Y
[38] A. M. Pegado. (2024). *IMOS Platform Overview and Project Insights*. [Online]. Available: https://youtu.be/HzBn3Vpyl_c
[39] *PyTorch Inference API*. Accessed: Apr. 13, 2024. [Online]. Available: https://pytorch.org/serve/inference_api.html
[40] *Ultralytics YOLOv5 GitHub*. Accessed: Apr. 2, 2024. [Online]. Available: https://github.com/ultralytics/yolov5
[41] *Roboflow*. Accessed: Mar. 13, 2024. [Online]. Available: https://roboflow.com/
[42] N. Freitas, S. O. Araujo, D. Alemão, J. Ramos, M. Guedes, J. Gonçalves, R. S. Peres, A. D. Rocha, and J. Barata, "Cloud-based machine learning application for predicting energy consumption in automotive spot welding," *Processes*, vol. 11, no. 1, p. 284, Jan. 2023, doi: 10.3390/pr11010284.
[43] *Helm Kubernetes*. Accessed: Jun. 27, 2024. [Online]. Available: https://helm.sh/docs/

ANTÓNIO MONTE PEGADO received the M.Sc. degree in electrical and computer engineering from the Department of Electrical Engineering, School of Science and Technology, NOVA University of Lisbon, in 2024, with a focus on robotics and digital systems, where he is currently pursuing the Ph.D. degree. His thesis research specializes in software engineering and integration applied to manufacturing. He was a Collaborative Researcher with the Centre of Technology and Systems (CTS), UNINOVA Institute. In addition to his academic endeavors, he has been involved in some entrepreneurial projects applying artificial intelligence and computer vision to enhance recycling processes. He is also the Co-Founder and the CTO of Trash4Goods, an innovative startup developing a gamified rewards platform to encourage higher recycling rates and promote environmental sustainability.

ANDRE DIONISIO ROCHA (Member, IEEE) received the M.Sc. and Ph.D. degrees in electrical and computer engineering from the NOVA University of Lisbon, Portugal, in 2013 and 2018, respectively. He is currently a Professor with the Department of Electrical Engineering, NOVA University of Lisbon, and a Senior Researcher with the Centre of Technology and Systems (CTS), UNINOVA Institute. He has participated in several research projects, most of them at European level (such as H2020 Avangard, H2020 GO0DMAN, H2020 PERFoRM, and H2020 openMOS), involving different programs (FP7, QREN, H2020, and P2020). His research interests include the design and development of cyber-physical production systems and the study of smart manufacturing solutions using distributed control systems, modular production systems, and artificial intelligence-based solutions to deliver more cost-effective and sustainable solutions. As a result of his research activities, he has published several scientific contributions in scientific journals and international conferences. He is highly involved and active in IEEE activities. He is a member of the IEEE-IES Technical Committee on Industrial Cyber-Physical Systems (TC-ICPS) and the IEEE-IES Technical Committee on Industrial Agents (TC-IA).



JOSE BARATA (Member, IEEE) received the Ph.D. degree in robotics and integrated manufacturing from the NOVA University of Lisbon, in 2004. He is currently a Professor with the Department of Electrical Engineering, NOVA University of Lisbon, and a Senior Researcher with the UNINOVA—Instituto de Desenvolvimento de Novas Tecnologias. He has participated in more than 15 international research projects involving different programs, including NMP, IST, ITEA, and ESPRIT. Since 2004, he has been leading the UNINOVA participation in EU projects, namely, EUPASS, self-learning, IDEAS, PRIME, RIVER WATCH, ROBO-PARTNER, and PROSECO. In the last years, he has participated actively in researching SOA-based approaches for the implementation of intelligent manufacturing devices, such as within the Inlife Project. He has authored or co-authored over 100 original papers in international journals and international conferences. His research interests include intelligent manufacturing, with an emphasis on complex adaptive systems involving intelligent manufacturing devices. He is a member of the IEEE Technical Committee on Industrial Agents (IES), the Self-Organization and Cybernetics for Informatics (SMC), and the Education in Engineering and Industrial Technologies (IES). He is also a member of the IFAC Technical Committee 4.4 (Cost-Oriented Automation).