applied sciences

MDPI

Article

Low-Code as Enabler of Digital Transformation in Manufacturing Industry

Raquel Sanchis [1,*], Óscar García-Perales [2], Francisco Fraile [3] and Raul Poler [1]

[1] Escuela Politécnica Superior de Alcoy, Research Centre on Production Management and Engineering, Universitat Politècnica de València, Calle Alarcón, 03801 Alcoy (Alicante), Spain; rpoler@cigip.upv.es
[2] Information Catalyst SL. Ptda Tosal de la Cometa, 1-F-1-D, BL 3501, 03710 Calpe (Alicante), Spain; oscar.garcia@informationcatalyst.com
[3] Research Centre on Production Management and Engineering, Universitat Politècnica de València, Camino de Vera s/n, 46022 Valencia, Spain; ffraile@cigip.upv.es
* Correspondence: rsanchis@cigip.upv.es

Received: 23 November 2019; Accepted: 15 December 2019; Published: 18 December 2019

Abstract: Currently, enterprises have to make quick and resilient responses to changing market requirements. In light of this, low-code development platforms provide the technology mechanisms to facilitate and automate the development of software applications to support current enterprise needs and promote digital transformation. Based on a theory-building research methodology through the literature and other information sources review, the main contribution of this paper is the current characterisation of the emerging low-code domain following the foundations of the computer-aided software engineering field. A context analysis, focused on the current status of research related to the low-code development platforms, is performed. Moreover, benchmarking among the existing low-code development platforms addressed to manufacturing industry is analysed to identify the current lacking features. As an illustrative example of the emerging low-code paradigm and respond to the identified uncovered features, the virtual factory open operating system (vf-OS) platform is described as an open multi-sided low-code framework able to manage the overall network of a collaborative manufacturing and logistics environment that enables humans, applications, and Internet of Things (IoT) devices to seamlessly communicate and interoperate in the interconnected environment, promoting resilient digital transformation.

Keywords: low-code; trend; vf-OS; digital; platform; apps

1. Introduction

Currently, enterprises have to face increasingly difficult problems due to the increasing complexity of their internal operations and the number and intensity of the relationships between the company and the entities of its supply network. Moreover, the changing market means enterprises need a rapid and flexible response to fulfil the variable requirements of the environment. For this reason, companies require the capacity to withstand the stresses of environmental loading. This capacity has been defined as enterprise resilience. Sanchis and Poler [1] defines it as the capacity to prevent and anticipate; to change enterprises' nature and adapt to the changing environment; and to respond to the dynamic requirements.

Additionally, it is also unquestionable that enterprises have to give a response to these varying market requirements quickly. Therefore, rapidity is a very important factor in the current context.

In order to enhance the resilience capacity of enterprises to make rapid and efficient responses to the market needs, efforts have been focused on development of software solutions for companies. A vast amount of research efforts in the history of computer science have been focused on the same objective: enabling the construction of software applications without recurring to traditional hand-coding and programming [2]. This is the reason why the use of agile methodologies is rising in the current context to support the digital transformation of companies. The results of the survey performed by [3] concludes that digital transformation is a work in progress, as the survey evaluates the enterprises' progress with digital transformation on a six-point scale. Respondents awarded their organisations a score of 3.74, meaning that digital transformation efforts are typically widespread, but not yet strategic or continuous. In light of this, low-code development platforms are considered as a trendy mechanism to facilitate the rapid development of software applications (apps) and also its automation to support the current enterprise needs and facilitate resilient digital transformation.

The term "low-code" was firstly coined by Forrester Research in 2014 (Cambridge, MA, USA) [4], which states that firms prefer to choose low-code alternatives for fast, continuous, and test-and-learn delivery. Low-code development platforms are ecosystems with which apps can be developed, minimizing hand-code definition manually, because it is already built and prefigured. Low-code development platforms emphasize visual interfaces to enable people, without a technological background, to create and deploy business apps with relative ease [5]. The main objective of the low-code development platforms is to allow enterprises to develop apps without complex engineering facilitating their configuration, and then achieving, rapidity and agility. Moreover, these platforms also offer enterprises a more economical way to fulfil the market and/or enterprises internal requirements. With the low-code development platforms, enterprises can create programs or apps for mobile or desktop devices, multifunctional and with high information-management capabilities.

Many experts [3,4,6] point out the successful future of these platforms, highlighting their main benefits:

- Privacy. As the apps can be developed by users without a deep expertise in technical issues, enterprises trust their staff and these development tasks are not usually outsourced to third parties but performed internally which increases the confidentiality [3].
- Rapidity. As the main part of the code is already developed, users only have to visually configure the apps instead of hand-coding them or make the necessary adjustments to develop the apps they need [3]. As the development time is reduced, the availability of the apps is very fast. A survey performed by Forrester [6] showed that low-code development platforms accelerated development by 5 to 10 times.
- Cost reduction. Due to the reduction in the development cycle from a time viewpoint, the cost is also reduced whether the app is developed by the company or by external developers [4].
- Complexity reduction. As the apps are not built from scratch, the apps development is simplified and this fact enables to focus more on customizing the software to fulfil users' requirements.
- Easy maintenance. The maintenance phase of software is vital to be able to quickly change what already has been developed to guarantee a permanent alignment between the service offered by the app and the business requirements. In light of this, as the essentials of the low-code development platforms are to offer little code, there is little code to maintain [3].
- Involvement of business profiles. These platforms provide simple and intuitive interfaces as a development environment for the deployment of apps. In this context, no technological knowledge is required, and the final users of these apps become the developers of such apps as they are the ones who have a deep knowledge about the business needs [5]. According to [3] 44% of the low-code development platform users are business users in collaboration with IT.
- Minimisation of unstable or inconsistent requirements. In the current software development process, potential conflicts might arise among requirements and the impacts on the app design of requirements' changes. However, following the idea of the previous benefit, the use of low-code means that developers quickly build minimum viable products to validate ideas and customer requirements before wasting resources on features and functionalities that customers may not value [6].

Based on the 'The State of Application Development' report [3] that shows the results obtained through a survey answered by more than 3300 IT professionals across different continents, the main reasons for using low-code development platforms are detailed in Figure 1. Of the respondents, 66% chose accelerate digital transformation and increase responsiveness to the business as the main motives why they use or will use low-code development platforms; 45% of the surveyed professionals pointed to the reduction of dependency on hard-to-hire technical skills.

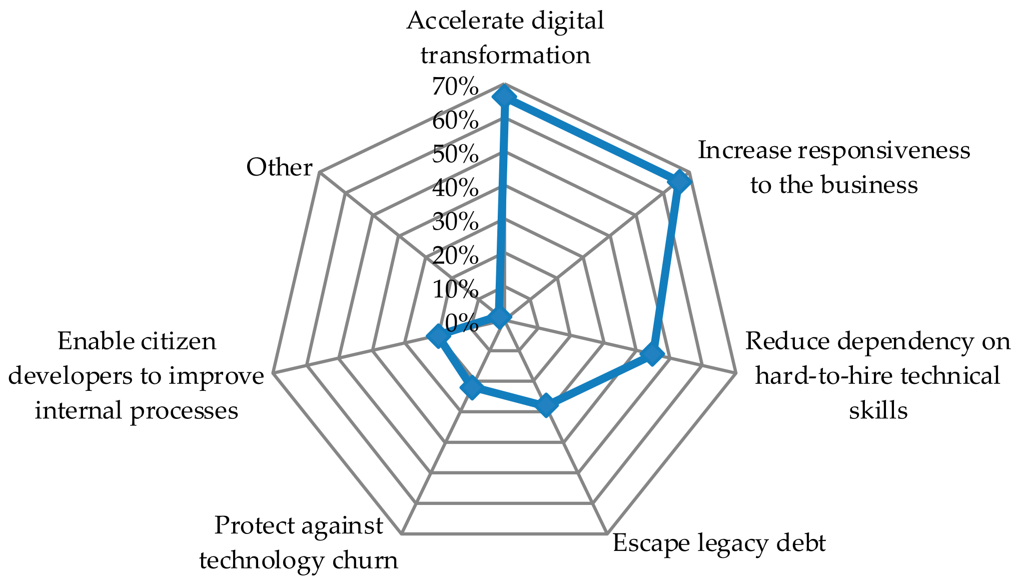

Figure 1. Main reasons for using low-code development platforms.

Richardson and Rymer [6] state that low-code development platforms bring several benefits, but some risks have also to be taken into consideration. Based on the previous advantages, low-code development platforms provide useful solutions in automating and speeding application delivery, and high vendor growth rates. However, Richardson and Rymer [6] also highlight the risk of dozens of small vendors selling outside of tech management, and customers with little consensus about how low-code development platforms fit into their broader portfolios. Tisi et al. [2] also highlight three main limitations that hamper the use of the low-code development platforms:

- Scalability: the authors point out that low-code development platforms are mainly addressed for the development of small apps but their application in large-scale projects and mission-critical enterprise applications is not covered currently.
- Fragmentation: different low-code development paradigms can be defined depending on each vendor and their specific programming model.
- Software-only systems: while enterprise developers have little expertise of programming, they are often experts in some other engineering areas. These experts expect to be able to use their knowledge in the application, at the right level of abstraction and using familiar formalisms.

Along the same lines, the survey conducted by [3] shows that the main reasons why organisations are not using a low-code platform, or are not thinking to use one are the lack of knowledge closely followed by concerns about lock-in, flexibility, scalability, and security (Figure 2).

Based on this, the objective of this paper is to depict the status of the existing automation software development tools focusing on the characterisation of low-code development platforms following the foundations of the computer-aided software engineering field. This research also pursues the identification of the potential low-code challenges for further research.

In light of this, the paper is addressed to the depiction of existing knowledge on low-code to describe the current context of the research focused on this topic and identify uncovered challenges and promising directions to provide foresight into the future of low-code research. As such, this study builds upon two research questions:

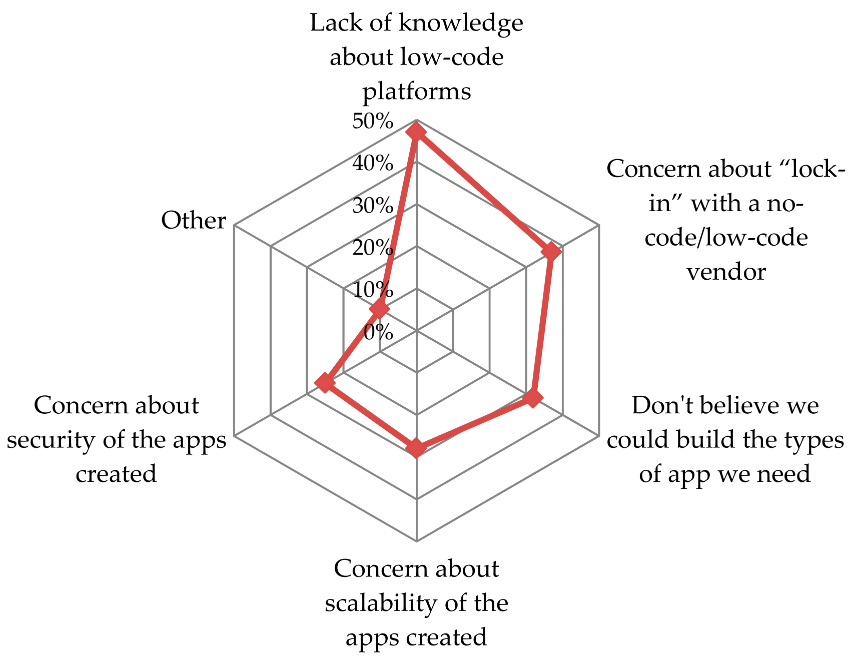

Figure 2. Main reasons for not using or considering low-code development platforms.

RQ1. What is the status about existing automation software development tools?

RQ2. What are the potential challenges in the context of automation software development tools for further research?

The paper is structured as follows. Section 2 offers an overview of the context analysis, describing the concept of computer-aided software engineering as a predecessor attempt to facilitate the automation of software development and then focusing on the current status of research related to the low-code development platforms. Section 3 performs a comparative analysis among the existing low-code development platforms targeting to manufacturing industry and describes the vf-OS (virtual factory open operating system) platform from a broad-spectrum and low-code viewpoints. Finally, in Section 4 the main conclusions, limitations and further research lines and trends are detailed.

2. Context Analysis

Software and apps development is increasingly necessary to deal with the great amount of information and data (big data) that companies have to manage, which is also becoming a more complex process. In order to facilitate the software and apps development, companies specialised in software development are demanding a higher level of automation in their software development work. Moreover, currently different technologies have become more accessible for non-professional users [7]. In consequence, there is an increasingly need for more efficient methods and tools for automatically developing and maintaining computer systems [8]. To deal with these needs, in the late 1980s the concept of computer-aided software engineering (CASE) emerged.

CASE technology encompasses a collection of automated tools and methods that assist software engineering in the phases of the software development life cycle [9]. Lundell and Lings [10] define CASE technology as an interoperable, computerised tool set designed to support stakeholder tasks and processes over the full information systems development lifecycle. CASE-tools are often based on the object-oriented approach to visual modelling of the unified modeling language (UML), content management systems and business process management systems [7].

Fuggetta [11] classifies CASE products in the production process technology into three categories:

- Tools considered as software components supporting a specific task in the software-production process. Such tool are in turn also classified as: editing tools (textual and graphical editors), programming tools (coding and debugging tools, code generators and code restructurers); verification and validation tools (Static and dynamic analysers, comparators, symbolic executors, emulators/simulators, correctness proof assistants, test-case generators and test-management tools); configuration-management tools (version management, item identification, configuration building, change control, library management); metrics and measurement tools (code analysers, execution monitor's timing analysers); project management (cost-estimation tools, project-planning tools, conference desks, e-mail, bulletin boards, project agendas, project note books), miscellaneous tools (hypertext systems and spreadsheets).

- Workbenches that integrate in a single application, several tools supporting specific software-process activities. Among them, the author highlights business planning and modelling; analysis and design; user-interface development; programming; verification and validation; maintenance and reverse engineering; configuration management and project management.
- Environments defined as a collection of tools and workbenches that support the software process. The environments involve: toolkits; language-centred; integrated; fourth generation and process-centred.

Little recent evidence of the use of CASE technology to develop tools is found in the literature (through the search of a set of keywords related to CASE in the Web of Science database). Some of the latest works (from 2017 to 2019) related to the development of CASE-tools are: Vileiniskis et al., [12] perform the conceptual design and implementation of a practical lightweight approach to model traceability in a CASE tool. Madoš et al. [13] who develop a CASE-tool, that supports programming the experimental multi-core tile-based system-on-a-chip. The proposed tool allows the creation of the program code in graphical form using data flow graph or in text representation using assembly language or machine code. Automatic transformations between those representations are supported. Al-Ashwal et al. [8] who develop a prototype of a CASE-tool for Java logical errors detecting using static and dynamic testing techniques. Their research utilizes the Junit and PMD (programming mistake detector) tools to detect the logical errors and analyse the potential causes of these errors based on Java common logical errors lists. Tarasiev et al. [7] who develop a prototype of CASE-tool to create automation systems based on web applications using code generation. Hadj Sassi et al. [14] propose a CASE-tool that supports the design of business intelligence for Internet of Things (IoT) architecture based on knowledge.

However, the adoption of CASE technology has been poor [10]. Troy and McQueen [15] developed a methodology for designing domain-specific CASE tools supporting model-based analysis and automatic code generation. Nevertheless, the authors, after producing a prototype methodology companion designed to assist the task of developing control software for programmable logic controllers in batch process manufacturing, point out that the mode of code generation limits the feasibility of functional description maintenance and automatic code generation. It is determined by [10] that one of the main reasons for the poor adoption of CASE technologies may be that expectations of CASE are unrealistic. Another reason highlighted by [10] may be that real user requirements are not being adequately met by CASE products. Huff [16] mentions that CASE tools are quite expensive and related training costs may exceed the original price of the tool. Orlikowski [17] underlines that the benefits of CASE do not accrue homogeneously to all interested groups affected, leading to opposition by some groups. Iivari [18] in his article: 'Why are CASE tools not used?' declares that the actual use of CASE technology was much less than one would expect. Some of his conclusions point at the fact that the productivity and quality of software development through CASE tools is overrated. Moreover, complexity is also found as a significant, negatively related, predictor of CASE effectiveness. Currently, all these findings are corroborated by some experts who state that CASE systems still have a number of shortcomings and cannot always be applied.

We do not know whether these limitations have motivated the emergence of low-code technology in the current context, however, what is unquestionable is that the low-code paradigm has burst strongly into the automation of software development. Waszkowski [5] highlights that there is an important potential for the development of low-code solutions. He justified this by the lack of software and apps developers and the growing requirements as to the scope and rate of changes introduced in IT systems. Hand-coding is time-consuming and labour-intensive. Therefore, in order to overcome all these barriers, this paper analyses the current research context related to the low-code paradigm as with this technology, programming is not necessary to build applications [5].

To analyse the context of the low-code development platforms, and based on a theory-building research methodology a systematic literature review has been performed through two databases (Scopus and Web of Science), covering all the range of years and domain categories. As the results were very scarce, then the search on Google Academic was added for the context analysis. The first approximation is focused on the definition of sets of general keywords to quantify the number of publications related to low-code solutions. Table 1 shows a summary of the results obtained.

Table 1. Number of publications according to the diverse searches performed in the literature review.

	"Low-Code"	**And "Application"**	**And "Platform"**	**And "App"**
Scopus	184	37	11	1
Web of Science	208	10	4	1
Google Academic	5290	3450	1410	548

The first set of keywords: "low-code" is then extended with "applications", "platform" and "app". These sets of keywords provide a smaller number of results. In Scopus, only 11 publications are identified to be related to "low-code" and "platform". In the Web of Science, this number decreases to only 4 publications. One important aspect to be taken into account is the publication year. Based on the 11 Scopus publications, 4 were published in 2018 and 4 more in 2019. The other 3 are from 2008, 2009 and 2012, however they are not related to the low-code scope of the present research. In light of this, it is evinced that there are very few publications related to low-code aspects and they are from the last two years, demonstrating its emerging trend.

2.1. Scientific Literature Review

One of the results found in the literature is a summary-report of the Forum session at the 2018 Enterprise Engineering Working Conference (EEWC 2018) [19]. In this report, it is highlighted the use of low-code development platforms to further increase the speed and robustness of the digital transformation by a gradual expansion of the meta-model of the enterprise design approach, continuously applied on real-life cases.

Zolotas et al. [20] presents RESTsec, a low-code platform based on representational state transfer (REST) that supports rapid security requirements modelling for enterprise services, abiding by the state of the art attribute-based access control (ABAC) authorisation scheme. According to the authors, RESTsec enables developers to embed the desired access control policy and generate the service, the security infrastructure, and the code in a rapid and automated way. Another of the results related to low-code aspects is focused on the domain-specific languages (DSLs) used in the development environment of OutSystems Platform (Boston, MA, USA) [21]. In this case, authors study the problematic of the DSLs for process modelling (business process technology, BPT), as it has a low adoption rate and is perceived as having usability problems hampering its adoption. For this reason, they develop and test a new version of BPT to improve developers' experience.

Another result is focused on the development of an integrated software platform (INTELLIT), (Institute of History and Literary Theory "G. Călinescu" of the Romanian Academy, "Lucian Blaga" University of Sibiu, National Institute for Research and Development in Informatics and Polytechnic University of Bucharest, Romania, 2018) as a support for the Virtual Library of Romanian Literature to contribute to preserving and capitalizing on the Romanian literary heritage. This publication highlights the importance of the low-code paradigm for choosing the technologies and models used in the development of the INTELLIT Platform [22]. However, the focus of its research is far from the main objective of the present study.

Tisi et al. [2] presents the project: Lowcomote, that is an innovative training network (ITN) aiming to train professionals in the design, development and operation of low-code development platforms. The motivation of this project is to overcome the limitations that low-code platform presents by being scalable (able to develop largescale applications); open (based on interoperable programming models and standards); and heterogeneous (supporting with models coming from diverse engineering areas).

Pantelimon et al. [23] describe NETIoT, an IoT platform built following the hpaPaaS principles, centred around the applications being supported by a set of IoT devices owned by a user. This study is focused on how to efficiently bring measurements from heterogeneous IoT hardware devices to the NETIoT platform in a low-code manner, with zero intervention on hardware and minimum configuration effort platform-wise. Along the same lines, Waszkowski [5] describes the use of the Aure BPM low-code platform for automating business processes in manufacturing.

Wu et al.'s [24] study is focused on the technical question and answer (Q&A) platforms, such as Stack Overflow. This platform is an instrument for users to ask and answer questions related to a wide variety of programming topics. Moreover, it offers hundreds of thousands of lines of source code that can be used by developers. One of the objectives of this study is to identify barriers that hinder code reuse. After an exploratory study and a survey, the authors found 3 main barriers to reuse code from Stack Overflow: (i) too much code modification required to fit in their projects, (ii) incomprehensive code, and (iii) low-code quality. This third barrier requires the need for next-generation Q&A platforms to improve or verify the code quality of source code snippets and highlights the current importance of the low-code.

2.2. Other Information Sources

As the number of results obtained in the search performed in scientific databases is scarce, due to the novelty of this domain, alternative information sources are used to analyse the current status of the low-code paradigm. For this purpose, Google Academic is the information source selected. As it is shown in Table 1, the number of results found is higher than in the other two scientific databases.

When the search is more narrowed down, adding "application, "platform" and "app" to the set of originals keywords ("low-code"), the number of results decreases considerably. Moreover, it is worth mentioning that more than 40% of the results identified in the search do not correspond to the purpose of the present research, as they are focused on the low code-rates when they are studying the error corrections codes. This concept is used for controlling errors in data over unreliable or noisy communication channels. To avoid problems, redundant bits are added to support the decodification and find out the true message encoded in the communication. The code-rate is defined as the ratio between the number of information bits and the total number of bits in a specific communication channel. A low code-rate close to zero means a strong code that uses many redundant bits to achieve a good performance, while a large code-rate close to 1 implies a weak code [9,25,26].

Taking this into account, it seems that low-code development platforms that provide the necessary functionalities to users to configure operational apps are a novel topic that currently is emerging. Narrowing down the results of the search "low-code" from 2014, that is the year when "low-code" was firstly coined to 2019; and deleting those related to "low-code rates", the total results are only 33% of the original results, what evinced that not too many references study this concept.

With regard to the type of documents, google academic provides as results, scientific publications but also other documents such as reports developed by prestigious global enterprise software companies and market research companies such as OutSystems [27] and Forrester [28]. These companies perform technological vigilance and report the results of such observance. Moreover, they perform surveys addressed to application developers, managers, or information technology (IT) leaders to analyse their challenges with digital transformation, application development, delivery speed, among others.

The market of the low-code development platforms (based on 42 suppliers of these services) generated $1.7 billion in revenue during 2015 [6]. Moreover, in a survey addressed to developers, 23% reported that they used low-code development platforms in 2018, and another 22% planned to do so within a year [29]. Fryling [30] points out that the average cost of a software development project ranges from $434,000 for projects implemented in small and medium-sized enterprise (SMEs) to $2,322,000 for larger projects [31]. Besides these figures, authors state that 31.1% of projects are not finished and they are finally cancelled, 52.7% of the projects exceed the budget in an 89% more, and only 16.2% are completed according to the planned schedule and budget [31].

Based on these results it seems that the low-code topic, more specifically its application through platforms, is a reality and will be a trend in coming years. However, when this topic is searched in scientific sources, not too many evidences are found, what points out the fact that this topic seems to be under-researched and only few low-code development platforms have been developed and launched. This is also more critical when the target markets are the industrial sector where the apps should also take into account Industry 4.0 aspects such as for example data coming IoT devices.

3. Low-Code Development Platforms

Whilst a visual programming language (VPL) is any programming language allowing software developers to craft programs and applications by graphically manipulating elements rather than by hand-coding [32,33], a low-code development platform is a software environment that allows these software developers to create application software through user interfaces (UIs) and configuration instead of traditional computer programming. In other words, VPL allows programming with visual expressions, or spatial arrangements of text and graphic symbols, while low-code development platforms may focus on design and development of a particular kind of application: such as databases, business processes, or user interfaces such as web applications.

For example, many VPLs (known as dataflow or diagrammatic programming) [34] are based on the idea of "boxes and arrows", where boxes or other screen objects are treated as entities, connected by arrows, lines or arcs which represent relations, while low-code development platforms may produce entirely operational applications, or require additional coding for specific situations.

These platforms are usually accessible through free or low-cost self-service offerings. They are based on free and freemium models. Most low-code development platforms are available as public cloud services [6]. The low-cost platforms normally do not require training for developers to get started. The exploitation business model for such platforms is usually based on fees for users and deployed apps. As mentioned, one of the most advantageous aspects of the low-code development platforms is the speed with which developers and their business partners test a business idea through an app, gain feedback, and iterate toward a finished product [6]. Low-code development platforms can also lower the initial cost of setup, training, and deployment [4].

3.1. General Overview of Virtual Factory Open Operating System (vf-OS) Platform

vf-OS is a project funded by the H2020 Framework Programme of the European Commission under Grant Agreement 723710 and conducted in the period October 2016 until August 2019. Further information can be found at [35].

As mentioned, traditional factories will increasingly be transformed into digital manufacturing environments but currently the full potential for ICT in manufacturing is far from being fully exploited [36]. This is the motivation for the vf-OS project.

In order to facilitate the app-building process and not to be dependent on off-the-shelf and third-party software, low-code development platforms are emerging quickly to design, build, customize, and deploy business apps minimizing hand coding. Moreover, as indicated by [6] the next wave of innovation is apps that include IoT devices. The next source of organic growth for low-code platforms will be applications that incorporate sensors and actuators. In this sense, vf-OS is an open multi-sided framework able to manage the overall network of a collaborative manufacturing and logistics environment that enables humans, applications, and devices (IoT) to seamlessly communicate and interoperate in the interconnected environment, giving response to the requirement identified as a trend in 2016 by [6].

vf-OS can be deployed in-cloud and on premises. vf-OS provides different services to connected factories of the future to integrate better manufacturing and logistics processes, allowing collaborative manufacturing based on cross-organisational manufacturing support, and allowing mobile manufacturing [37]. Table 2 shows the functionalities that vf-OS offers [38].

Table 2. Description of virtual factory open operating system (vf-OS) functionalities.

Virtual Factory Input/Output Interface (vf-IO)
It is a set of modules that virtualise factory's real assets and connect them to their images in vf-OS. vf-IO implements plug-and-play mechanisms and device drivers for seamless/open access and smart virtualisation of the factory resources; it is composed by devices drivers, application programming interface (API) connectors, security, and data access. Thus, the vf-IO is composed of the modules that enable connectivity to assets like legacy enterprise resource planning (ERPs) or customer relationship management (CRMs), cyber-physical systems (CPSs), smart objects or wireless sensor networks.
Virtual Factory System Kernel (vf-SK)
It is the core of the operating system, responsible for providing key system resources and a set of specific services, which is open and accessible to other components of the system.
Virtual Factory Devices Drivers and Open Application Programming Interfaces (APIs).
Open APIs, interconnection modules and drivers serve as interoperability mechanisms between the factory and vf-OS applications. The integration between both is seamless and secure. It provides interfaces to physical assets (e.g., sensors) and virtual assets (e.g., ERP systems and data) and eases their use in vf-OS.
Virtual Factory Middleware (vf-MW)
It consists of system services and a data bus, which provide a set of modules to integrate data from arbitrary sources, including, but not limited to CPS, smart objects, radio frequency identification (RFID) devices, and wireless sensor networks. Moreover, the use of cloud-based data storage avoids vendor lock-in issues and minimises the risk of system failures. Accessibility of data is facilitated through connectors and wrappers.
Virtual Factory Open Applications Development Kit (vf-OAK)
A complete and fully open development kit addressed to the software producing community. The aim is addressed to guarantee the growth of specific applications running in vf-OS across all industrial sectors and scenarios. It is composed of a software development kit (SDK) to develop applications, a system dashboard, the OAK Frontend Environment, the OAK Development Studio and a developer engagement hub to engage developers. The SDK implements all the necessary APIs needed to develop vApps. The OAK System Dashboard represents the core software services for allowing system monitoring and configuration; the OAK Frontend Environment provides a framework that facilitates a general 'look, feel, and composition' to vApps and assists rapid development, by providing a compilation of UI elements including business logic via the OAK Studio; the vf-OS Development Studio is a desktop development environment that facilitates software developers to compose their applications for running within vf-OS. Additionally, the Developer Engagement Hub is a collaboration platform for developers to support each other.
vf-OS Applications (vApps)
Manufacturing smart applications enables and optimise communication and collaboration among supply networks across all manufacturing sectors and in all the stages of manufacturing and logistic processes: demand forecast, planning, supply, manufacturing, distribution, storage, replacement, and recycling.
vf-OS Store (vf-Store)
Virtual Factory Manufacturing Application Store offers fundamental services of a modern e-Commerce platform for consumer and developers. On one hand, vf-Store enables software developers to offer assets (demanded or initiative), and on the other hand, users can search for, obtain and rate existing vApps. Furthermore, the vf-Store acts as a mediator between developers and users. Therefore, the vf-Store is the central point for developers to get in contact with users. In addition to view/set ratings, review, and provide technical information about the asset's behaviour, the vf-Store supports users to get in contact with developers to offer

ideas for new assets.
Virtual Factory Platform (vf-P)
This is a holistic service platform, which is the foundation for all services and end user applications that vf-OS provides. vf-P encapsulates and acts as the interface and runtime environment between the components, connectors, OAK functions, marketplace, the service framework (supporting the running of intrinsic services and vApps) and the end user applications/developers. The vf-P can run locally and in cloud environments.

The Virtual Factory Platform is a multi-sided market platform that can create benefits for each of the customer segments presented in Figure 3 and described in Table 3.

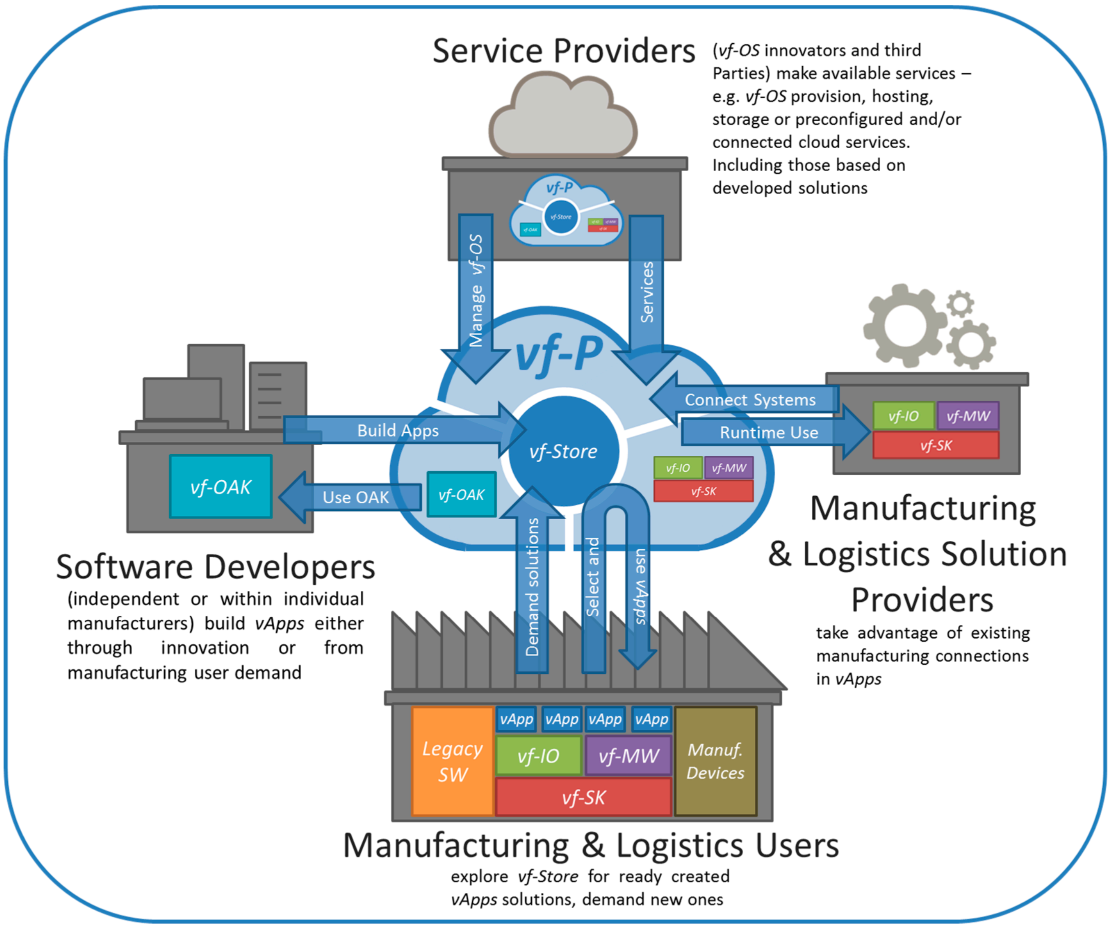
Service Providers
(vf-OS innovators and third Parties) make available services – e.g. vf-OS provision, hosting, storage or preconfigured and/or connected cloud services. Including those based on developed solutions
vf-P
vf-Store
Manage vf-OS
Services
Connect Systems
Runtime Use
Build Apps
Use OAK
vf-OAK
vf-IO
vf-MW
vf-SK
Demand solutions
Select and use vApps
Software Developers
(independent or within individual manufacturers) build vApps either through innovation or from manufacturing user demand
Manufacturing & Logistics Solution Providers
take advantage of existing manufacturing connections in vApps
vApp
Legacy SW
Manuf. Devices
Manufacturing & Logistics Users
explore vf-Store for ready created vApps solutions, demand new ones

Figure 3. vf-OS big picture.

Table 3. Customer segments that can profit from vf-OS.

Software Developers
These users can access a new promising and high-growth potential market for the development of vApps. These are developed using the vf-OAK to quickly build applications running over the vf-SK and using the vf-IO and the vf-MW. The software developers can be independent or work within IT departments of particular manufacturers.
Manufacturing and Logistics Users
This group of users can search vApps in the vf-Store. The search can be filtered based on features, cost, or ratings amongst others. These users can request specific requirements to the vf-OS software development community when demanding custom vApps.
Manufacturing and Logistics Solution Provider
This segment exposes their ICT interfaces and manufacturing connections to the vApps. They are also able to contribute to the development of vApps that may be added and commercialised in

the vf-Store.
External Service Providers
These users provide services (hosting, storage, connected cloud services, etc.) including those based on developed solutions.

All in all, vf-OS proposes an open platform, linked by strong network externalities and exploiting advanced information and communication technology (ICT) (i.e., cyber-physical systems (CPS), IoT, cloud-models, machine to machine-M2M, security by design, etc.), fulfilling the actual need on the market for open services for interoperability based on data exchange. vf-OS provides a set of open services, rooted in the cloud, and instantiated via vf-P that move from the device-centric to the user-centric paradigm. These are implemented through a multi-sided market exploitation strategy and with clear value proposition to manufacturers, machine and device providers, logistic operators, and end users. Moreover, vf-OS profits from existing technologies (e.g., FIWARE, Talend, etc.) to quickly provide value to final users and developers.

For hardware functions, such as a factory's input and output sensors, the operating system acts as an intermediary between the application behaviour of the factory and the factory hardware itself. This enables the factory functionalities and services to be virtualised or executed directly by the hardware, allowing system calls to the operating system function to appropriately manage the manufacturing requirements.

3.2. Analysis of the Current Low-Code Development Platforms

There are only some low-code development platforms addressed to different industrial sectors. However, there is a lack of transversal platforms developed on open standards to create an ecosystem for building and deploying, in an automated manner, working apps to complete a specific task or solve a particular industrial problem. The following solutions have been identified as low-code development platforms: (i) Siemens MindSphere, (ii) PTC ThingWorx, (iii) GE Predix, (iv) IBM Cloud (formerly IBM BlueMix), (v) Microsoft Azure IOT Suite, and (vi) Software AG ADAMOS [39]. Table 4 shows the description and main characteristics of each one, including vf-OS, that will be described in the following sections.

Based on [39], analysis and comparison of the previously existing and commercial low-code development platforms have been performed against the vf-OS platform that will be described in the following section. Firstly, the main features of a low-code development platform have been identified. Based on this, the second step has been focused on analysing and performing a benchmarking among the different low-code development platforms using the following nomenclature: ● supported, ◖ supported to some extent, and - not supported. Table 5 shows the results of this analysis.

It is worth mentioning, as a limitation of this study, that this benchmarking has been performed based only on the available information of references [35, 40-45]. Moreover, the comparison is only against vf-OS and Table 4 main goal is not addressed to directly compare the other products mentioned.

Table 4. Description of the existing and low-code development platforms.

Low-code development platforms and its description
Siemens MindSphere (Version 3.0, Siemens, Berlin, Germany, 2019) [40]

MindSphere is Siemens' cloud-based, open Internet of Things (IoT) operating system that connects real things to the digital world and enables powerful industry applications and digital services to drive business success. MindSphere's open PaaS enables a rich partner ecosystem to develop and deliver new applications. MindSphere allows users to:

- Connect real things to the digital world, by including secure connectivity, an easy to set up environment, and using open standards for connectivity.
- Use Cloud-based services, PaaS, with support, open application development, and data/ metadata characterisation.
- Use built-in apps and real use-cases to shorten the deployment time, e.g., fleet management, building performance, or control loop performance analytics.

PTC ThingWorx (Version 8.5, PTC, Boston, MA, USA, 2019) [41]

ThingWorx provides the ability to source, contextualise and synthesise data while orchestrating processes and delivering powerful web, mobile and AR experiences. As claimed in their product page, ThingWorx is the fastest way to deliver industrial innovation through:

- Leverage connected and enterprise system data to improve the performance of services and customer experience, support and usability.
- Optimise business processes, combining real-time data with existing enterprise systems to increase efficiency.
- Drive new revenue streams to unlock new business models and opportunities.
- Differentiate product and service offerings to increase the pace of innovation.

GE Predix (GE, Boston, MA, USA, 2019) [42]

Predix combines sophisticated asset modelling, big data processing, analytics, and applications to provide the IT foundation for industrial operations as follows:

- Edge-to-cloud platform to deploy processing and analytics power to control edge assets in real time or analyse big data in the cloud.
- Digital twin to understand, predict, and optimise asset performance.
- Analytics and machine learning to use the industrial analytics library to create and deploy machine learning models that detect anomalies and predict maintenance.
- Applications catalogue for building custom apps and end-to-end solutions.
- Secure and resilient by design, includes edge-to-cloud data protection, security standards support, full tenant segregation and access controls.
- Comprehensive developer environment with all the right services, tools, techniques, and supporting community to rapidly develop and deploy Industrial Internet of Things (IIoT) apps.

IBM Cloud (formerly IBM BlueMix) (IBM, Armonk, NY, USA, 2019) [43]

IBM Cloud (Bluemix) is a cloud PaaS that supports several programming languages and services as well as integrated DevOps to build, run, deploy and manage applications on the cloud. Bluemix comes with a catalogue, where own and third-party applications have been uploaded for their deployment with the following features:

- IT Infrastructure, either a dedicated server or a virtual server, for load balancing, process deployment, shared storage and cloud infrastructure.
- Means for developing mobile apps and extracting all the knowledge from the data.
- Connection to different sources of data including IoT, unstructured data (through Watson) and integration of third-party sources through the application programming interface (API) Connect Secure Gateway.
- A secure environment through hardware security module and leveraging Intel trusted execution technology. All applications are SSL-Certified
- Technologies such as blockchain for developing the rules of the message hub, or containers in Kubernetes clusters for deploying secure and highly available apps.

Microsoft Azure IoT Suite (Microsoft, Redmond, WA USA, 2019) [44]

Azure IoT Suite is a set of preconfigured solutions that facilitates a quick start and can be customisable to meet specific requirements from the customer. It is an open source implementation of a common IoT solution patterns that can be deployed to Azure using the subscription means of the customer. Each preconfigured solution combines custom code and Azure services to implement a specific IoT scenario or scenarios. It features:

- Scenarios covered: data visualisation, rules and alarm configuration, device management jobs scheduling, physical and/or virtual devices provision, and troubleshooting.
- Solutions available: remote monitoring, predictive maintenance, and connected factory.
- Azure services available: IoT Hub, Event Hubs, Time series insights, container services, stream analytics, web apps, cosmos DB, Azure storage..

Software AG ADAMOS (DMG MORI, Dürr, Software AG and ZEISS as well as ASM PT, Germany, 2019)[45]

The ADAMOS IIoT platform's basic functionality is offered in the core areas of device connectivity & management, real-time analytics and visualisation, workflow automation and enterprise & cloud integration. ADAMOS is an open and manufacturer-neutral IIoT platform that envisions a world of digitally networked manufacturing, and intelligent services around existing products for machinery and plant engineering enterprises with the following features:

- Platform infrastructure, being available on demand as a cloud service. It is infrastructure-independent and can be operated flexibly according to requirements with any infrastructure provider.
- Machine learning: machine data can be evaluated in real time by machine learning models to optimise decision-making in manufacturing (e.g., predictive maintenance).
- Real-time analytics: use of predefined or individual analytics rules that are evaluated in real time on the platform and are reusable in a wide range of applications.
- Data storage: access to data within the platform (e.g., machine and sensor data).
- Security: use of state-of-the-art security concepts and standards for physical safety, network, access, and application security.
- Device connectivity: flexible connectivity options for heterogeneous machine landscapes based on certified gateways, IoT protocols and SDKs.
- Device management: device life-cycle management for efficient device usage from on- to off-boarding, such as firmware and software management, real-time alarm management, connection and configuration management.
- Dashboards: individually configurable and expandable real-time dashboards for monitoring and analysing, for example, machine status and anomalies.
- Integration: support of all IIoT integration scenarios to digitise the entire value chain, from machine to cloud integration and through applications.

Table 5 provides the foundations to give response to the research question RQ1 related to the status about existing automation software development tools. In this case, and focused on low-code development platforms, it seems that vf-OS is well positioned as software platform within IIoT domain. Of the 16 features considered, all the analysed low-code development platforms (except GE Predix), are considered as operating systems and highlight security, as one of their main features.

Moreover all the platforms provide IIoT connectivity in terms of protocols with different coverage degrees and OnPremise deployment and InCloud, with the exception of MS Azure IOT Suite, unless a further Microsoft license is acquired. A marketplace; different analytics functionalities and features covering a good spectrum of algorithms, methodologies, data ingestion, and ETL; and messaging and Pub/Sub also belong to the common features offered by all the analysed platforms.

There are some features that are well covered by many of the software platforms analysed. BlueMix and Software AG's ADAMOS up-front provide application programming interfaces (APIs) to access third party software. Within the rest of platforms, the APIs are offered through a configuration facility. In addition, MindSphere, BlueMix and ADAMOS have a good coverage in terms of development environment. However, this is not included, by default, in the MS Azure IOT Suite. In the case of PTC, they have different tools for different purposes and for GE Predix the development environment is under development.

Table 5. Benchmarking of the existing and low-code development platforms against vf-OS.

Feature	**Siemens MindSphere**	**PTC ThingWorx**	**GE Predix**	**IBM Cloud (BlueMix)**	**MS Azure IOT Suite**	**Software AG ADAMOS**	**vf-OS**
Operating System	●	●	-	●	●	●	●
Security by design	●	●	●	●	●	●	●
Connecting IIoT	●	●	●	●	●	●	●
APIs to access third party software	◖[1]	◖[1]	◖[1]	●	-	●	●
OnPremise	●	●	●	●	●	◖	●
InCloud	●	●	●	●	-	◖	●
Marketplace	●	●	●	●	-	◖[2]	●
Development Environment	●	◖[3]	◖[4]	●	-	●	●
Business Process Modelling	-	●	-	●	-	-	●
Analytics	-	●	-	●	●	●	●
Data Ingestion and ETL	◖	◖	◖	●	●	◖	●
Developers' Hub	-	-	-	-	●	-	●
Open Source	●	-	-	-	●	-	●
Innovation	●	●	●	●	●	●	●
Messaging and Pub/Sub	◖	◖	◖	●	●	●	●
Product Management, Conception, Simulation	●	●	●	●	-	-	◖[5]

◖[1] via configuration, ◖[2] through Digital Marketplace, ◖[3] different tools for different purposes, ◖[4] under development, ◖[5] via vf-OS Assets.

With regard to the main differences between the analysed platforms and vf-OS, it is worth mentioning that only two platforms: PTC ThingWorx and BlueMix have the business process modelling feature. Moreover, the developers' hub is only offered by the MS Azure IOT Suite and vf-OS. Finally, from an open source viewpoint, only Siemens MindSphere and MS Azure IOT Suite are considered (themselves) as open source platforms while Microsoft specifies that the IOT Suite is open source but not the complete version of the Azure Platform.

3.3. vf-OS Platform from the Low-Code Viewpoint

The vf-OS Platform integrates different tools and components to facilitate the development of new applications and foster development best practices, particularly code reusability, interoperability, and security-by-design. As mentioned above, the vf-OS OAK provides developers with a complete development kit to develop new applications. The vf-OS Integrated Development Environment (IDE) integrates the different development tools that facilitate application development. The micro-service architecture of vf-OS is a central part of the value proposal to developers, since it allows application functionalities to be decomposed into several micro-services and deliver specialised tools to develop specific application functionalities. There are development tools to facilitate the development of frontend micro-services and backend micro-services.

The Frontend Environment is an intuitive what you see is what you get (WYSIWYG) tool to develop application frontends. Based on the principles of web components, the frontend environment allows to compose the application frontend by combining reusable UI components (e.g., buttons, images, panels, tables, graphs) through either a palette and drag'n'drop canvas or a plain text editor with XML tags representing UI components. From this basic representation, the FrontEnd Environment generates a fully functional frontend service based on web standards and technologies such as HTML (hypertext markup language), CSS (Cascading Style Sheets) and JavaScript. Developers can visualize and test the generated code as they type, allowing to create the application frontend and connect it to backend services in minutes.

At the backend, the Process Designer allows to build the business logic of the application by drawing diagrams representing the orchestration of different vf-OS Platform micro-services. The diagrams use a business process model notation (BPMN)-like workflow, where each artefact represents a micro-service and each connection object a connection between different micro-services. The graphical user interface allows developers to drag'n'drop available micro-services to the canvas and access micro-service configuration by double-clicking the artefact, thus allowing to design complex business logic workflows with few mouse clicks.

Developers have additional tools to develop new vf-OS assets that can be later integrated into the business logic of the application, and even make them available to other developers through the marketplace under different use terms (e.g., free or licensed). In this sense, the Input/Output (IO) toolkit is a software generator that facilitates the development of gateway micro-services to interconnect the vf-OS Platform with legacy devices or software applications. The user prompts guide the developer in the creation of a micro-service scaffold that implements standard interfaces to interconnect to other vf-OS components. Developers can use available templates to integrate well-known communication standards like the object linking and embedding for process control unified architecture (OPC UA) or Open Data Protocol (OData) in a secure way, reusing already tested libraries. The Data Analytics model generator facilitates the development of analytic services, including machine learning services, providing development tools and user interfaces to train and test models. Data analytics assets are packaged as simple objects that can loaded and executed in the Data Analytics component. The data integration generator allows users to create complex data extraction-transformation-load (ETL) processes using a graphical user interface to define data integration workflows. Again, data integration assets are distributed as objects that can be loaded and executed in the Data Harmonisation component. These generators are integrated into the vf-OS Platform in such a way that allows developers to build, deploy, test, and validate new vf-OS assets locally before they are eventually published to the marketplace. For this purpose, the vf-OS platform implements a staging area to test new micro-services and means to deploy them locally. Additionally, developers can publish their newly developed assets in the curated vf-OS Marketplace, so that they can be used by other applications.

All in all, it seems that low-code development platforms are the future direction for time and cost reduction in the software development context. However, and in order to give response to research question RQ2, it is worth mentioning that aspects such as integration, interoperability, communication, real-time data processing, homogenisation, ergonomics and security to mention some, should be addressed properly to facilitate adoption of the low-code paradigm.

vf-OS has been tested in three different manufacturing domains: manufacturing and logistics/automation, construction-industrialisation, and manufacturing assembly and collaboration. In these domains, a total of 21 vApps (smart manufacturing applications) have been developed involving topics such as failure prevention and monitoring, collaboration analysis, quality control, product identification or document management. These vApps have been developed to meet users' needs and, as such, they have also been validated against their needs and metrics. The vf-OS platform has been applied in the European project "Zero Defects Manufacturing Platform" (ZDMP) for the development of apps related to products' and processes' quality to support industrial companies with the achievement of zero-defects in manufacturing.

To offer an overview of the ease with which a vApp is developed, the steps for its development are described as follows: (i) developer (D) logs in to the vf-OS Platform, where D has access and opens the vf-Studio; (ii) D uses the different components to create the vApps, mainly the process designer, for the skeleton or backend, and the frontend environment, for the UI. When the skeleton is being created, D has access to the vf-Store for purchasing vf-OS assets that have some packed functionality, such as access to machine-learning or ETL routines, or access to drivers, etc.; (iii) D uses then the vf-Studio again to finish off the development of the vApp (e.g., bug fixing or test-run of the vApp) before publication; (iv) once the vApp has been stabilised, D can publish the vApp in the vf-Store providing details such as the price or description; (v) industrial user, (IU) access the vf-Store and seek for a vApp that does what they need. If no vApp is found, then the IU can create a request for a new vApp (which then reaches a developer that proceeds following the steps before); (vi) IU purchases the desired vApp and, after payment, UI receives a link to download the vApp; and (vii) IU logs into their vf-OS platform and, through the system dashboard, they can install the purchased vApp. When this process is compared to the traditional one, some benefits such as rapidity, cost and complexity reduction arise.

4. Conclusions

Given the scant references found in the literature related to low-code, it can be stated that low-code is still a young emerging technology. Forrester Research forecasts [29] total spending on low-code issues of $21.2 billion by 2022. If the forecast is accurate, it seems that low-code development platforms will be regularly accepted as a common mean to build apps.

During the development of this analysis, aspects such as improved agility, cost reduction, faster apps building and delivery, among others have been highlighted as key factors in the automation of software development tools. To this end, CASE technology emerged in the late 1980s, however its poor adoption has currently led to search for alternative technologies such as the low-code development platforms. However, it is worth mentioning that the use of low-code development platforms also presents some barriers. Humans are by nature reluctant to change and the universal adoption of low-code may be diminished due to the resistance to change by companies. Otherwise, the impatience in its adoption can also lead to 'shadow IT' since the IT department of a company does not test and approve the technology that the company is using. This can also lead to risks with compliance and security. The future of low-code paradigm is still uncertain. So, to provide some certainty in this domain, and after analysing the literature, it can be concluded that the low-code technology as an automation software development tools is under-researched.

As the results from the literature are scant, the main features that a low-code development platforms should have and offer (open source, developers' hub, messaging and Pub/Su) have been identified. Then, benchmarking among existing low-code development platforms has been performed against vf-OS. This has resulted in a set of findings that highlight the lack of transversal platforms developed on open standards to create ecosystems for building and deploying, in an automated manner, apps. All this indicates that the issues addressed in the research question RQ1 have been answered.

Based on these findings, vf-OS platform emerges to facilitate the app-building process and to not be dependent on off-the-shelf and third-party software.

Based on the RQ2, one of the key challenge is to address the issues of real-time data processing these applications require [6]. In light of this, the fundamental aspect for further study is the linkage between low-code development platforms and IoT devices to consolidate Industry 4.0. Doing so requires new further attempts focused on low-code research for the apps development that facilitates the communication and optimisation of intelligent, interconnected equipment and products along the entire value chain. To this end, vf-OS was born. vf-OS is an open multi-sided framework able to manage the overall network of a collaborative manufacturing and logistics environment that enables humans, applications, and devices (IoT) to seamlessly communicate and interoperate in the interconnected environment, promoting digital transformation. vf-OS may be considered as an example from where future research efforts should be directed, taking into account its multi-sided and transversal characteristics. In light of this, one of the main challenges of the current approach is the integration of the different development environments in a single IDE. To provide a satisfactory user experience for developers, it is necessary to deliver a high-level of integration between the different development tools, not only enabling access to the different tools in an integrated environment, but also implementing the same look and feel design and usability considerations across different tools to improve user experience. Also, the development of wizards that guide developers through the development steps across the different development tools can improve significantly the user experience when developing new applications.

Another important challenge is the integration of secure code analysis into the development workflows. Currently, the development tools provide static security code analysis reports when building new applications, based on the source code provided. However, there is no integration with dynamic application security analysis tools that test vulnerabilities during runtime. Integration of this type of tools into the vf-OS Platform via the vf-OS Privacy and Security component is another line of future work.

Finally, experts predict that low-code will be possibly crucial for working efficiently and being competitive. In the literature, there are examples of many companies that saw their business continuity in danger because they were not resilient enough and they did not adapt quickly to the new wave of digital technology promoted by Industry 4.0. Enterprises need to develop new digital solutions much faster than their competitors and automation software development tools such as the low-code development platforms may be the response to these new companies' digital requirements.

Author Contributions: Conceptualization, R.S. and R.P.; Investigation, F.F., O.G.-P., R.S.; Methodology, R.S. and R.P.; Project administration, R.P.; Resources, R.P.; Supervision, R.P. and O.G.-P.; Writing—original draft, F.F., O.G.-P. R.S.; Writing—review and editing, F.F., R.S., R.P. and O.G.-P. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported in part by the European Commission under the Grant Agreements No. 723710 and 825631.

Acknowledgments: The authors would like to acknowledge the support of the researchers participating in the collaborative projects "Virtual Factory Open Operating System" (vf-OS) (www.vf-os.eu) and "Zero Defects Manufacturing Platform (ZDMP) (www.zdmp.eu).

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Sanchis, R.; Poler, R. Enterprise Resilience Assessment-A Quantitative Approach. *Sustainability* **2019**, *11*, 4327.
2. Tisi, M.; Mottu, J.M.; Kolovos, D.; De Lara, J.; Guerra, E.; Di Ruscio, D.; Pierantonio, A.; Wimmer, M. Lowcomote: Training the Next Generation of Experts in Scalable Low-Code Engineering Platforms. 2019. Available online: https://www.se.jku.at/lowcomote-training-the-next-generation-of-experts-in-scalable-low-code-engineering-platforms/ (accessed on 4 December 2019).
3. OutSystems. *The State of Application Development. Is IT Ready for Disruption?* OutSystems: Boston, MA, USA, 2019.

4. Richardson, C.; Rymer, J.R. *New Development Platforms Emerge For Customer-Facing Applications*; Forrester: Cambridge, MA, USA, 2014.
5. Waszkowski, R. Low-code platform for automating business processes in manufacturing. *IFAC-PapersOnLine* **2019**, *52*, 376–381.
6. Richardson, C.; Rymer, J.R. *Vendor Landscape: The Fractured, Fertile Terrain of Low-Code Application Platforms*; Forrester: Cambridge, MA, USA, 2016.
7. Tarasiev, A.; Filippova, M.; Aksyonov, K.; Aksyonova, O. Developing Prototype of CASE-Tool to Create Automation Systems Based on Web Applications Using Code Generation. In Proceedings of the XII International Scientific and Technical Conference Dynamics of Systems, Mechanisms and Machines, Omsk, Russia, 13–15 November 2018; IEEE: Piscataway, NJ, USA, 2019; pp. 1–4.
8. Al-ashwal, D.; Al-Sewari, E.Z.; Al-Shargabi, A.A. A CASE Tool for JAVA Programs Logical Errors Detection : Static and Dynamic Testing. In Proceedings of the 2018 International Arab Conference on Information Technology (ACIT), Sidon, Lebanon, 28–30 November 2018; pp. 1–6.
9. Glover, N.; Dudley, T. *Practical Error Correction Design for Engineers*; Data Systems Technology Corporation: Reston, VA, USA, 1991.
10. Lundell, B.; Lings, B. Changing perceptions of CASE technology. *J. Syst. Softw.* **2004**, *72*, 271–280.
11. Fuggetta, A. A Classification of CASE Technology. *Comput. J.* **1993**, *35*, 25–38.
12. Vileiniskis, T.; Skersys, T.; Pavalkis, S.; Butleris, R.; Butkiene, R. Lightweight approach to model traceability in a CASE tool. In *AIP Conference Proceedings*, *Proceedings of the ICNAAM 2017*, *Thessaloniki, Greece*, *25–30 September 2017*; American Institute of Physics: College Park, MD, USA, 2017; Volume 1863.
13. Madoš, B.; Ádám, N.; Baláz, A.; Šinal'ová, K. The CASE tool for programming of the multi-core System-on-a-Chip with the data flow computation control. In Proceedings of the IEEE 15th International Symposium on Applied Machine Intelligence and Informatics (SAMI 2017), Herl'any, Slovakia, 26–28 January 2017; pp. 165–168.
14. Sassi, M.S.H.; Jedidi, F.G.; Fourati, L.C. Computer-aided software engineering (CASE) tool for big data and IoT architecture. In Proceedings of the 2019 15th International Wireless Communications & Mobile Computing Conference (IWCMC), Tangier, Morocco, June 24–28, 2019; pp. 1403–1410.
15. Troy, D.; McQueen, R. An approach for developing domain specific CASE tools and its application to manufacturing process control. *J. Syst. Softw.* **1997**, *38*, 165–192.
16. Huff, C.C. Elements of a realistic CASE tool adoption budget. *Commun. ACM* **1992**, *35*, 45–55.
17. Orlikowski, W.J. CASE Tools as Organizational Change: Investigating Incremental and Radical Changes in Systems Development. *MIS Q.* **1993**, *17*, 309–340.
18. Iivari, J. Why Are CASE Tools Not Used? *Commun. ACM* **1996**, *39*, 94–103.
19. Land, O.; Der Zanden, V. Enterprise Design—A practice-driven response for generating and implementing business models. In Proceedings of the 8th Enterprise Engineering Working Conference (EEWC 2018), Luxembourg, 28 May–1 June 2018.
20. Zolotas, C.; Chatzidimitriou, K.C.; Symeonidis, A.L. RESTsec: A low-code platform for generating secure by design enterprise services. *Enterp. Inf. Syst.* **2018**, *12*, 1007–1033.
21. Henriques, H.; Lourenço, H.; Amaral, V.; Goulão, M. Improving the developer experience with a low-code process modelling language. In Proceedings of the 21th ACM/IEEE International Conference on Model Driven Engineering Languages and Systems, Copenhagen, Denmark, 14–19 October 2018; pp. 200–210.
22. Gavrila, V.; Bajenaru, L.; Dobre, C. Modern Single Page Application Architecture: A. Case Study. *Stud. Inform. Control* **2019**, *28*, 231–238.
23. Pantelimon, S.G.; Rogojanu, T.; Braileanu, A.; Stanciu, V.D.; Dobre, C. Towards a seamless integration of iot devices with iot platforms using a low-code approach. In Proceedings of the IEEE 5th World Forum Internet Things, WF-IoT 2019, Limerick, Ireland, 15–18 April 2019; pp. 566–571.
24. Wu, Y.; Wang, S.; Bezemer, C.P.; Inoue, K. How do developers utilize source code from stack overflow? *Empir. Softw. Eng.* **2019**, *24*, 637–673.
25. Hamming, R.W. Error detecting and error correcting codes. *Bell Syst. Tech. J.* **1950**, *29*, 147–160.
26. Tse, D. *Fundamentals of Wireless Communication*[1]; Cambridge University Press: Cambridge, UK, 2004.
27. OutSystems. *Build Enterprise-Grade Apps Incredibly Fast;* OutSystems: Boston, MA, USA, 2019. Available online: https://www.outsystems.com/ (accessed on 18 September 2019).
28. Forrester. Available online: https://go.forrester.com/ (accessed on 18 September 2019).

29. Rymer, J.R.; Koplowitz, R. *The Forrester Wave™: Low-Code Development Platforms for AD&D Professionals, Q1 2019*; Forrester: Cambridge, MA, USA, 2019.
30. Fryling, M. Low code app development. *J. Comput. Sci. Coll.* **2019**, *34*, 119.
31. Clancy, T. *The Standish Group Chaos Report*; Standish Group: West Yarmouth, MA, USA, 2014.
32. Jost, B.; Ketterl, M.; Budde, R.; Leimbach, T. Graphical programming environments for educational Robots: Open Roberta—Yet another one? In Proceedings of the 2014 IEEE International Symposium on Multimedia ISM 2014, Taichung, Taiwan, 10–12 December 2014; pp. 381–386.
33. Dehouck, R.E.M.I. The Maturity of Visual Programming. Режим доступу. 2015. Available online: http://www. craft. ai/blog/the-maturity-of-visualprogramming (accessed on 17 December 2019).
34. Bragg, S.D.; Driskill, C.G. Diagrammatic-graphical programming languages and DoD-STD-2167A. In Proceedings of the AUTOTESTCON '94, Anaheim, CA, USA, 20–22 September 1994; pp. 211–220.
35. Virtual Factory Operating System. 2019. Available online: www.vf-OS.eu (accessed on 2 September 2019).
36. Thompson, H. Road4FAME Consultation Event "Digital Revolution in Europe: Converging Visions for a Smarter World". In Proceedings of the Road4FAME Consultation, Brussels, Belgium, 22 May 2015.
37. García-Perales, O. vf-OS D1.1: Vision Consensus. 2017. Available Online: https://www.vf-os.eu/results (accessed on 20 October 2019).
38. vf-OS Wiki. Available online: https://cigipsrv1.cigip.upv.es:4430/mediawiki/index.php/Wiki_Home (accessed on 10 September 2019).
39. García-Perales, O. vf-OS D2.1: Global Architecture Definition. 2017. Available Online: https://www.vf-os.eu/results (accessed on 10 October 2019).
40. Siemens MindSphere. 2019. Available online: https://new.siemens.com/vn/en/products/software/mindsphere.html (accessed on: 10 November 2019).
41. PTC ThingWorx Platform. 2019. Available online: https://www.ptc.com/en/resources/iiot/product-brief/thingworx-platform (accessed on 20 October 2019).
42. GE Predix. 2019. Available online: https://www.ge.com/digital/iiot-platform (accessed on 3 October 2019).
43. IBM Cloud. 2019. Available online: https://www.ibm.com/cloud (accessed on 30 September 2019).
44. Microsoft Azure IOT Suite. 2019. Available online: https://azure.microsoft.com/es-es/blog/microsoft-azure-iot-suite-connecting-your-things-to-the-cloud/ (accessed on 10 November 2019).
45. Software AG ADAMOS. 2019. Available online: https://www.softwareag.com/corporate/company/adamos/default.html (accessed on: 17 November 2019).